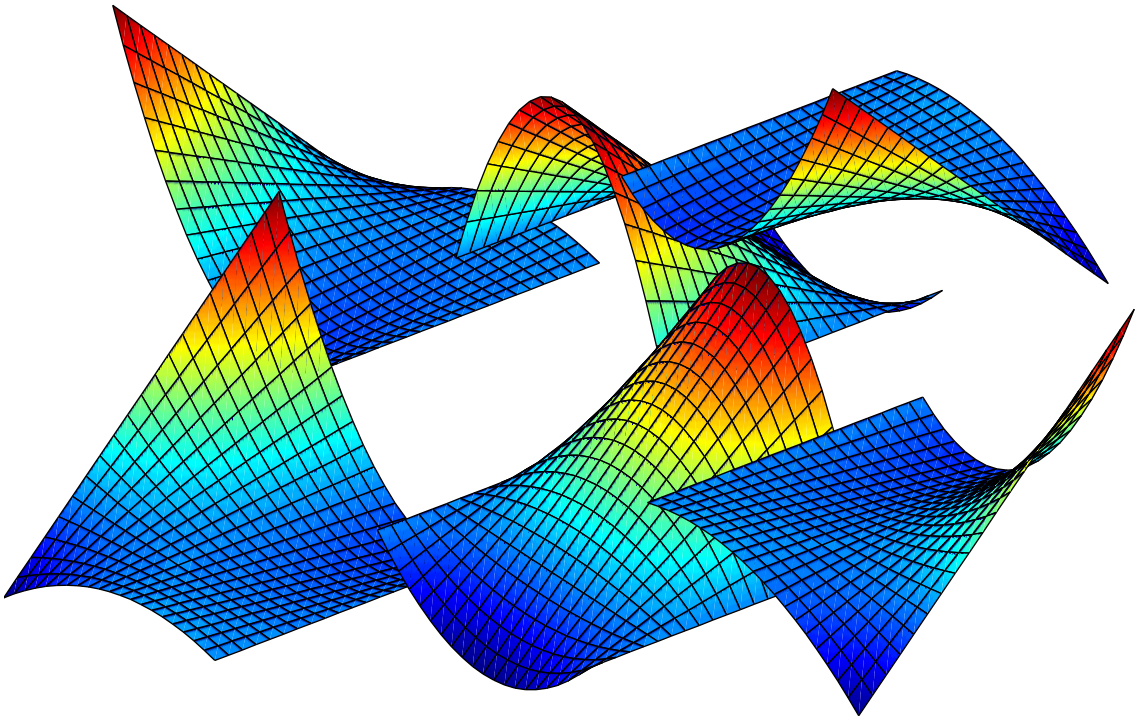
Mimetic Spectral Element Method for Elliptic Problems

Bouman M.P.



February 10, 2010

Aerodynamics

Faculty of Aerospace Engineering

TU Delft

Delft University of Technology

Mimetic Spectral Element Method for Elliptic Problems

Master of Science Thesis

For obtaining the degree of Master of Science in Aerospace Engineering
at Delft University of Technology

Bouman M.P.

February 10, 2010

Faculty of Aerospace Engineering · Delft University of Technology

Delft University of Technology

DELFT UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF AERODYNAMICS

The undersigned hereby certify that they have read and recommend to the Faculty of Aerospace Engineering for acceptance the thesis entitled **"Mimetic Spectral Element Method for Elliptic Problems"** by **Bouman M.P.** in fulfillment of the requirements for the degree of **Master of Science**.

Dated: February 10, 2010

Supervisors:

dr.ir. M.I. Gerritsma

prof. dr. ir. drs. H. Bijl

Prof. dr. ir. K. Vuik

Ir. J. J. Kreeft

Msc. A. Palha

Preface

This thesis presents my work to obtain the degree of Master of Science in Fluid Dynamics at the Faculty of Aerospace Engineering of Delft University of Technology (DUT), the Netherlands. My work here began October 2008 when I came to my supervisor dr.ir. M.I. Gerritsma to talk about subject for my final thesis. I had just finished working on an assignment which introduced me to the use of Spectral Element Methods for solving partial differential equations. This caught my interest and following the work of P.B. Bochev and J.M. Hyman we started to look into compatible discretizations, but for Spectral Element methods.

After a few months studying and trying out all sorts of configurations and element types we were starting to get encouraging results. When we tried to switch to deformed elements we noticed that the theory at hand was not sufficient. We could not use vector calculus to derive the correct discretization. Thanks to the many discussions with my supervisor and two PhD student J. Kreeft and A. Palha we decided that it was best to leave vector calculus for what it was and started to use differential geometry to describe the equation that we tried to solve. This meant quite a big leap forward and spawned the idea of the using different types of basis functions for different types of variables, which is one of the key points in this work.

Without such close contact with the aforementioned people it would not have been possible to work on such a fundamental and interesting topic, of which I was lucky enough to have been allowed to give a presentation at the mathematical conference ICOSAHOM'09 and produce a, as of yet not published, paper. Therefore my thanks go out to ir. J. Kreeft for the almost daily chats about this (and other) topics, A. Palha for the many discussions especially in the beginning of my work and the interesting conference in Trondheim. Special thanks go out to dr.ir. M.I. Gerritsma for his interest in my work and the input that he has given which helped shape this thesis to what it is.

Abstract

Solving Partial Differential Equations (PDE's) numerically requires that the PDE or system of PDE's be replaced with a system of algebraic equations. The replacing system of algebraic equation should be mimetic in the sense that discrete operators that make up the PDE mimic the vector identities that connect the continuous operators.

The equation that we focus on is the Poisson equation. The Poisson equation can be split up into two first order equations, where one equation is the divergence relation for some conserved quantity. We then rewrite this system of equations in terms of differential geometry. The advantages of using differential geometry is twofold. The first is that there is a very obvious link to its discrete counterpart which is algebraic topology. Second is that the mapping of spaces and the functions defined in these spaces is very well defined. With these properties we are able to derive a compatible (mimetic) discretization for the Poisson equation for arbitrarily shaped curved spectral elements. On these curved elements we are able to retain exponential convergence.

In creating a compatible discretization we have to introduce a type of basis function that does not reconstruct the continuous field from nodal values at collocation points, but rather reconstruct a continuous field from integrated values. We call these functions, edge functions, because of their connection with edges rather than nodes. Making use of these edge functions the method shows absolute conservation for arbitrary element order and arbitrarily shaped elements.

With a minor adaptation we can extend the method to the more general elliptic type problem, especially anisotropic diffusion problems like Darcy flow. In both of these cases the use of multiple elements follows quite naturally, and also in the multi-element case we retain theoretical convergence rates and absolute conservation.

Table of Contents

Preface v

Abstract vii

1 Introduction 1

1.1 Thesis Outline . . . 2

2 Differential geometry and Algebraic topology 3

2.1 Differential forms . . . 4

2.2 The exterior- (wedge-) product . . . 5

2.3 Exterior derivative and Stokes's theorem . . . 5

2.4 The Hodge $\star$ operator . . . 7

2.5 Mappings . . . 9

2.6 Algebraic topology . . . 10

2.7 Differential geometry and algebraic topology connected . . . 14

3 Basis Functions 15

3.1 Nodal basisfunctions . . . 15

3.2 Edge functions . . . 17

4 Poisson equation on orthogonal domains 19

4.1 Poisson equation . . . 19
4.2 Support operator . . . 20
4.3 Definition of the spectral element dual cell-complexes . . . 21
4.4 The conservation law $dq^{(1)} = f^{(2)}$ . . . 23
4.5 Discretization of $\phi^{(0)}$ . . . 25
4.6 Formulation of the discrete grad operator . . . 27

5 Results on orthogonal domain **33**
5.1 Boundary conditions and solution procedure . . . 33
5.2 Error norm definitions . . . 34
5.2.1 Evaluation error $u^{(1)}$ . . . 35
5.2.2 Evaluation error ϕ . . . 36
5.3 Test case for orthogonal domain . . . 36
5.3.1 Convergence results . . . 38
5.3.2 Conservation . . . 39
5.4 Comparison error formulation . . . 40

6 Poisson equation on non-orthogonal domains **43**
6.1 Support operator with mapping . . . 43
6.2 Integral Evaluation . . . 44
6.2.1 Inner product . . . 45
6.2.2 Discretization of $\phi^{(0)}$ and $q^{(1)}$ and $f^{(2)}$ . . . 46

7 Results on non-orthogonal domains **51**
7.1 Errors on deformed domains . . . 51
7.2 Case 1: Linear deformation . . . 52
7.3 Case 2: Harmonic deformation . . . 54

8 Multi element **57**

8.1 Cell-complex definition . . . 57
8.2 Test cases . . . 60
8.2.1 Exponential convergence . . . 61
8.2.2 h-convergence . . . 62
8.3 Flow around cylinder . . . 63

9 Anisotropic diffusion **67**
9.1 Symmetric tensor $\mathbb{K}$ . . . 67
9.1.1 Introducing $\mathbb{K}$ . . . 67
9.1.2 Flux operator . . . 68
9.2 Anisotropic diffusion . . . 69
9.2.1 Discontinuous $\mathbb{K}$ in element . . . 69
9.2.2 Discontinuous $\mathbb{K}$ on boundary . . . 70

10 Conclusions and recommendations **73**
10.1 Conclusions . . . 73
10.2 Recommendations . . . 74

Bibliography **75**

Bibliography **75**

A Lagrangian polynomials **77**
A.1 Legendre polynomials . . . 78
A.2 Gauss-Lobatto . . . 78
A.3 Gauss . . . 80
A.4 Extended Gauss . . . 80
A.5 Gauss-Lobatto-Legendre edge function . . . 82

B Extended Gauss Interpolation **85**

B.1 Testing first candidate 87
B.2 Testing second candidate 88
B.3 Final Candidate 90

C Matrix construction **93**

D Numerical integration of line segments and surfaces **99**

E Mappings and Forms **101**

E.1 0-forms 101
E.2 1-forms 102
E.3 2-forms 103

F Large equations **105**

Chapter 1

Introduction

Science and engineering rely highly on the use of Partial Differential Equations (PDEs) to model problems in the fields of, electromagnetism, chemistry, nuclear physics and aerodynamics. For practical problems these equations do not have exact analytical solutions, and must therefore be solved numerically. In order to numerically solve a PDE we must replace it by a system of algebraic equations. It is paramount that this system of algebraic equations yield a compatible (mimetic) and physically consistent system. That the system should be physically consistent is an obvious property, if this were not the case the obtained solution would not represent the physical solution.

We propose a discretization that is directed at being mimetic. The term mimetic means that the discrete system of equations, with which the PDE is replaced, preserves analytical relations such as vector identities and integral theorems. Now instead of using vector calculus to describe our Partial Differential Equation we use differential geometry. In differential geometry we no longer work with scalars and vectors, these are replaced by the so called differential forms. In differential geometry the standard operators from vector calculus **div**,**grad** and **rot**) are replaced with the single operator **d** that operates on the differential forms.

We restrict ourselves to solving partial differential equations that have their basis in the conservation law $\mathbf{div}\mathbf{q} = f$. By introducing a potential (ϕ) and defining $\mathbf{q} = \mathbf{grad}\phi$ we have the system that makes up the Poisson equation. First we define a discrete **div** operator with which we derive the discrete **grad** operator. Because we derive the **grad** operator implicitly we respect the vector identities and integral theorems that connect these operators, thus mimicking the structure of the PDE. The resulting system of equations is always symmetric, as is the Poisson equation. Similar, but low order, approaches have been studied by, Lipnikov and Gyra (2008), Bochev and Hyman (2006) and Bochev (2003). Besides using differential geometry to describe the Poisson equation we discretize differential forms and the equations on spectral elements. Spectral elements are elements on which the differential forms are approximated using high order polynomial basis functions. Using differential geometry and its discrete counterpart, algebraic topology, it is natural to discretize the vector **q** as the

flux integrated over a surface, as proposed by Gerritsma (2009). In doing so, the obtained solution for **q** satisfies the conservation relation identically. The most interesting on this topic is that the use of differential geometry shows that the property of exact divergence also holds for curvilinear domains.

1.1 Thesis Outline

In Chapter 2 we discuss the necessary basics of differential geometry and algebraic topology, k-forms are explained by their analogues in vector calculus. When these concepts of differential geometry and algebraic topology are explained we show, for undeformed grids, how to use the **div** operator as a support for the **grad** operator. This again is done both using differential geometry and vector calculus, again to make the reader more accustomed to the notation of differential geometry. In Chapter 4 we define, using algebraic topology as our guide, the spectral element on which we discretize these operators. We test the resulting system and will show that we obtain spectral convergence and satisfy the divergence equation up to machine accuracy. All of this is done on a square orthogonal grid. In Chapter 6 and 7 the true advantage of differential geometry shows when we expand the method to arbitrary curvilinear domains. Here we speak of transforming differential forms from one domain to another and what this does to their values. In Chapter 8 we explain that with the tools available for single element calculations, one can logically expand the method to the use of multiple elements. Here we show that we retain the property of exact conservation. We also introduce the use of a symmetric tensor $\mathbb{K}$ with which we can expand the method such that we can compute Darcy flows (Neuman (1977)) or general anisotropic diffusion problems.

Chapter 2

Differential geometry and Algebraic topology

In this chapter we want to give the reader an overview of differential geometry and its discrete counterpart which is algebraic topology. We do this so that we can rewrite the Poisson equation in terms of differential geometry. Using this formulation we have the basis to create a mimetic discretization for the Poisson equation. Using terminology from vector calculus one writes the elliptic Poisson equation as,

$$\Delta\phi = f \ , \tag{2.1}$$

which we can rewrite as the following system of first order equations,

$$\nabla\phi = \mathbf{u} \quad \text{or,} \quad \text{grad}\phi = \mathbf{u} \tag{2.2a}$$
$$\nabla\cdot\mathbf{q} = f \quad \text{or,} \quad \text{div}\mathbf{q} = f \tag{2.2b}$$
$$\mathbf{q} = \mathbf{u}. \tag{2.2c}$$

Here we make a distinction between $\mathbf{q}$ and $\mathbf{u}$ only later to equate them. This is for a reason, as it will become clear to the reader that $\mathbf{q}$ and $\mathbf{u}$ are not equal when described using differential geometry. We present the basics of differential geometry by using analogies from vector calculus and some simple examples. The power of differential geometry partially lies in the fact that its formal definitions hold for arbitrary dimensional spaces $\mathbb{R}^n$. These definitions however are rather general and, for someone unaccustomed to differential geometry, not too clarifying. This is why we present examples here that hold in $\mathbb{R}^3$, a 3-dimensional space. We discuss differential forms, the exterior product for multiplication of differential forms, the exterior derivative, the Hodge $\star$ operator and the mapping of differential forms from one domain to another. For a deeper understanding of differential geometry we refer to Flanders (1962) and Bochev (2003).

2.1 Differential forms

In $\mathbb{R}^3$ there are 4 types of differential- (k-) forms, the first is the 0-form. The 0-form is a function and in vector calculus is known as the scalar valued function like a velocity potential or a temperature. It is associated with points in a domain. Let's say the coordinates in $\mathbb{R}^3$ are x,y and z, then the zero form $\phi^{(0)}$ is given by (2.3).

$$\phi^{(0)} = \phi(x, y, z). \tag{2.3}$$

The second form in $\mathbb{R}^3$ is the 1-form, typical of the 1-form is that it is associated with line segments, more precisely with line integrals. It can be associated with a vector, e.g. a velocity or temperature gradients. The 1-form $u^{(1)}$ is defined by (2.4)

$$u^{(1)} = u_x dx + u_y dy + u_z dz. \tag{2.4}$$

Where one can make a one to one correspondence between 1-forms and vectors in the following way,

$$u^{(1)} = u_x dx + u_y dy + u_z dz \quad \longrightarrow \quad \mathbf{u} = u^x \mathbf{e}_x + u^y \mathbf{e}_y + u^z \mathbf{e}_z, \tag{2.5}$$

and in the case of $\mathbb{R}^3$, $u_x = u^x; u_y = u^y; u_z = u^z$. dx,dy and dz play the same role for the 1-form as $\mathbf{e}_x, \mathbf{e}_y$ and $\mathbf{e}_z$ play for the vector. It is common to refer to the vector $\mathbf{u}$ as the proxy of the 1-form $u^{(1)}$.

The third form in $\mathbb{R}^3$ is the 2-form. Typical of 2-forms is that they are associated with surfaces, more precisely, with surface integrals. So in differential geometry one discriminates between different types of vectors, one is associated with lines and the other with surfaces. An example of a 2-form is,

$$\omega^{(2)} = \omega_x dydz + \omega_y dzdx + \omega_z dxdy, \tag{2.6}$$

Again it is possible to establish a one to one correspondence between a 2-form and a vector.

$$\omega^{(2)} = \omega_x dydz + \omega_y dxdz + \omega_z dxdy \longrightarrow \quad \omega = \omega^x \mathbf{e}_x + \omega^y \mathbf{e}_y + \omega^z \mathbf{e}_z, \tag{2.7}$$

where in this case $dxdy$,$dxdz$ and $dydz$ play the same role for 2-forms as $\mathbf{e}_x, \mathbf{e}_y$ and $\mathbf{e}_z$ play for the vector. Again in $\mathbb{R}^3$, $\omega_x = \omega^x; \omega_y = \omega^y; \omega_z = \omega^z$. Due to this bijection we can also refer to vectors as proxies for 2-forms. We see that where in vector calculus no distinction is made between the 2 types of vectors, in differential geometry one does discriminate between these 2 types.

The fourth and final k-form in $\mathbb{R}^3$ is the 3-form. This is the equivalent of a scalar value in vector calculus. It is however associated with a volume. A typical example of a 3-form is density. As an example, the 3-form $f^{(3)}$ that is used for the Poisson equation is given by (2.8)

$$f^{(3)} = f dxdydz. \tag{2.8}$$

Note that the value f here is associated with the volume $dxdydz$.

2.2 The exterior- (wedge-) product

In differential geometry multiplication of differential forms is indicated with the wedge product. The wedge product is multi-linear and has the following property

$$a^{(l)} \wedge b^{(k)} = (-1)^{kl}\, b^{(k)} \wedge a^{(l)}. \tag{2.9}$$

This indicates that the wedge product is antisymmetric, as a consequence we have that $a \wedge a = 0$. A product of a k-form and an l-form is a (k+l)-form,

$$a \in \Lambda^k(\Omega), \quad b \in \Lambda^l(\Omega) \quad \longrightarrow \quad a \wedge b \in \Lambda^{k+l}(\Omega). \tag{2.10}$$

We can show that the wedge product for two 1-forms, in $\mathbb{R}^3$, behaves just like the cross product. Say we take $a^{(1)} \wedge b^{(1)}$,

$$\begin{aligned} a^{(1)} \wedge b^{(1)} &= (a_x dx + a_y dy + a_z dz) \wedge (b_x dx + b_y dy + a_z dz) \\ &= a_x b_x \underbrace{dx \wedge dx}_{=0} + a_x b_y dx \wedge dy + a_x b_z \underbrace{dx \wedge dz}_{=-dz \wedge dx} + ... \\ &\quad a_y b_x \underbrace{dy \wedge dx}_{=-dx \wedge dy} + a_y b_y \underbrace{dy \wedge dy}_{=0} + a_y b_z dy \wedge dz + ... \\ &\quad a_z b_x dz \wedge dx + a_z b_y \underbrace{dz \wedge dy}_{=-dy \wedge dz} + a_z b_z \underbrace{dz \wedge dz}_{=0} \\ &= (a_x b_y - a_y b_x) dx \wedge dy + (a_z b_x - a_x b_z) dz \wedge dx + (a_y b_z - a_z b_y) dy \wedge dz, \end{aligned} \tag{2.11}$$

which shows a one to one correspondence with,

$$\begin{vmatrix} \mathbf{e_x} & \mathbf{e_y} & \mathbf{e_z} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} = (a_x b_y - a_y b_x)\mathbf{e_z} + (a_x b_z - a_z b_x)\mathbf{e_y} + (a_y b_z - a_z b_y)\mathbf{e_x}. \tag{2.12}$$

So the wedge product behaves just like the exterior (cross) product of the proxies of the 1-forms. As a consequence of (2.9) we can also state the following.

$$\omega^{(l)} \wedge \omega^{(k)} = 0, \quad \text{for} \quad k + l > n. \tag{2.13}$$

Meaning that the wedge product of two k-forms $\omega^{(l)}$ and $\omega^{(k)}$ whose dimension $(l + k)$ is greater than n is equal to zero. In lengthy expressions we often abbreviate a product like $dx \wedge dy$ by $dxdy$, as in (2.6) and (2.8).

2.3 Exterior derivative and Stokes's theorem

In differential geometry differentiation is performed using the exterior derivative d. The exterior derivative d is a map and takes a k-form into a k+1-form. Depending on the k-form

on which it is operating, it behaves like **div**, **grad** or **curl** on its proxy. When operating on a 0-form we have,

$$d\phi^{(0)} = \frac{\partial\phi}{\partial x}dx + \frac{\partial\phi}{\partial y}dy + \frac{\partial\phi}{\partial z}dz, \tag{2.14a}$$

whose vector proxy is,

$$\nabla\phi = \frac{\partial\phi}{\partial x}\mathbf{i} + \frac{\partial\phi}{\partial y}\mathbf{j} + \frac{\partial\phi}{\partial z}\mathbf{k}. \tag{2.14b}$$

We see that the exterior derivative applied to a 0-form as the same as taking the gradient of a scalar field in $\mathbb{R}^3$ if one looks at the proxy. The exterior derivative applied to a 1-form $u^{(1)} = u_x dx + u_y dy + u_z dz$ is defined as:

$$\begin{aligned} du^{(1)} &= du_x \wedge dx + du_y \wedge dy + du_z \wedge dz && (2.15a)\\ &= \frac{\partial u_x}{\partial x}\underbrace{dx \wedge dx}_{=0} + \frac{\partial u_x}{\partial y}dy \wedge dx + \frac{\partial u_x}{\partial z}dz \wedge dx + ... \\ &\quad \frac{\partial u_y}{\partial x}dx \wedge dy + \frac{\partial u_y}{\partial y}\underbrace{dy \wedge dy}_{=0} + \frac{\partial u_y}{\partial z}dz \wedge dy + ... \\ &\quad \frac{\partial u_z}{\partial x}dx \wedge dz + \frac{\partial u_z}{\partial y}dy \wedge dz + \frac{\partial u_z}{\partial z}\underbrace{dz \wedge dz}_{=0} \\ &= (\frac{\partial u_z}{\partial y} - \frac{\partial u_y}{\partial z})dy \wedge dz + (\frac{\partial u_x}{\partial z} - \frac{\partial u_z}{\partial x})dz \wedge dx + (\frac{\partial u_y}{\partial x} - \frac{\partial u_x}{\partial y})dx \wedge dy, && (2.15b) \end{aligned}$$

whose vector proxy again is the vector,

$$\nabla \times \mathbf{u} = (\frac{\partial u_z}{\partial y} - \frac{\partial u_y}{\partial z})\mathbf{i} + (\frac{\partial u_x}{\partial z} - \frac{\partial u_z}{\partial x})\mathbf{j} + (\frac{\partial u_y}{\partial x} - \frac{\partial u_x}{\partial y})\mathbf{k}. \tag{2.15c}$$

The exterior derivative applied to this 1-form acts as taking the curl of a vector proxy. As a generalisation of the product rule for differentiation we can also take the exterior derivative of a product of k-forms,

$$d(\omega^k \wedge \omega^l) = (d\omega^k) \wedge \omega^l \ + \ (-1)^k \omega^k \wedge (d\omega^l) \tag{2.16}$$

Which resembles to Leibniz's rule in vector calculus. From vector calculus we know that the following holds, $\mathbf{curl} \times \mathbf{grad} = 0$ and $\mathbf{div} \times \mathbf{curl} = 0$. In differential geometry these rules are combined in,

$$dd\omega^{(k)} = 0^{(k+2)}. \tag{2.17}$$

Meaning that the exterior derivative applied twice to a given k-form results in zero. One very elegant theorem that combines differentiation and integration is Stokes's theorem which is given by,

$$\int_\Omega d\omega = \int_{\partial\Omega} \omega. \tag{2.18}$$

Meaning that the integral over the domain Ω of a derivative of a differential form is equal to the integral over the boundary of domain Ω of the differential form itself. This theorem combines, among others, the divergence theorem of Gauss and Stokes's circulation theorem in one elegant formula. For a k equals $0, 1$ and 2 the stokes theorem reduces to the well known vector relations,

$$\begin{aligned} k=0 \qquad & \int_a^b \mathbf{grad}\phi dx = \phi(b)-\phi(a) && : H_P \longrightarrow H_L \\ k=1 \qquad & \int_S \mathbf{curl} A dS = \int_{\partial S} A ds && : H_L \longrightarrow H_S \\ k=2 \qquad & \int_V \mathbf{div} A dV = \int_{\partial V} A ds && : H_S \longrightarrow H_V. \end{aligned}$$

We can say that the **grad** maps points onto lines, the **curl** maps from lines to surfaces and the **div** from surfaces to volumes, which, when put into a sequence, looks like

$$\mathbb{R} \hookrightarrow H_P \overset{\text{grad}}{\longmapsto} H_L \overset{\text{curl}}{\longmapsto} H_S \overset{\text{div}}{\longmapsto} H_V. \tag{2.19}$$

2.4 The Hodge $\star$ operator

The exterior derivative gives rise to the exact sequence (De Rahm complex),

$$\mathbb{R} \hookrightarrow \Lambda^0(\Omega) \overset{d}{\mapsto} \Lambda^1(\Omega) \overset{d}{\mapsto} \Lambda^2(\Omega) \overset{d}{\mapsto} \Lambda^3(\Omega) \overset{d}{\mapsto} 0, \tag{2.20}$$

which we recognise as (2.19). The De Rahm complex represents the fact that the exterior derivative maps k-forms into $k+1$-forms. It is exact because of (2.17). From (2.19) it becomes clear that we cannot apply the **div** directly to a **grad**, because the divergence operates on surfaces, yet the grad maps onto lines. We can however, copy the exact sequence in reverse and define a map that connects the two exact sequences,

$$\begin{array}{ccccccccccc} \mathbb{R} & \longrightarrow & \Lambda^0(\Omega) & \overset{d}{\longrightarrow} & \Lambda^1(\Omega) & \overset{d}{\longrightarrow} & \Lambda^2(\Omega) & \overset{d}{\longrightarrow} & \Lambda^3(\Omega) & \overset{d}{\longrightarrow} & 0 \\ & & \star\updownarrow & & \star\updownarrow & & \star\updownarrow & & \star\updownarrow & & \\ 0 & \overset{d}{\longleftarrow} & \tilde{\Lambda}^3\Omega) & \overset{d}{\longleftarrow} & \Lambda^2(\Omega) & \overset{d}{\longleftarrow} & \Lambda^1(\Omega) & \overset{d}{\longleftarrow} & \Lambda^0(\Omega) & \longleftarrow & \mathbb{R} \end{array} \tag{2.21}$$

We see then that in $\mathbb{R}^3$ 0-forms map onto 3-forms, 1-forms onto 2-forms, 2-forms onto 1-forms and 3-forms map back onto 1-forms. That map is called the Hodge $\star$ operator. Writing in this sequence in terms of **div**,**grad** and **curl** gives,

$$\begin{array}{ccccccccccc} \mathbb{R} & \longrightarrow & H_P & \overset{\mathbf{grad}}{\longrightarrow} & H_L & \overset{\mathbf{curl}}{\longrightarrow} & H_S & \overset{\mathbf{div}}{\longrightarrow} & H_V & \longrightarrow & 0 \\ & & \star\updownarrow & & \star\updownarrow & & \star\updownarrow & & \star\updownarrow & & \\ 0 & \longleftarrow & H_V & \overset{\mathbf{div}}{\longleftarrow} & H_S & \overset{\mathbf{curl}}{\longleftarrow} & H_L & \overset{\mathbf{grad}}{\longleftarrow} & H_P & \longleftarrow & \mathbb{R} \end{array} \tag{2.22}$$

Important in this work is the application of the **div** operator to a **grad** operator. In order to do this we see that we must first map the form that results from the **grad** onto H_S before

we can apply the **div**. Or

$$\Lambda^1 \xrightarrow{\star} \Lambda^{n-1}, \qquad \text{here } n = 3 \tag{2.23}$$

In terms of differential geometry the Laplace operator operating on a 0-form is then written as,

$$d \star d\phi^{(0)} \tag{2.24}$$

When we apply the Hodge $\star$ to a k-form we essentially map information from one geometrical object to another. When we define an inner product $(.,.)$ for k-forms in terms of their vector proxies, the Hodge $\star$ operator is defined by,

$$(a,b)\omega_n = a \wedge \star b, \quad \text{in } R^3 \;\; (a,b) = a^x b^x + a^y b^y + a^z b^z \tag{2.25}$$

Here ω_n is the normalised n-form which satisfies $\omega_n \wedge \star\omega_n = \omega_n$, see Flanders (1962), the map is linear such that,

$$\star(\alpha dx + \beta dy + \gamma dz) = \alpha \star dx + \beta \star dy + \gamma \star dz. \tag{2.26}$$

The mappings of the Hodge $\star$ operator in $\mathbb{R}^3$ are,

$$\begin{aligned} \star 1 &= dxdydz \longrightarrow \star dxdydz = 1 \\ \star dx &= dydz \quad \star dy = dzdx \quad \star dz = dxdy \end{aligned}$$

In this work an important case of the Hodge $\star$ is when it is applied to the 1-form $u^{(1)}$, then according to (2.27) we get,

$$u^{(1)} = u_x dx + u_y dy + u_z dz \;\; \longrightarrow \;\; \star u^{(1)} = u_x dydz + u_y dzdx + u_z dydz \tag{2.27}$$

This form $\star u^{(1)}$ plays an important role through this thesis, we therefore define $\star u^{(1)} = q^{(n-1)}$. Now if we apply the exterior derivative to $q^{(n-1)}$ we see that this is analogous to taking the divergence of the vector field **u**.

$$\begin{aligned} dq^{(n-1)} &= \frac{\partial u_x}{\partial x} dxdydz + \frac{\partial u_x}{\partial y} dydydz + \frac{\partial u_x}{\partial z} dzdydz + ... \\ &\quad \frac{\partial u_y}{\partial x} dxdzdx + \frac{\partial u_y}{\partial y} dydzdx \frac{\partial u_y}{\partial z} dzdzdx + \dots \\ &\quad \frac{\partial u_z}{\partial x} dxdxdy + \frac{\partial u_z}{\partial y} dydydz + \frac{\partial u_z}{\partial z} dzdxdy \\ &= (\frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{\partial u_z}{\partial z}) dxdydz, \end{aligned} \tag{2.28a}$$

whose vector proxy is given by,

$$\nabla \cdot \mathbf{u} = (\frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{\partial u_z}{\partial z}) \tag{2.28b}$$

2.5 Mappings

One of the key points in using differential geometry, is the use mappings from one domain to another. For differential forms these are very well defined and prove to be of great value when we move to deformed domains. Let us define a continuously differentiable map Φ from an orthogonal domain Ω' to an arbitrary curvilinear domain Ω with coordinates by the operator Φ,

$$\Phi \quad : \quad \Omega' \longrightarrow \Omega. \tag{2.29}$$

The action of this mapping is demonstrated by Figure 2.1. We associate with each point in Ω' a point in Ω, this is the map Φ. This map does not need to be bijective, meaning that a point in Ω does not necessarily have to be associated to a *single* point in Ω'. From this map

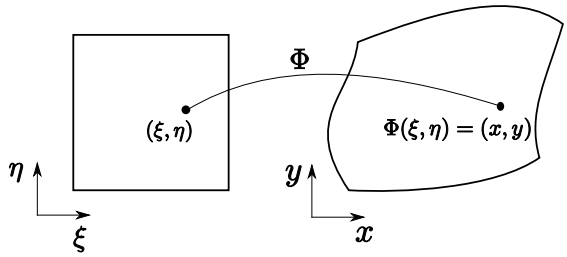


Figure 2.1: A visual representation of the mapping operator

Φ one can construct an *induced* mapping $\Phi^\star$ that maps k-forms from domain Ω to domain Ω',

$$\Phi^\star \quad : \quad \Lambda^k(\Omega) \longrightarrow \Lambda^k(\Omega'). \tag{2.30}$$

The mapping of k-forms from domain Ω to Ω' by $\Phi^\star$ is often called, to pull back. Therefore the operator $\Phi^\star$ is often named the pullback operator. Take note that for each k-form (k=0,1,2) the action of the pullback operator differs, but all of these actions are denoted by one symbol. The reason for introducing the pullback operator is because we need it in our derivation of the discrete set of equations for transformed domains. These properties are given by (2.31a) and 2.31b, the first is that the pullback operator commutes with the wedge product. The second is that is commutes with the exterior derivative.

$$\Phi^\star(a \wedge b) = \Phi^\star(a) \wedge \Phi^\star(b) \tag{2.31a}$$
$$\Phi^\star(da) = d(\Phi^\star a). \tag{2.31b}$$

We also use the change of coordinate formula, which in fact defines the pullback operator.

$$\int_{\Phi(\Omega')} \alpha^{(k)} = \int_{\Omega'} \Phi^\star(\alpha^{(k)}). \tag{2.32}$$

Where Ω' is the k-dimensional domain of integration. For a more thorough discussion on the pullback operator we refer to Flanders (1962).

2.6 Algebraic topology

When setting up a discrete system to approximate the solution of a PDE we need a discrete representation of the differential forms involved in the PDE. Now, in this process a number of choices need to be made. These choices involve the shape of the grid on which the differential forms are discretized. To guide us in these choices we use concepts developed in the field of algebraic topology, of which we present here a brief introduction. For a more detailed description we refer to Tonti (1972) and Mattiussi (2000).

Chains and cells: Let us introduce the concept of a k-cell, in $\mathbb{R}^3$ a k-cell represents a node,an edge,a surface or a volume, respectively these are named 0-cell, 1-cell, 2-cell or a 3-cell. We will denote a k-cell with the symbol σ^k. Now a linear combination of oriented k-cells is called a k-chain which we express as,

$$C_k = \sum_{i=1}^{r_k} m_i \sigma_i^k. \tag{2.33}$$

Where r_k denotes the number of k-cells that make up the k-chain and the multiplicity m_i takes values -1, 1 and 0 where 0 means that the cell is not part of the chain. The orientation of a k-cell can be internal or external, the orientations important throughout this thesis (in $\mathbb{R}^2$) are given in Fig 2.2. Here we have the node, the edge and the surface. The surface we use can be a source or a sink, likewise the node. The internally oriented edge can be associated with gradients. The externally oriented edge with fluxes. There is also the possibility of rotation for the node and the surface but we omit these since they are not used in our derivations that follow.

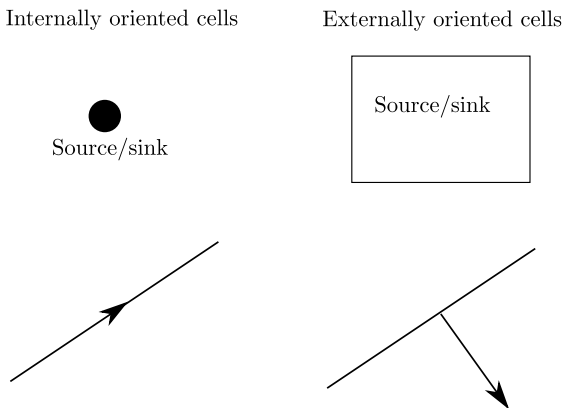


Figure 2.2: Orientations of the node, edge and surface in $\mathbb{R}^2$

Now for $1 < k \leq n$ the boundary of a k-cell, σ^k consists of a (k-1)-chain.

$$\partial C_k = \sum_{i=1}^{r} m_i \partial \sigma_i^k. \tag{2.34}$$

We denote the boundary of a k-cell by $\partial\sigma^k$, where the symbol ∂ represents the boundary operator. As an example, the boundary of an oriented volume (3-cell) is a collection of

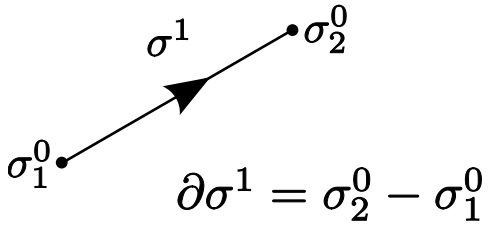


Figure 2.3: Example of the boundary operator operating on a 1-cell

oriented surfaces (2-cells). The boundary of an oriented 2-cell is given by a collection of oriented lines (1-cells) and the boundary of an oriented edge is given by points (0-cells), as shown in Fig. 2.3. One can show that the boundary of the boundary of any cell complex is empty,

$$\partial\partial C_k \equiv 0. \tag{2.35}$$

Or that the boundary of the boundary of a k-chain is always zero. One can think of the k-cells as the nodes, edges, surfaces and volumes (the geometrical objects) that make up a grid that is used in a numerical computation. Important is that in algebraic topology it is only the connectivity that matters. There is no notion of distance or angle. A grid can be course or dense, in algebraic topology this has no meaning. In the introduction of this section we talked about the shape of the grid, it is better to talk of connectivity. Since shape also has the notion of curves, distances and angles. But changing distances and angles in a grid does not influence the connectivity.

Cochains: Now that we have defined the the k-chain we can associate values to these chains. Lets associate to a each k-cell in a chain (σ^k) the integral of the k-form ($\omega^{(k)}$). The association

$$C_k \longrightarrow \left\{ \int_{\sigma_i^k} \omega^k \right\}, \tag{2.36}$$

between a k-form and its representation on a k-chain is called *cochain* and is denoted by $C_k^\star$. Next we define an operator δ, called the coboundary operator. Let C_{k+1} be a (k+1)-chain then its boundary ∂C_{k+1} is a k-chain (C_k) and we can formally write,

$$\langle \partial C_{k+1}, C_k^\star \rangle = \langle C_{k+1}, \delta C_k^\star \rangle. \tag{2.37}$$

where the coboundary operator δ is the formal adjoint of the boundary operator ∂, we can also interpret this as the discrete version of the Stokes theorem (2.18) where δ assumes the role of the exterior derivative d. Note that,

$$\delta \ : \ C_k^\star \longrightarrow C_{k+1}^\star. \tag{2.38}$$

We can also repeat (2.37) and say that,

$$\langle \partial\partial C_{k+2}, C_k^\star \rangle = \langle C_{k+2}, \delta\delta C_k^\star \rangle, \tag{2.39}$$

which combined with (2.35) gives,

$$\delta\delta C_k^\star \equiv 0. \tag{2.40}$$

Remember that one can think of a cell complex as a topological grid on which the cochains are defined that contain the information from k-forms.

Incidence matrices The action of the coboundary operator on cochains can be seen as a matrix operator with the vector containing the cochains (reduced k-forms). These matrices are called incidence matrices, they are purely topological and remain unchanged under deformation of the cell-complex. As long as the topology (connection of points, lines and surfaces) remains the same. Consider for instance the cell-complex as shown in Fig. 2.4.

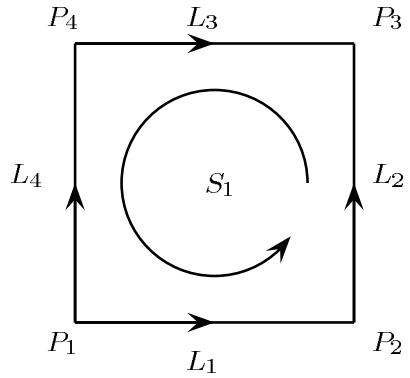


Figure 2.4: Example of the boundary operator operating on a 1-cell

The incidence matrix $\mathbb{E}^{1,0}$ (2.41) relates 0-cochains to 1-cochains, i.e. this matrix is the discrete gradient operator. In this matrix, each row represents a 1-cochain. And such a 1-cochain consists of two 0-cochains, leaving a node results in -1 and arriving at a node in a $+1$. The value 0 indicates that the 0-cochain is not part of the boundary of the 1-cell. The matrix $\mathbb{E}^{2,1}$ relates 1-cochains and 2-cochains (2.41), it can be see as the discrete curl operator. Here a -1 corresponds to a line with an orientation in the opposite direction as the positive rotation, and a $+1$ to a line with an orientation in the same direction as the positive rotation.

$$\mathbb{E}^{1,0} = \begin{pmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 \\ -1 & 1 & 0 & 0 \end{pmatrix}, \qquad \mathbb{E}^{2,1} = \begin{pmatrix} 1 & 1 & -1 & -1 \end{pmatrix} \tag{2.41}$$

One can readily check that

$$\mathbb{E}^{2,1} \cdot \mathbb{E}^{1,0} = \begin{pmatrix} 1 & 1 & -1 & -1 \end{pmatrix} \begin{pmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 \\ -1 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}, \tag{2.42}$$

which is the matrix expression for $\delta\delta C_0^\star = 0$.

Dual cell complex In differential geometry the exterior derivative gives rise to the exact sequence (2.20). The Hodge $\star$ operator allows one to construct a map from one discrete sequence to another, thus enabling to describe an equation like the Poisson equation using differential geometry. With algebraic topology being the discrete counterpart of differential geometry the we also need a mapping from a k-cochain to a $(n-k)$-cohain, analogous to the coupling of k-forms and $(n-k)$-forms. This can be accommodated by introducing a dual cell complex. The dual cell complex is constructed such that every p-cell on the primal complex corresponds to a $(n-p)$-cell on the dual complex, see Fig 2.5. The orientation of the

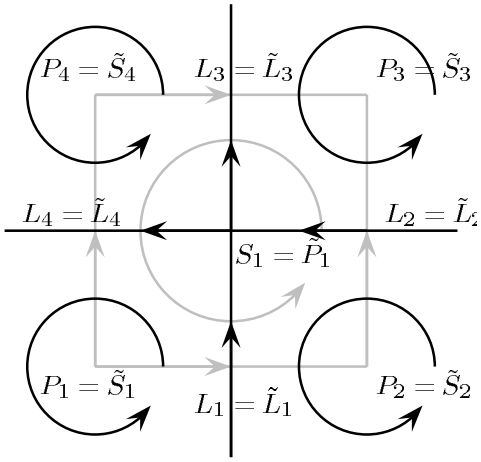


Figure 2.5: Example of the boundary operator operating on a 1-cell

dual p-cells is determined by the orientation of the n-dimensional space in which the complex is embedded. Here we can construct for instance the incidence matrix $\tilde{\mathbb{E}}^{1,0}$, which relates 0-cochains on the dual grid to 1-cochains on the dual grid.

$$\tilde{\mathbb{E}}^{1,0} = \begin{pmatrix} 1 \\ 1 \\ -1 \\ -1 \end{pmatrix} = \mathbb{E}^{2,1} \tag{2.43}$$

This matrix corresponds to the *divergence* operator on the primal grid. Similarly we have that $\tilde{\mathbb{E}}^{2,1}$, which relates 1-cochains on the dual grid to 2-cochains on the dual grid, is equal to $\mathbb{E}^{1,0}$.

$$\tilde{\mathbb{E}}^{2,1} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix} = \mathbb{E}^{1,0}. \tag{2.44}$$

The mapping that maps information from a k-cell complex to its *dual* $(n-k)$-cell complex can be seen as the discrete equivalent of the Hodge $\star$ operator. It can be constructed in a multitude of ways and depends strongly on how we reconstruct the continuous differential forms from the discrete cochains. It is also in this discrete Hodge $\star$ operator where the errors are introduced in the discretization, the coboundary operator itself is exact, yet when trying to couple information from the two dual cell complexes an error is introduced.

2.7 Differential geometry and algebraic topology connected

When we connect values of differential forms to k-cells we are in fact reducing the continuous k-forms to discrete values using integration. Lets define this integration or reduction of a k-form ω onto a k-chain c as,

$$\langle \mathcal{R}, c \rangle = \int_c \omega. \tag{2.45}$$

In this case c is a metric chain, not solely a topological chain. The mapping $\omega \rightarrow \mathcal{R}\omega$ establishes discrete representation of k-forms in terms of global quantities associated with a chain complex. Thus, we encode discrete k-forms as k-cell quantities, k-cochains. This mapping has a property, which is known as the commuting diagram property, that is given by,

$$\mathcal{R}d = \delta\mathcal{R}. \tag{2.46}$$

When solving a PDE numerically we need to reduce the continuous k-forms to a discrete representation on a cell complex. However we might need to be able to reconstruct from this discrete representation a continuous field again. We denote this reconstruction by the symbol $\mathcal{I}$ for interpolation, here $\mathcal{I}$ is an approximate left-inverse of $\mathcal{R}$. Where reduction is a fairly straightforward integration of k-forms, interpolation can be done in a multitude ways and depends highly on the shape of the cell complex on which the differential forms are discretized. This thesis however discusses the use of spectral elements for solving the Poisson equation, this means that our interpolators are combinations of Lagrangian interpolators based on the Legendre polynomials. For mimetic reconstruction operators $\mathcal{I}$ we impose a condition that serves to coordinate the action of the exterior derivative and the coboundary operator, this condition is known as the second commuting diagram property,

$$d\mathcal{I} = \mathcal{I}\delta \tag{2.47}$$

Such a map is called a *conforming reconstruction operator*.

Chapter 3

Basis Functions

In the introduction we mention the word Spectral Element Method, originally stemming from the use of Fourrier series methods, the word Spectral now also includes the use of polynomial basis functions, which is the route we take. Spectral methods are known for their exponential convergence, being the fastest convergence possible. In this Chapter we discuss 2 types of basis functions using examples of how to interpret and use them. The first of these types of basis functions is the nodal basis function, this type is well known and is extensively treated by Sherwin and Karniadakis (1999). The second type of basis function we discuss is called the edge function, it has only recently been introduced in a paper by Gerritsma (2009) and is the key to constructing a suitable interpolator (2.47) that is used in the discretization that we propose.

3.1 Nodal basisfunctions

One of the two types of basis functions that we use is the nodal basis functions. These basis functions are a set of normalized unique Lagrangian (App. A) polynomials with which one expands a function. As a basis for these Lagrangian polynomials, often some form of the Legendre polynomial (App A.1) is used. One such basis function in particular is called the Legendre-Gauss-Lobatto (GL) basis function. These basis functions ($h_i(\xi)$) are associated to the GL-nodes (ξ_i) for which hold that,

$$(1-\xi_i)^2 L'_N(\xi_i) = 0, \qquad \text{for} \quad -1 \leq \xi_i \leq 1 \tag{3.1}$$

where L'_N is the derivative of the N^{th} order Legendre polynomial. The basis functions which are constructed using,

$$h_i(\xi) = \frac{\left(1-\xi^2\right) L'_N(\xi)}{N(N+1)L_N(\xi_i)(\xi_i - \xi)}\,, \quad i = 0, \ldots, N\,. \tag{3.2}$$

are depicted in Fig. 3.1 for $N = 3$. Note that in this case there are four GL nodes $(\xi_1 \dots \xi_4)$. The nodal basis functions' orthogonality property is expressed by,

$$h_i(\xi_k) = \begin{cases} 1 & \text{if } i = k \\ \\ 0 & \text{if } i \neq k \end{cases} \tag{3.3}$$

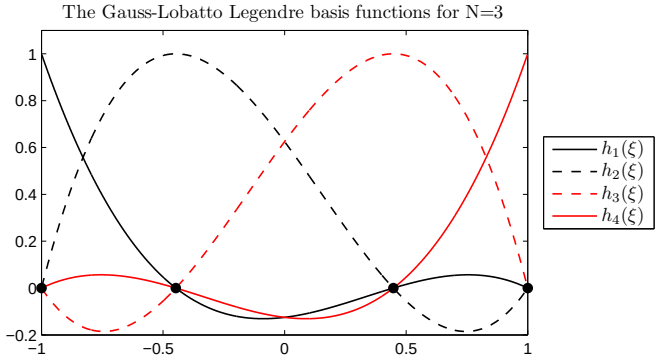


Figure 3.1: The shape of the GL Lagrangians for $N = 3$

We can use these basis functions to approximate a function $f(\xi)$ on the interval $-1 \leq \xi \leq 1$ by using,

$$f(\xi) \approx f^N(\xi) = \sum_{i=1}^{N+1} f(\xi_i) h_i(\xi). \tag{3.4}$$

Naturally this expansion gives the best results when the function $f(\xi)$ is a polynomial itself. We can however use this approximation for non polynomial but *smooth* functions. When the function is not smooth the exponential convergence, that we mentioned in the chapter's introduction is lost. Now say we want to approximate the function $f(\xi) = sin(\pi\xi)$ using the GL basis functions. The approximation in that case is given by,

$$\sin(\pi\xi) \approx \sum_{i=1}^{N+1} \sin(\pi\xi_i) h_i(\xi). \tag{3.5}$$

Fig 3.2 shows for the case of $N = 3, 4, 6$ and 8 the resulting approximation $f^N(\xi)$ and the exact value $f(\xi)$. The difference between $f(\xi)$ and $f^N(\xi)$ goes down quite fast for increasing N..

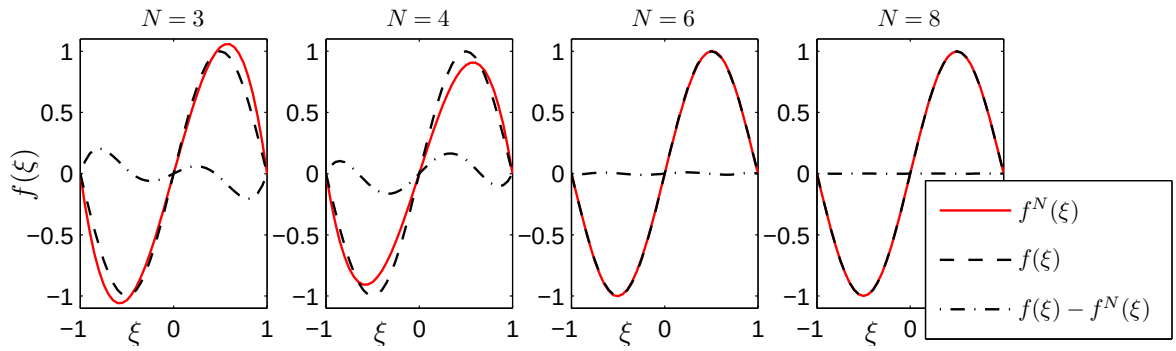


Figure 3.2: The shape of the GL Lagrangians for $N = 3$

3.2 Edge functions

In Chapter 2 we discussed the notion that the discrete representation of a differential (k) -form should be connected to a similar type of k-cell. So a 0-form is reduced on a 0-cell (node). When we want to reconstruct the continuous field we use nodal basis functions like the Legendre-Gauss-Lobatto basis functions discussed in the previous section. Now similarly a 1-form should be connected to a 1-cell, this is done by integration along a line segment, giving us the 1-cochain. This means that we also need to be able to reconstruct from this integral line (edge) value the continuous field. The basis functions with which we do this, we call edge functions. The edge functions, which we denote by the symbol $e_i(\xi)$, according to Gerritsma (2009), are defined by:

$$e_i(\xi) = -\sum_{k=0}^{i-1} dh_k(\xi) \; . \tag{3.6}$$

Here the basis functions h_k can be a set of nodal basis functions like the Legendre-Gauss-Lobatto basis functions, (3.2). The edge functions are constructed such that,

$$\int_{\xi_{k-1}}^{\xi_k} e_i(\xi) = \begin{cases} 1 & \text{if } \; i = k \\ \\ 0 & \text{if } \; i \neq k \end{cases} \tag{3.7}$$

For $N = 6$ we show e_3 in Fig. 3.3, here we see that the edge function integrated between two GL points other than ξ_3 and ξ_4 gives the value 0. If it is integrated between ξ_3 and ξ_4 it gives the value 1. We can use these basis functions to approximate a function $f(\xi)$ on the interval $-1 \leq \xi \leq 1$ by using,

$$f(\xi) \approx f^N(\xi) = \sum_{i=1}^{N+1} \left(\int_{\xi_i}^{\xi_{i+1}} f(\xi_i) d\xi \right) \; e_i(\xi). \tag{3.8}$$

We show in Fig 3.4 the interpolation of $f(\xi) = \sin(\xi\pi)$. For a thorough discussion and derivation of the edge function we refer to Gerritsma (2009). We cannot stress enough how important these basis functions are in creating a compatible discretization of the Poisson equation.

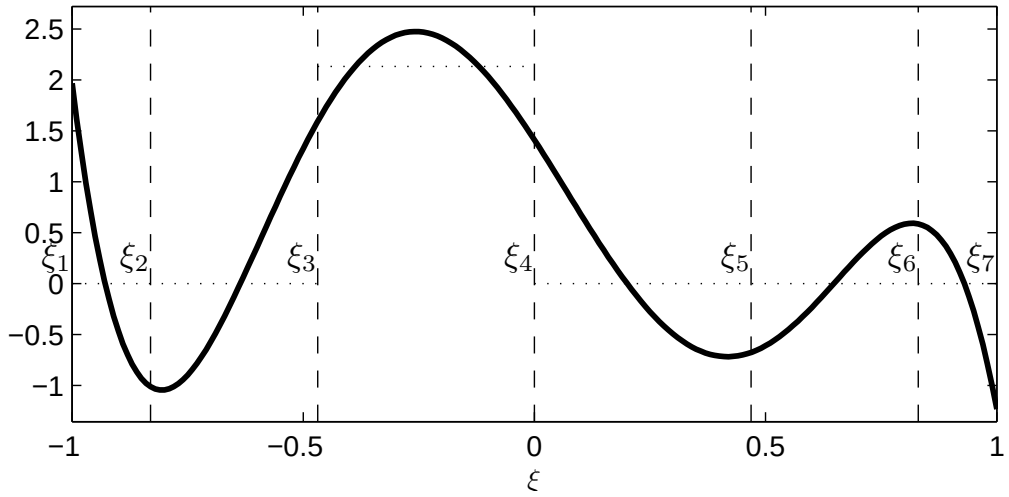


Figure 3.3: The edge function e_3 for $N = 6$

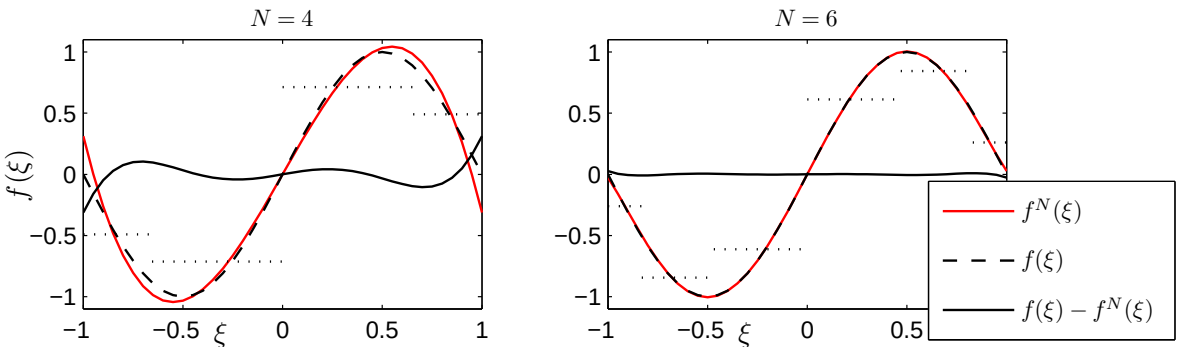


Figure 3.4: Interpolation using the edge functions for $N = 4$ and 6

Chapter 4

Poisson equation on orthogonal domains

In this chapter the idea behind the support operator method is presented. We start by stating the form of the Poisson equation in differential geometry, after which we can derive the general connection between the **div** and **grad** operator. Then we discuss the discretization and reconstruction of the k-forms involved in the Poisson equation, with these discretizations we automatically define the discrete differential operators with which we setup the final system of equations.

4.1 Poisson equation

In Chapter 2 basic concepts of differential geometry like differential forms, the exterior derivative and the wedge product have been discussed. Using these concepts we can reformulate the Poisson equation ((2.2a),(2.2b) and (2.2c)). In terms of differential forms this system is given by

$$d\phi^{(0)} = u^{(1)} \tag{4.1a}$$
$$dq^{(n-1)} = f^{(n)} \tag{4.1b}$$
$$q^{(n-1)} = \star u^{(1)}. \tag{4.1c}$$

Here $u^{(1)}$ denotes a 1-form, $\phi^{(0)}$ is a 0-form and $f^{(n)}$ is a prescribed n-form. The Hodge $\star$-operator associates the 1-form $u^{(1)}$ with the corresponding $(n-1)$-form $q^{(n-1)}$. Using this representation of the Poisson equation in differential geometry we create a compatible discretization. As stated briefly in the introduction, the compatibility of the discretization originates from the way we define the discrete operators with which we replace the system of first order equations. For ease of notation we will use the terms **div** and **grad** when referring to the exterior derivative operating on a $(n-1)$-form or a 0-form respectively. Using differential geometry we show that the **div** operator and **grad** are related and are each others

formal adjoint. We can use this relation to define implicitly a discrete **grad** operator by using a discrete **div** operator as a support, thus preserving as much of the structure of the PDE as possible. This way of relating the two operators is often called, the support operator method. Since one discrete operator serves as a support for the other. In a finite difference setting this was done by Hyman (1997) and Bochev and Hyman (2006).

4.2 Support operator

The basis of the support operator method lies with the definition of the hodge $\star$-operator. So let's start from the definition of the Hodge (2.25) and integrate over the domain of interest, which we call Ω.

$$\begin{aligned}
\int_\Omega (\star d\phi^{(0)}, q^{(n-1)})\omega_n &= \int_\Omega d\phi^{(0)} \wedge q^{(n-1)} && \text{(4.2a)}\\
&= \int_\Omega d(\phi^{(0)} \wedge q^{(n-1)}) - \int_\Omega \phi^{(0)} \wedge dq^{(n-1)} && \text{(4.2b)}\\
&= \int_{\partial\Omega} \phi^{(0)} \wedge q^{(n-1)} - \int_\Omega \phi^{(0)} \wedge dq^{(n-1)}. && \text{(4.2c)}
\end{aligned}$$

The right hand side of (4.2a) can be rewritten by making use of the general product rule for differentiation (2.16) to split this term in two parts. After which we make use of the generalized Stokes theorem that combines the Newton-Leibniz Theorem, Stokes Circulation Theorem, and Gauss Divergence Theorem, to give rise to the integral over the boundary of Ω indicated with $\partial\Omega$. This integral shows the connection between the **grad**ϕ which is the term $\star d\phi^{(0)}$ and the **div**q which is *partially* the term $dq^{(n-1)}$. In this case, where we deal with an *undeformed orthogonal* domain, the equation can also be derived using vector calculus with its differentiation and integration rules, resulting in,

$$\int_\Omega (\nabla\phi, u) = \int_{\partial\Omega} \phi(\mathbf{u}\cdot\mathbf{n}) - \int_\Omega \phi(\nabla\cdot\mathbf{u}). \tag{4.3}$$

Now say we have homogeneous Dirichlet or Neumann boundary conditions, then (4.2c) reduces to,

$$\begin{aligned}
\int_\Omega (\star d\phi^{(0)}, q^{(n-1)})\omega_n &= -\int_\Omega \phi^{(0)} \wedge dq^{(n-1)} && \text{(4.4a)}\\
\int_\Omega (\text{grad}\phi, \mathbf{q})\omega_n &= -\int_\Omega (\phi \text{div}\mathbf{q})\omega_n. && \text{(4.4b)}
\end{aligned}$$

Let us now define the following two inner products in the scalar space S and in the vector space V,

$$(a,b)_S = \int_\Omega a^{(0)} \wedge b^{(n)}, \qquad (A,B)_V = \int_\Omega (A^{(1)}, B^{(1)})\, \omega_n, \tag{4.5}$$

then the identity (4.4b) can be written as,

$$(\text{grad}\phi, \mathbf{q})_V = -(\phi, \text{div}\mathbf{q})_S. \tag{4.6}$$

This expression states that the gradient and the divergence operators are each others negative adjoint.

$$\text{grad} = -\text{div}^T. \tag{4.7}$$

This relationship between the divergence and the gradient operator is the basis of our approach in finding a mimetic discretization for the Poisson equation. In this example we show the connection between **div** and **grad**, and assume that either $\mathbf{q}^{(1)}$ or $\phi^{(0)}$ is zero at the boundary, this might not always be the case however. But by defining an operator $\mathcal{D}$ operating on $\mathbf{q}^{(1)}$ as (4.8) then the gradient operator $\mathcal{G}$ is the negative adjoint of the divergence operator.

$$\mathcal{D}\mathbf{q} = \begin{cases} \text{div}q \quad for, & \in \Omega \\ -\mathbf{q}\cdot\mathbf{n} \quad for, & \in \partial\Omega \end{cases} \tag{4.8}$$

Actually the term $\mathbf{q}\cdot\mathbf{n}$ is incorrect, but here we use this formulation to indicate that the divergence at the boundary is the in- *or* out-flux. We can see why the formulation including the outward normal is incorrect, if we look at the definition of q^{n-1} in $\mathbb{R}^2$,

$$q^{n-1} = -u_\eta d\xi + u_\xi d\eta. \tag{4.9}$$

It is quite clear that, without focusing on the sign, u_η and u_ξ are normal to $d\xi$ and $d\eta$ respectively. This means that the integration of q^{n-1} along a path, is integration of the components of $u^{(1)}$ that are normal to that path. Adding $\mathbf{n}$ is therefore incorrect but is meant to be clarifying.

4.3 Definition of the spectral element dual cell-complexes

As stated in the introduction we propose a mimetic spectral element method. This means we approximate differential forms using a set of polynomial basis-functions. Spectral methods are known for their exponential convergence rates, it is mainly because of this high accuracy that we investigate the use of a mimetic spectral element method. First we need to define a grid (cell-complex) on which we represent (reduce) our differential forms. We choose to use two staggered grids (dual cell-complexes) as depicted in Fig. 4.1. The black (dual) cell-complex is formed by connecting the Legendre-Gauss-Lobatto (GL) nodes. The red (primal) cell-complex is formed by taking the Gauss-Legendre (G) nodes for the interior and supplement these with nodes at the boundary of the domain, which we call the Extended Gauss (EG) cell complex. For a thorough discussion on spectral element meshes and basis functions we refer to Sherwin and Karniadakis (1999). There are two routes in explaining the use of this staggered configuration. The first is through algebraic topology. From differential geometry we know that $u^{(1)}$ and $q^{(n-1)}$ are connected by the Hodge dual ($\star$). And that because of this connection through the Hodge their discrete representation is on dual cell-complexes. These two staggered grids represent this duality. Another motivation to create the staggered

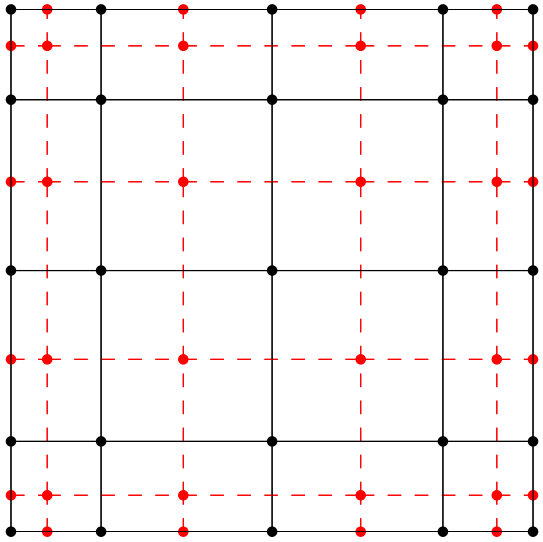


Figure 4.1: Dual spectral element grids, black squares the GLL points, red squares the EG points

configuration can be found by looking at the integral relation (4.2c), which forms the basis of our mimetic discretization. The right hand side of this equation contains an integral over the complete domain Ω,

$$\int_{\Omega} \phi^{(0)} \wedge dq^{(1)}. \tag{4.10}$$

A natural position for these integration points are in the Gauss points, these lie nearly in the centre of the sub-domains enclosed by the Gauss-Lobatto grid. The Gauss points (ξ_i) are given by,

$$\text{Gauss points: } L_N(\tilde{\xi}_i) = 0\ , \quad i = 1, \ldots, N. \tag{4.11}$$

Where L_N is the N^{th} order Legendre polynomial, see App A.1. The second integral from the r.h.s. in (4.2c) is the boundary integral,

$$\int_{\partial\Omega} \phi^{(0)} \wedge q^{(1)}. \tag{4.12}$$

This integral suggests that we also have ϕ nodes on the boundary of the domain. Together the points that make up the primary cell-complex are the Gauss points $(\tilde{\xi}_i, \tilde{\eta}_j)$ supplemented with the boundary nodes

$$(\tilde{\xi}_0, \tilde{\eta}_j) = (-1, \tilde{\eta}_j)\ , \quad (\tilde{\xi}_{N+1}, \tilde{\eta}_j) = (1, \tilde{\eta}_j)\ , \quad j = 1, \ldots, N\ ,$$

$$(\tilde{\xi}_i, \tilde{\eta}_0) = (\tilde{\xi}_i, -1)\ , \quad (\tilde{\xi}_i, \tilde{\eta}_{N+1}) = (\tilde{\xi}_i, 1)\ , \quad i = 1, \ldots, N\ .$$

Note that referring to an EG-point is done with the symbol tilde $(\tilde{\xi}_i)$. The cell-complex dual to this primary cell-complex is the black grid in Fig. 4.1. Its nodes (ξ_i) are the zeros of $(1-\xi^2)L'_N$ where L'_N is the derivative of the Legendre polynomial of degree N for $i = 0, \ldots, N$. The choice of this dual cell-complex is twofold, first is that it is natural that the flux, which is tightly connected to inflow and outflow of some quantity, is discretized along the boundary of a domain which is included by the GL points. Second is that the Legendre-Gauss-Lobatto (GL) points are well documented, which is mainly an advantage in their use.

4.4 The conservation law $dq^{(1)} = f^{(2)}$

In order to go from a discrete representation (co-chains) back to a continuous representation (differential forms) we need to choose an interpolation. In this section we show, that by choosing a suitable interpolator for $\mathcal{R}q^{(n-1)}$,we get an exact representation of the conservation law. The (n-1)-form $q^{(n-1)}$ is reduced to discrete values on the GLL-grid which is the black grid in Fig. 4.1. Now from algebraic topology we know that reducing a k-form is done by integration. In $\mathbb{R}^2$ the object to which $q^{(n-1)}$ is connected is a (n-1)-cell, or 1-cell. So let us associate with each line segment (1-cell) that connects two GLL-points the integral of $q^{(n-1)}$ over that line segment (4.13).

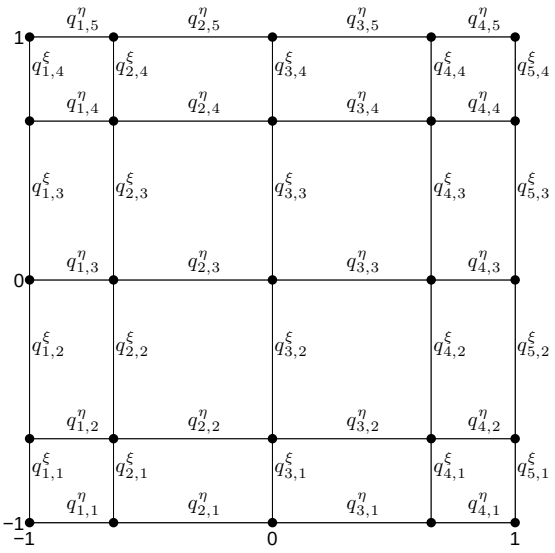


Figure 4.2: Locations of discrete values of $\mathbf{q}$

$$q^{\xi}_{i,j} = \int_{\eta_{j-1}}^{\eta_j} q^{(n-1)} \quad \text{and} \quad q^{\eta}_{i,j} = \int_{\xi_{i-1}}^{\xi_i} q^{(n-1)} \ . \tag{4.13}$$

Here (ξ, η) are the coordinates so,

$$u^1 = u_\xi d\xi + u_\eta d\eta \quad \text{and} \quad q^{(n-1)} = -u_\eta d\xi + u_\xi d\eta. \tag{4.14}$$

The minus sign comes from the definition of the Hodge (2.25) and the anti symmetry of the wedge product (2.9).

A suitable interpolator for $\mathcal{R}q^{(n-1)}$ is,

$$\mathcal{I}\mathcal{R}q^{(n-1)}(\xi,\eta) = -\sum_{i=1}^{N}\sum_{j=0}^{N} q^{\eta}_{i,j} e_i(\xi) h_j(\eta)\, d\xi + \sum_{i=0}^{N}\sum_{j=1}^{N} q^{\xi}_{i,j} h_i(\xi) e_j(\eta)\, d\eta. \tag{4.15}$$

Here the basis functions indicated with an h are defined by (3.2). These are the one of the most commonly used nodal basis functions and are called Legendre-Gauss-Lobatto (GL) basis functions, more about these can be found in App. A.2 and Chapter 3.1. The other type of basis function we use is indicated with an e and is defined by (3.6). These basis functions are edge interpolants, and are used in conjunction with the integral edge values to reconstruct the field values. These edge interpolants (or histopolants) are essential in creating a suitable interpolator for the 1-cochain and 2-cochain. The basis functions e_i are created such that they have the property,

$$\int_{\xi_{k-1}}^{\xi_k} e_i(\xi) = \begin{cases} 1 & \text{if } i = k \\ 0 & \text{if } i \neq k \end{cases} \tag{4.16}$$

So, integrated over the line-segment they belong to the edge functions return unity, if integrated over any other line segment connecting two GL points they return zero. This amounts to $\mathcal{R}\mathcal{I} = Id$, so this interpolation is the right inverse of the reduction operator $\mathcal{R}$. In other words, integration of the interpolation is exact. The interpolation $\mathcal{I}$ however, is only an approximate left inverse of $\mathcal{R}$.

$$\mathcal{I}\mathcal{R} = Id + O(h^s). \tag{4.17}$$

Here the error depends on the smoothness of the approximated function and on the polynomial degree that is used to interpolate it. So (4.15) is a finite dimensional approximation to the true solution $q^{(n-1)}$. In other words, even though the integral of the interpolation is exact the interpolation to a continuous field is still an approximation.

Now let us look at the properties of the interpolation under differentiation. We can take the exterior derivative of (4.15) and treat the interpolation as an ordinary (n-1)-form.

$$\begin{aligned} d\mathcal{I}\mathcal{R}q^{(n-1)}(\xi,\eta) &= -\sum_{i=1}^{N}\sum_{j=0}^{N} q^{\eta}_{i,j} e_i(\xi)\frac{dh_j(\eta)}{d\eta}\, d\xi + \sum_{i=0}^{N}\sum_{j=1}^{N} q^{\xi}_{i,j}\frac{dh_i(\xi)}{d\xi} e_j(\eta)\, d\eta && \text{(4.18a)} \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N}\left(q^{\xi}_{i,j} + q^{\eta}_{i,j} - q^{\xi}_{i-1,j} - q^{\eta}_{i,j-1}\right) e_i(\xi)e_j(\eta) d\xi d\eta && \text{(4.18b)} \\ &= \mathcal{I}\delta\mathcal{R}q^{(n-1)}\,, && \text{(4.18c)} \end{aligned}$$

In order to get from (4.18a) to (4.18b) we have to use a recombination of the basis functions, how this is done is explained briefly in App. A.5 and more so in Gerritsma (2009). (4.18c) shows that taking the exterior derivative of this (suitable) interpolation is equal to the action of the coboundary operator acting on the cochains of $q^{(n-1)}$. This shows that the interpolation operator satisfies the commuting property $d\mathcal{I} = \mathcal{I}\delta$, called the second commuting diagram property, *CDP2* by Bochev and Hyman (2006). To see how well this approximation of the divergence is, we use the generalized Stokes theorem from which we know that the divergence integrated over a certain area is equal to the sum of the fluxes over the boundary of that area. In other words this equation says that the purely topological form of the divergence can be

used in combination with an interpolator that reconstructs from this metric- and error -free topological divergence in points.

We define $\mathcal{R}f^{(2)}$ as the integral of $f^{(2)}$ over the volumes $[\xi_{i-1}, \xi_i] \times [\eta_{j-1}, \eta_j]$,

$$f_{i,j} = \mathcal{R}f^{(2)} := \int_{\xi_{i-1}}^{\xi_i} \int_{\eta_{j-1}}^{\eta_j} f^{(2)} \ , \quad i,j = 1 \ldots, N \ . \tag{4.19}$$

The interpolation of $f_{i,j}$ is defined as

$$\mathcal{I}f_{i,j} = \sum_{i=1}^{N} \sum_{j=1}^{N} f_{i,j} e_i(\xi) e_j(\eta) \ . \tag{4.20}$$

Using (4.18b) and (4.20) we can write $dq^{(n-1)} = f^{(2)}$ as

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \left(q^{\xi}_{i,j} + q^{\eta}_{i,j} - q^{\xi}_{i-1,j} - q^{\eta}_{i,j-1} - f_{i,j} \right) e_i(\xi) e_j(\eta) = 0 \ . \tag{4.21}$$

And since the basis-functions are linearly independent we have that,

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \left(q^{\xi}_{i,j} + q^{\eta}_{i,j} - q^{\xi}_{i-1,j} - q^{\eta}_{i,j-1} - f_{i,j} \right) = 0 \ . \tag{4.22}$$

This equation can be interpreted as a spectral interpolation of a finite volume method. In the sense that this equation merely states that the divergence or the flux over a cells boundary is equal to the value $f_{i,j}$ in that cell. This is exact. The interpolation function is only used when a value at a specific point in the domain is needed. Using concepts from algebraic topology we can write the discrete divergence in terms of the incidence matrix $E^{2,1}$, which is a matrix representation of the coboundary operator (2.38), acting on 1-cochains $q_{i,j} = (\mathcal{R}q^{(n-1)})$ in the dual cell complex (GL grid)

$$\left[E^{2,1} q_{i,j} - f^{(2)}_{i,j} \right] = 0 \ . \tag{4.23}$$

When this equation is satisfied the divergence is satisfied exactly.

4.5 Discretization of $\phi^{(0)}$

For reasons explained earlier, the mimetic discretization of the Poisson equation involves using two different cell-complexes on which to reduce the k-forms. We saw that the flux $(q^{(n-1)})$, the divergence $dq^{(n-1)}$ and $f^{(2)}$ are discretizated on the primary cell-complex. Consequently the 0-form $\phi^{(0)}$ and the 1-form $u^{(1)}$ are discretized on the dual cell-complex (EG-mesh) which is the red mesh in Fig. 4.1. This mesh is depicted in Fig 4.3 in more detail. The reduction operator for the 0-form $\phi^{(0)}$ is defined by sampling the scalar function $\phi^{(0)}$ in the EG-points, i.e.

$$\tilde{\phi}_{i,j} = \mathcal{R}\phi^{(0)} := \phi^{(0)}(\tilde{\xi}_i, \tilde{\eta}_j) \ . \tag{4.24}$$

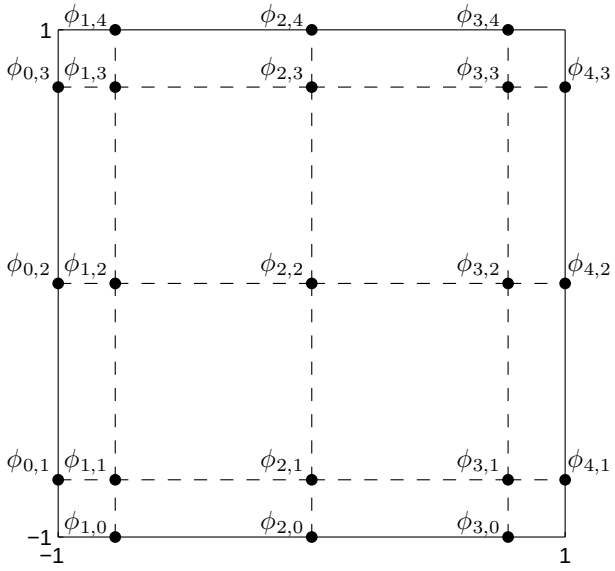


Figure 4.3: The cell complex on which ϕ is discretized ($N = 3$)

Due to the choice the node locations we must separate the interpolation $\phi_{i,j}$ in a part that is used for the interior of Ω.

$$\mathcal{IR}\phi^{(0)} = \sum_{i=1}^{N}\sum_{j=1}^{N} \phi_{i,j}\tilde{h}_i(\xi)\tilde{h}_j(\eta) \quad \text{, for} \qquad \xi, \eta \ \in \Omega, \tag{4.25}$$

and a part that reconstructs the value of ϕ along the boundary,

$$\mathcal{IR}\phi^{(0)} \begin{cases} \phi(-1,\eta) = \sum_{j=1}^{N} \phi_{0,j}\tilde{h}_j(\eta) \\ \phi(1,\eta) \ \ = \sum_{j=1}^{N} \phi_{N+1,j}\tilde{h}_j(\eta) \\ \phi(\xi,-1) = \sum_{i=1}^{N} \phi_{i,0}\tilde{h}_i(\xi) \\ \phi(\xi,1) \ \ = \sum_{i=1}^{N} \phi_{i,N+1}\tilde{h}_i(\xi) \end{cases} \quad \text{for} \quad \xi, \eta \in \partial\Omega. \tag{4.26}$$

where

$$\tilde{h}_i(\xi) = \frac{L_N(\xi)}{L'_N(\xi_i)(\xi - \tilde{\xi}_i)} \quad \text{for} \ \ i = 1 \ldots N. \tag{4.27}$$

Here the basis functions $\tilde{h}$ are the Legendre-Gauss Lagrangian basis functions. A thorough discussion on the choice of interpolator for ϕ is given in App. B. Besides $\phi^{(0)}$ we also discretize $u^{(1)}$ on the dual cell-complex though it is not explicitly used in the discretization of the PDE. The coboundary $\mathcal{R}u^{(1)}$ is the integrated value *along* the line segments connecting the ϕ nodes. By definition the edge value $u_{i,j}$ is the difference between the ϕ's of its endpoints. The reconstruction is given by,

$$\mathcal{IR}u^{(1)} = \sum_{j=1}^{N}\sum_{i=1}^{N+1} u^{\xi}_{i,j}\overline{e}_i(\xi)\tilde{h}_j(\eta)\, d\xi \ + \ \sum_{i=1}^{N}\sum_{j=1}^{N+1} u^{\eta}_{i,j}\overline{e}_j(\eta)\tilde{h}_i(\xi)\, d\eta. \tag{4.28}$$

The basis functions $\overline{e}$ are the Extended Gauss edge interpolants which are given by.

$$\overline{e}_i(\xi) = -\sum_{k=0}^{i-1} d\overline{h}_k(\xi) \; . \tag{4.29}$$

The functions $\overline{h}_k$ are the Extended Gauss polynomials given by A.17. Note how the distinction between the different type of grids is made. Polynomials on the Extended Gauss grid are referred to with $\overline{h}$, on the Gauss grid with $\tilde{h}$ and on the Gauss-Lobatto grid just with the symbol h.

4.6 Formulation of the discrete grad operator

Now following the work of Hyman (1997) we do not proceed by defining the gradient operator explicitly. But instead use (4.2c) to find a discrete gradient operator that is the natural adjoint of the discrete divergence operator. Here we repeat (4.2c).

$$\int_\Omega (\star d\phi^{(0)}, q^{(n-1)})\omega_n = \int_{\partial\Omega} \phi^{(0)} \wedge q^{(n-1)} - \int_\Omega \phi^{(0)} \wedge dq^{(n-1)}.$$

Let $\mathcal{G}\phi = \mathcal{R} \star d\phi^{(0)}$, then because $\star d\phi^{(0)} = q^{(1)}$, it is natural that it is discretized on the primary (GL) cell complex and that the reconstruction of $\mathcal{G}\phi$ is given by,

$$\mathcal{I}\mathcal{G}\phi = -\sum_{k=1}^{N}\sum_{l=0}^{N} \left(\mathcal{G}^\eta\phi\right)_{k,l} e_k(\xi)h_l(\eta)\, d\xi + \sum_{k=0}^{N}\sum_{l=1}^{N} \left(\mathcal{G}^\xi\phi\right)_{k,l} h_k(\xi)e_l(\eta)\, d\eta \; . \tag{4.30}$$

Having the reconstruction of $\mathcal{G}\phi$ and $\mathbf{q}$ we can evaluate the left hand side of (4.2c). For the integration we make use of Gaussian quadrature, which is explained in Appendix A. Using GL quadrature means we have to reconstruct both $\mathbf{q}$ and $\mathcal{G}\phi$ at the GL locations, calculate the inner product in that place and multiply this product by the corresponding GL weight.

$$\begin{aligned}\int_\Omega \left(\star d\phi^{(1)}, \mathbf{q}^{(n-1)}\right) = (\mathcal{I}\mathcal{G}\phi, \mathcal{I}\mathbf{q})_V = \sum_{p=0}^{N}\sum_{q=0}^{N} \Bigg\{ \sum_{j=1}^{N} q^\xi_{p,j} e_j(\tilde{\eta}_q) \sum_{l=1}^{N} \left(\mathcal{G}^\xi\phi\right)_{p,l} e_l(\tilde{\eta}_q) + \ldots \\ \sum_{i=1}^{N} q^\eta_{i,q} e_i(\tilde{\xi}_p) \sum_{k=1}^{N} \left(\mathcal{G}^\eta\phi\right)_{k,q} e_k(\tilde{\xi}_p) \Bigg\} w_p^{GL} w_q^{GL}.\end{aligned} \tag{4.31}$$

So the equation above represents the numerical integration of the inner product of the gradient of ϕ on the primal grid and the flux $\mathbf{q}$ on the primal grid. A closer look at the integral also shows that the basis functions h_i cancel. This is why we choose the GL points as integration points, the final resulting system of equations is a much sparser matrix than when using other integration points. The summation can also be rewritten in a matrix vector notation. The structure that this way of writing brings makes a final description of the system of algebraic

equations quite easy to construct. Let $[a, b]$ denote the inner product between two vectors of equal dimension ($a^T b$) we then write,

$$(\mathcal{I}\mathcal{G}\phi, \mathcal{I}\mathbf{q})_V = [\mathcal{I}^{GL}\tilde{\mathbf{q}}, W^{GL}\mathcal{I}^{GL}\mathcal{G}\phi] \ , \quad \forall \tilde{\mathbf{q}} = \begin{pmatrix} q^\xi \\ q^\eta \end{pmatrix}. \tag{4.32}$$

Here $\mathcal{I}^{GL}$ is the matrix that reconstructs from the edge values of both $\mathbf{q}$ and $\mathcal{G}\phi$ their values in the Gauss-Lobatto points. W^{GL} is the diagonal matrix containing the products of the Gauss-Lobatto weights. Both of these matrices are discussed in more detail in App. C.

From the right hand side of (4.2c) the integral over Ω is evaluated by,

$$\begin{aligned} \int_\Omega \phi^{(0)} \wedge dq^{(1)} &= \sum_{p=1}^{N}\sum_{q=1}^{N} \phi_{p,q} \sum_{i=1}^{N}\sum_{j=1}^{N} (q^\xi_{i,j} + q^\eta_{i,j} - q^\xi_{i-1,j} - q^\eta_{i,j-1}) e_i(\tilde{\xi}_p) e_j(\tilde{\eta}_q) w^G_p w^G_q \ . \\ &= [\mathcal{I}^G_i \mathcal{D}_i \mathbf{q} \ , \ W^{GG}\phi]. \end{aligned} \tag{4.33}$$

Here we call the divergence matrix in the inner part of the domain $\mathcal{D}_i$, with the subscript i to indicate the inner part of the domain, it operates on the cochain $q = \mathcal{R}q^{(n-1)}$ to give the discrete divergence for each cell. This matrix is equal to the incidence matrix $E^{2,1}$ for the divergence. $\mathcal{I}^G_i$ reconstructs from this discrete divergence the divergence in the Gauss-Gauss points which can then be multiplied with ϕ. This multiplication is integrated using the weight matrix W^{GG} containing products of the Gauss weights. We use Gauss-Gauss integration because the nodes of ϕ lie on the Gauss-Gauss points, thus there is no need for interpolation of ϕ.

From the right hand side of side of (4.2c), the boundary integral is evaluated by,

$$\begin{aligned} \int_{\partial\Omega} \phi^{(0)} \wedge q^{(1)} &= \sum_{p=1}^{N}\sum_{i=1}^{N} e_i(\tilde{\xi}_p) \left\{ q^\eta_{i,N}\phi_{p,N+1} - q^\eta_{i,0}\phi_{p,0} \right\} w^G_p + \dots \\ &\qquad \sum_{q=1}^{N}\sum_{j=1}^{N} e_j(\tilde{\eta}_q) \left\{ q^\xi_{N,j}\phi_{N+1,q} - q^\xi_{0,j}\phi_{0,q} \right\} w^G_q \\ &= [\mathcal{I}^G_b \mathcal{D}_b \mathbf{q} \ , \ W^G_b \phi]. \end{aligned} \tag{4.34}$$

Here the interpolator $\mathcal{I}^G_b$ operating on $\mathcal{D}_b\mathbf{q}$ reconstructs this oriented flux of $\mathbf{q}$ at the Gauss points on the boundary such that its product with ϕ can be numerically integrated by the Gauss weights from W^G. $\mathcal{D}_b$ indicates the orientation of the k-cell on the boundary, containing only positive and negative unit values. We use the subscript b to indicate the boundary of the domain. It is because of this integral that we create the Extended Gauss mesh, opposed to just a Gauss mesh with only points in the inner part of Ω. Without the boundary nodes ϕ the final algebraic system of equations that discretizes the Poisson equation is not symmetric. Note that also the exact definition of these matrices is given in App. C

If we now substitute (4.2c) by its discrete counterpart,

$$[\mathcal{I}^{GL}\tilde{\mathbf{q}}, W^{GL}\mathcal{I}^{GL}\mathcal{G}\phi] = [\mathcal{I}_b^G\mathcal{D}_b\mathbf{q}\ ,\ W_b^G\phi] - [\mathcal{I}_i^G\mathcal{D}_i\mathbf{q}\ ,\ W^{GG}\phi]. \tag{4.35}$$

And reformulate the right hand side by merging $\mathcal{D}_i$ and $\mathcal{D}_b$ into a single discrete divergence operator $\mathcal{D}$ we get the expression,

$$[\mathcal{I}^{GL}\mathbf{q}, W^{GL}\mathcal{I}^{GL}\mathcal{G}\phi] = [\mathcal{I}^G\mathcal{D}\mathbf{q}\ ,\ W^G\phi]. \tag{4.36}$$

With,

$$\mathcal{D} = \begin{pmatrix} -\mathcal{D}_i \\ \mathcal{D}_b \end{pmatrix}, \quad \mathcal{I}^G = \begin{pmatrix} \mathcal{I}_i^G & 0 \\ 0 & \mathcal{I}_b^G \end{pmatrix}, \quad W^G = \begin{pmatrix} W_i^G & 0 \\ 0 & W_b^G \end{pmatrix}. \tag{4.37}$$

If (4.36) is to hold for any $\mathbf{q}$ and ϕ then we can rewrite this to,

$$\begin{aligned} \mathbf{q}^T\mathcal{I}^{GL^T}W^{GL}\mathcal{I}^{GL}\mathcal{G}\phi &= \mathbf{q}^T\mathcal{D}^T\mathcal{I}^{G^T}W^G\phi \\ \underbrace{\mathcal{I}^{GL^T}W^{GL}\mathcal{I}^{GL}}_{A}\mathcal{G} &= \mathcal{D}^T\underbrace{\mathcal{I}^{G^T}W^G}_{B}. \end{aligned} \tag{4.38}$$

Which defines the discrete **grad** operator as,

$$\mathcal{G} = \mathbf{A}^{-1}\mathcal{D}^T\mathbf{B}. \tag{4.39}$$

when we combine this formulation of the gradient operator the Poisson equations is discretized as,

$$d \star d\phi^{(0)} = f^{(2)} \tag{4.40a}$$

$$\mathcal{D}\mathbf{A}^{-1}\mathcal{D}^T\mathbf{B}\vec{\phi} = \mathbf{f}. \tag{4.40b}$$

And by premultiplying with $\mathbf{B}^T$ we have the symmetric representation

$$\mathbf{B}^T\mathcal{D}\mathbf{A}^{-1}\mathcal{D}^T\mathbf{B}\vec{\phi} = \mathbf{B}^T f. \tag{4.41}$$

Our goal is to create a Laplace operator with which we solve the Poisson equation , mimicking the underlying connection between **div** and **grad** that make up the Laplace operator. Algebraic topology provides us with the tools to mimic discretely these operations on a discrete level. To do this using a spectral mesh we had to introduce the edge functions. By using the discrete divergence operator as a support for the gradient operator we create a discretization in which we preserve the symmetry of the PDE. Looking closer at (4.40b) we see that because of the symmetry of $\mathbf{A}$, the part $\mathcal{D}\mathbf{A}^{-1}\mathcal{D}^T$ is always symmetric, for solving purposes this is an advantage since for symmetric systems fast solvers are available. The structure of matrix $\mathbf{A}$ and the discrete Laplacian is depicted in Fig. 4.4. The surface plot shows the highly diagonal structure of $\mathbf{A}$, in the other plot the dots represent the non-zero elements. Note the symmetric structure of this Laplacian. It does not show in this sparsity plot but the structure of the discrete Laplacian resembles that of the standard finite volume 5 point stencil. This resemblance shows when we check if,

$$(L^{FV})^{-1}(L^{SEM}) \approx Id. \tag{4.42}$$

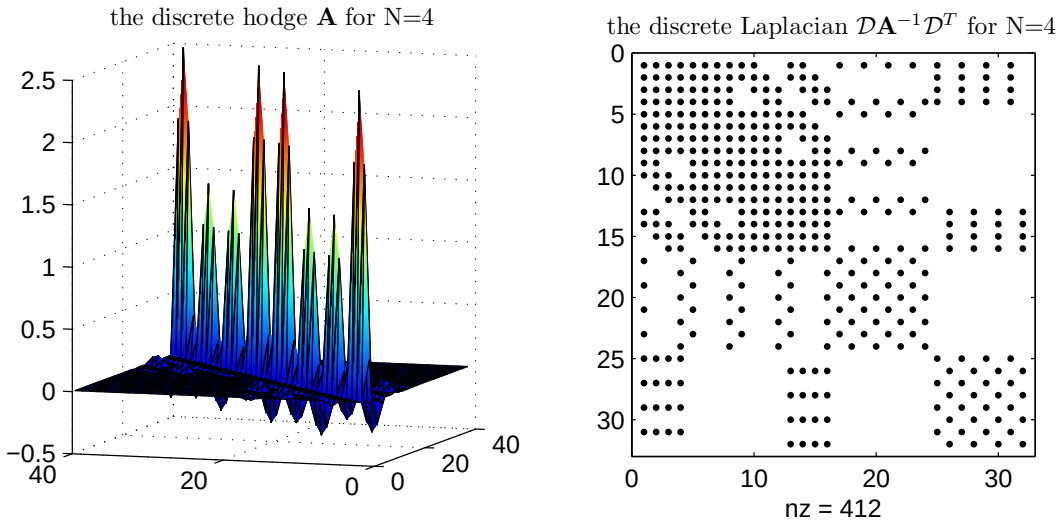


Figure 4.4: The locations of the nonzero components of the matrices that compose discretization of the Poisson equation.

Where L^{FV} and L^{SEM} are the finite volume Laplacian and the spectral element Laplacian. Fig. 4.5 shows a surface plot of matrix that result from performing (4.42) for $N = 4$. Note that the complete matrix is nearly zero except the diagonal which has a value near unity, this matrix indeed resembles the identity matrix. This means that when solving the discrete system of equations making use of an iterative method, such as a conjugate gradient method one can solve the FV system first, after which we can solve, with a good starting value for the iteration the SEM system. This has potential benefits in speeding up the SEM calculation. Another approach is to use matrix L^{FV} as a preconditioner for L^{SEM}, Fig 4.6 shows the eigenvalues of the preconditioned system. These eigenvalues are positioned near unity on the real axis.

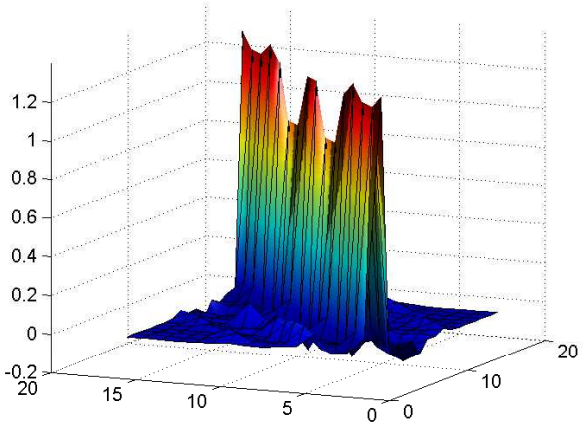


Figure 4.5: A surface plot of $(L^{FV})^{-1}(L^{SEM})$ showing that this is approximately Id

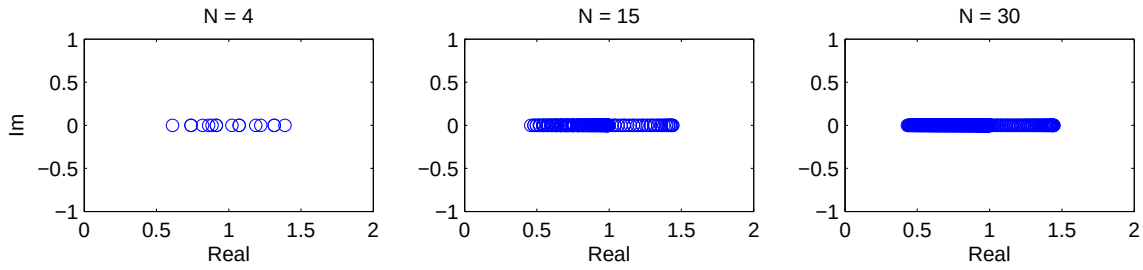


Figure 4.6: The eigenvalues of the preconditioned system for $N = 4, 15$ and 30

Chapter 5

Results on orthogonal domain

Having derived a mimetic formulation of the Poisson equation we show here strategies to solve the resulting system of equations. We restrict ourselves to Dirichlet and Neumann boundary conditions, by which we impose on the boundary either a known ϕ or $\mathbf{q}$. We will see how well the conservation relation is satisfied, that the solutions for ϕ and $\mathbf{q}$ converge towards their exact solution spectrally and that their solutions are optimal for the type of basis functions that are used.

5.1 Boundary conditions and solution procedure

We solve the final system of equations using either Dirichlet or Neumann boundary conditions, boundary conditions of the mixed (Robin) type should be possible, but we do not consider those boundary conditions. Robin boundary conditions require a Hodge because these are defined on the another cell complex than $\mathbf{q}$. Say we have a Dirichlet boundary condition (known ϕ at a given boundary), then we impose this strongly . Now lets rewrite the discrete Poisson equation (4.40b) as,

$$\begin{bmatrix} L_{ii} & L_{ib} \\ L_{ib} & L_{bb} \end{bmatrix} \begin{bmatrix} \phi_i \\ \phi_b \end{bmatrix} = \begin{bmatrix} f_i \\ f_b \end{bmatrix}. \tag{5.1}$$

Where the index i indicates that this part of the matrix operates on the internal ϕ nodes, b indicates the boundary. Then when we impose the Dirichlet boundary conditions strongly by solving the following system,

$$[L_{ii}][\phi_i] = [f_i] - [L_{ib}][\phi_b]. \tag{5.2}$$

The symmetry of the final system of equations is retained, and the boundary conditions for ϕ influence the solution by a change in the forcing vector $[f_i]$.

Contrary to the Dirichlet boundary conditions, the Neumann boundary conditions are imposed weakly. To see how this works we rewrite (4.40b) into the following system,

$$\begin{bmatrix}\mathcal{D}_i\\ \mathcal{D}_b\end{bmatrix}[\mathcal{G}][\phi] = \begin{bmatrix}f\\ q_b\end{bmatrix}. \tag{5.3}$$

Here q_b is the vector containing the integrated fluxes with a negative or positive sign depending on the orientation of the k-cell.

Now, to solve the final system of equations there are two main formulations which we propose. The first formulation is the one where we explicitly find a formulation of (4.40b), this means we use Matlab® to first efficiently find an inverse of $\mathbf{A}$ and construct the system of equations and then solve for ϕ. In constructing the complete system with the form (4.40b) we have to multiply $\mathcal{D}\mathbf{A}^{-1}\mathcal{D}^T$ with matrix $\mathbf{B}$. But matrix $\mathbf{B}$ is very dense. Especially when the polynomial degree of the element (N) increases, multiplying $\mathbf{B}$ with any other relatively full matrix is a computationally expensive operation. The resulting matrix is also very full and thus is costly to invert. Therefore the final system is solved in two steps. In the first step we solve for $\mathbf{B}\phi$ by solving the system,

$$[\mathcal{D}\mathbf{A}^{-1}\mathcal{D}'][\mathbf{B}\phi] = f. \tag{5.4}$$

In the second step we solve for ϕ. Even though these are two steps, it takes about 30% less time to compute the solution.

In the second formulation we do not aim to formulate (4.40b) completely and then solve for ϕ, but rather we create a larger system (5.5) in which we solve for $\mathbf{q}$ and ϕ simultaneously.

$$\begin{bmatrix}-\mathbf{A} & \mathcal{D}^T\mathbf{B}\\ \mathcal{D} & 0\end{bmatrix}\begin{bmatrix}\mathbf{q}\\ \phi\end{bmatrix} = \begin{bmatrix}0\\ f\end{bmatrix}. \tag{5.5}$$

The advantage of this is that we do not have to find the gradient matrix $\mathcal{G}$ explicitly thus $\mathbf{A}$ does not have to be inverted directly. In this approach it is also faster to first solve the system.

$$\begin{bmatrix}-\mathbf{A} & \mathcal{D}^T\\ \mathcal{D} & 0\end{bmatrix}\begin{bmatrix}\mathbf{q}\\ \mathbf{B}\phi\end{bmatrix} = \begin{bmatrix}0\\ f\end{bmatrix}. \tag{5.6}$$

This gives the flux $\mathbf{q}$ and whenever needed we can solve for ϕ afterwards. The downside of this approach is that the final system of equations is larger and for this standard non-deformed GL grid costs actually more time to invert. But as we will see later, when moving to transformed grid the matrix $\mathbf{A}$ becomes very full, in that case pays off to solve for $\mathbf{q}$ and ϕ simultaneously.

5.2 Error norm definitions

In order to evaluate the quality of a solution one must define a norm that quantifies the difference between the exact and the numerical solution. One way to quantify the error is by

using difference between the exact and approximate solution ($\mathbf{x}$) in the L_2- norm, which is typically defined by,

$$\| \mathbf{x}-\mathbf{x}_{ex} \|_2 = \left(\int_\Omega (\mathbf{x}-\mathbf{x}_{ex})^T (\mathbf{x}-\mathbf{x}_{ex}) \right)^{1/2} d\Omega. \tag{5.7}$$

Here $\mathbf{x}$ is the solution of which we need to quantify the error, $\mathbf{x}_{ex}$ is the exact solution and Ω is the domain in which the error is calculated. But having reformulated the system of equations and its variables in terms of differential geometry, we look to quantify the error using a formulation that is more natural using differential geometry.

$$\| \mathbf{x}-\mathbf{x}_{ex} \|_2 = \left(\int_\Omega (x^{(k)}-x^{(k)}_{ex}) \wedge \star (x^{(k)}-x^{(k)}_{ex}) \right)^{1/2}. \tag{5.8}$$

These definitions are actually equal, this is a consequence of the relation that defines the Hodge star operator (2.25). However in their numerical evaluation these definitions do differ. This is because the k-form $x^{(k)}$ and its Hodge dual $x^{(n-k)}$ are defined on different, dual or primal, cell complexes.

5.2.1 Evaluation error $u^{(1)}$

In this section we show how to evaluate the error of the 1-form $u^{(1)}$ numerically. In order to properly evaluate (5.8) we need to consider $u^{(1)}$ and its discretization. The most natural discretization is done on the same grid on which $\phi^{(0)}$ is discretized. In order to calculate $Ru^{(1)}$ we make use of the fact that that, $u^{(1)} = d\phi^{(0)}$. This means that the integral of $u^{(1)}$ along a line segment is equal to the difference in ϕ at the endpoints of the line segment.

$$\int_a^b u^{(1)} = \phi^{(0)}(b) - \phi^{(0)}(a). \tag{5.9}$$

Therefore $\mathcal{R}u^{(1)}$ is connected to the same cell-complex as $\mathcal{R}\phi^{(0)}$. One can show then that $\mathcal{R}u^{(1)} = \mathcal{D}^T \mathcal{R}\phi^{(0)}$. So when ϕ is calculated we can quite easily find $\mathbf{u}$ using the transpose of the discrete divergence operator. This means that we can evaluate (5.8) for $x = u^{(1)}$,

$$\| \mathbf{u}_{num}-\mathbf{u}_{ex} \|_2 = \left(\int_\Omega \left(\mathcal{I}u^{(1)}_{num} - u^{(1)}_{ex} \right) \wedge \left(\mathcal{I}q^{(n-1)}_{num} - q^{(n-1)}_{ex} \right) \right)^{1/2}, \tag{5.10}$$

where q_{num} is the obtained discrete solution. This integral must be evaluated numerically for which we use a Gauss Lobatto grid of size M where $M \gg N$. So the integration order is higher than the polynomial degree in which the solution is expanded. The error of $\mathbf{u}_{num}$

fully written out reads,

$$\| \mathbf{u}_{num} - \mathbf{u}_{ex} \|_2 = \left(\sum_{n=1}^{M} \sum_{m=1}^{M} \Bigg[\left(\sum_{j=1}^{N} \sum_{i=1}^{N+1} u^{\xi}_{i,j} \overline{e}_i(\xi_n) \tilde{h}_j(\eta_m) \; - \; \mathbf{u}^{\xi}_{ex} \right) \left(\sum_{i=0}^{N} \sum_{j=1}^{N} q^{\xi}_{i,j} h_i(\xi_n) e_j(\eta_m) \; - \; \mathbf{q}^{\xi}_{ex} \right) + \left(\sum_{i=1}^{N} \sum_{j=1}^{N+1} u^{\eta}_{i,j} \overline{e}_j(\eta_m) \tilde{h}_i(\xi_n) \; - \; \mathbf{u}^{\eta}_{ex} \right) \left(\sum_{i=1}^{N} \sum_{j=0}^{N} q^{\eta}_{i,j} e_i(\xi_n) h_j(\eta_m) \; - \; \mathbf{q}^{\eta}_{ex} \right) \Bigg] w^{GL}_n w^{GL}_m \right)^{\frac{1}{2}}. \tag{5.11}$$

5.2.2 Evaluation error ϕ

In a similar way we can determine the error of $\phi^{(0)}$ which is given by,

$$\begin{aligned} \| \phi - \phi_{ex} \|_2 &= \left(\int_{\Omega} (\phi^{(0)} - \phi^{(0)}_{ex}) \wedge \star (\phi^{(0)} - \phi^{(0)}_{ex}) \right)^{1/2} \\ &= \left(\int_{\Omega} (\mathcal{I}\mathcal{R}\phi^{(0)} - \phi^{(0)}_{ex}) \wedge (\mathcal{I}\mathcal{R} \star \phi^{(0)} - \star \phi^{(0)}_{ex}) \right)^{1/2}. \end{aligned} \tag{5.12}$$

Here the term $\mathcal{R}\phi^{(0)}$ is known since these are the nodal values of $\phi^{(0)}$, the term $\star\phi^{(0)}$ however needs to be calculated as a postprocessing step. We can do this by making use of definition of the Hodge $\star$ (2.25). The expansion of $\star\phi^{(n)}$ is given by,

$$\star\phi^{(0)} = \sum_{i=1}^{N} \sum_{j=1}^{N} \star\phi_{i,j} e_i(\xi) e_j(\eta). \tag{5.13}$$

By coupling the inner product with the wedge product we can calculate the coefficients $\star\phi_{i,j}$. We do this by numerically evaluating,

$$\begin{aligned} &\int_{\Omega} (\phi, \phi) \omega_n = \int_{\Omega} \phi \wedge \star\phi \\ &\vec{\phi}^T W^G \vec{\phi} = \vec{\phi}^T W^G I^G_i \vec{\phi^\star} \quad \longrightarrow \quad \vec{\phi^\star} = (I^G_i)^{-1} \vec{\phi}. \end{aligned} \tag{5.14}$$

Here the vectors $\vec{\phi}$ and $\vec{\phi^\star}$ contain the coefficients of the expansions of $\phi^{(0)}$ and $\star\phi^{(0)}$ respectively. The matrix I^G_i performs the interpolation, defined by (5.13), to the integration points. The integration matrix W^G cancels from the equation. With $\mathcal{R} \star \phi^{(0)}$ known we can evaluate the error (5.12).

5.3 Test case for orthogonal domain

In this section we look at convergence properties of the proposed spectral method, we also look at the divergence equation and how well this is satisfied. Since the basis functions are

polynomials we test the convergence properties using a non-polynomial function, a trigonometric function. These are well approximated using polynomials but are never exact. Say we have a potential given by,

$$\phi^{(0)}(x,y) = sin(n\pi\xi)sin(m\pi\eta), \tag{5.15a}$$

then by taking the d first and the $d\star$ next, $u^{(1)}$ and forcing $f^{(n)}$ are given by.

$$d\phi^{(0)} = \underbrace{m\pi cos(n\pi\xi)sin(m\pi\eta)}_{u_\xi}\, d\xi \;+\; \underbrace{m\pi sin(m\pi\xi)cos(m\pi\eta)}_{u_\eta}\, d\xi d\eta \tag{5.15b}$$

$$dq^{(n-1)} = d\star d\phi^{(0)} = f^{(n)} = -2\pi^2(m^2)sin(m\pi\xi)sin(m\pi\eta)d\eta. \tag{5.15c}$$

We can vary the complexity of the solution by varying n or m, the higher n the more complex the solution. Fig. 5.1 shows the potential for $m = 1$ and $m = 3$.

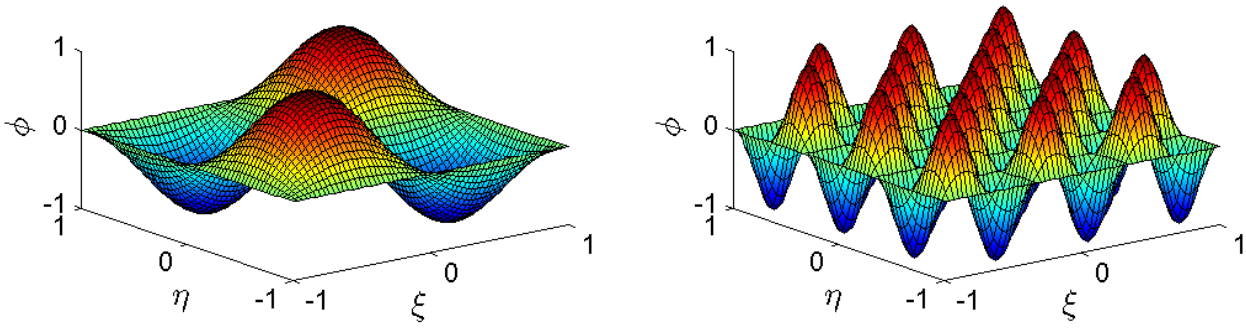


Figure 5.1: The two test cases $\phi(\xi,\eta) = \sin(\phi\pi\xi)\sin(\pi\eta)$ and $\phi(\xi,\eta) = \sin(3\pi\xi)\sin(3\pi\eta)$

Before we solve the system we must setup a correct forcing for the right hand side. This involves calculating $\mathcal{R}f^{(2)} = f_{i,j}$ and in the case of Neumann boundary conditions this also involves evaluating and $\mathcal{R}q^{(1)} = \mathbf{q}_{i,j}$ for $(i,j) \in \partial\Omega$. Both of these involve integration. While for this particular case the integration can be done analytically, this is generally not the case. Therefore the integration is done using Gauss quadrature, this also makes the testing easy for different types of functions. Gauss integration can be chosen to be arbitrarily accurate, so for smooth functions and we do not have to introduce significant errors in doing this. In App. D we explain the procedure. Note also that we do not use GL integration, though this could be done; we prefer Gauss integration. This is partially because Gauss integration does not involve the endpoints of the domain in which the integration needs to be done. Values at the endpoints or corner points of elements can be troublesome in the sense that sometimes jumps in the value of the integrand occur at these locations. We circumvent any problems that might occur in trying to associate values to these points by not using them in the numerical integration.

5.3.1 Convergence results

Figure 5.2 shows the error (ϵ) in the norm defined by (5.7) against the element order N. This figure shows four lines, these represent:

- $\|\mathcal{I}\phi_{num} - \phi_{ex}^{(0)}\|_{L^2}$ The error of the numerical solution of ϕ.
- $\|\mathcal{I}\mathcal{R}\phi_{ex}^{(0)} - \phi_{ex}^{(0)}\|_{L^2}$ The error of expanding ϕ using exact nodal values.
- $\|\mathcal{I}u_{num} - u_{ex}^{(1)}\|_{L^2}$ The error of the numerical solution of u
- $\|\mathcal{I}\mathcal{R}u_{ex}^{(1)} - u_{ex}^{(1)}\|_{L^2}$ The error of expanding $u^{(1)}$ using exact edge values.

Now clearly from Fig. 5.2 we see that in both the case of $m = 1$ and $m = 3$ the error decays exponentially, thus there is spectral convergence. The error for $m = 3$ does decay slower, this however, is to expected since for $m = 3$ both $\phi^{(0)}$ and $q^{(n-1)}$ are more complex than in the case of $m = 1$. Therefore the polynomial order that is needed to approximate the solution is higher and consequently the accuracy is about an order less. Important is that the convergence rate itself is the same for ϕ as for u.

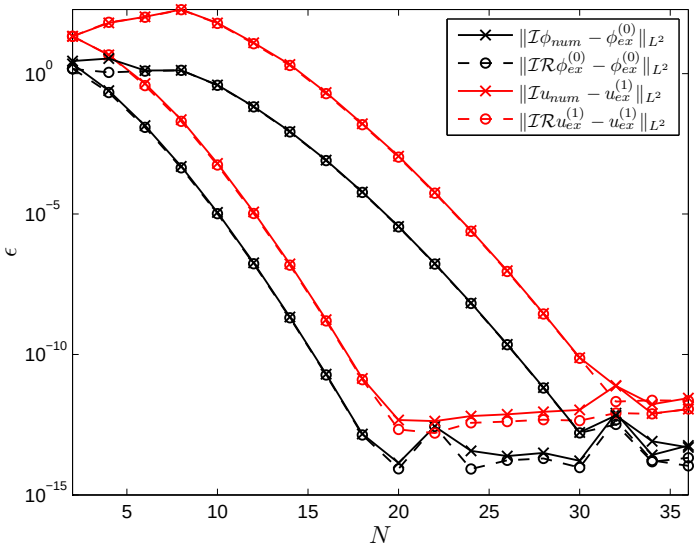


Figure 5.2: Convergence of the errors for the solutions $\mathcal{I}\phi_{num}$ and $\mathcal{I}u_{num}$ and the polynomial expansion using exact nodal values $\mathcal{I}\mathcal{R}\phi_{ana}^{(0)}$

Fig. 5.2 also shows the error made by expanding the solution in terms of its exact nodal values. When the error of the solution ($\|\mathcal{I}\phi_{num} - \phi_{ex}^{(0)}\|_{L^2}$) is the same as that of the analytical expansion ($\|\mathcal{I}\mathcal{R}\phi_{ex}^{(0)} - \phi_{ex}^{(0)}\|_{L^2}$), which is the interpolation error, the solution is optimal. This is indeed the case, only for very high N there is a small discrepancy. This is due to the fact

that the matrix inversions introduce small errors in the order of machine accuracy, therefore when the errors start to approach this accuracy the numerical expansion and the analytical expansions are no longer equal. Since the errors are equal one could assume that the nodal values in the EG points are exact. This however proves not to be the case. The values of ϕ_{ex} and ϕ_{num} differ in the order of the interpolation error. As we can see from Fig. 5.3. In the right plot the difference $\phi_{num} - \phi_{ex}$ is given and also the total sum of this difference. The total sum of this difference is zero. Apparently the fact that this sum of the difference is zero is an indicator of the fact the our solution has the same error as the interpolation error. Looking at this from another angle this means that we have found a solution that *globally* minimises the difference between the interpolation of the numerical and the analytic solution. Globally this difference is $O(10^{-16})$, *locally* this difference is in the order of the interpolation. The error of u_{num} is larger in magnitude than the error of ϕ_{num}, this could be explained by the fact that the expansion of q is partially done using the edge functions. These edge functions are accurate when integrated over and less accurate for actual function approximations. From the fact that the error of $\mathcal{I}u_{num}$ is equal to that of $\mathcal{I}\mathcal{R}u_{ex}^{(1)}$ we know that the solution, u_{num}, is optimal as well.

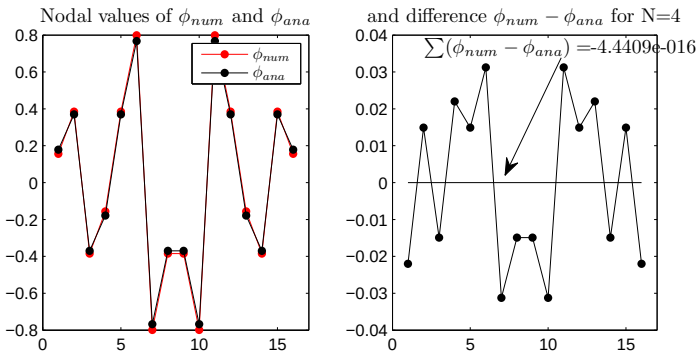


Figure 5.3: (Left) The exact and numerical nodal values of $\phi^{(0)}$ (Right) The difference between these numerical and analytical values and the total sum of this difference. ($N = 4$)

5.3.2 Conservation

The previous example shows that for a smooth solution the error decays exponentially. One can also look at how well the divergence relation is satisfied. We show this in Fig 5.4, here we look at the value $Dq - f$ for the test problem (5.15b). Here D is the discrete and purely topological divergence operator, q is the vector containing $\mathcal{R}q^{(n-1)}$ and f the discrete forcing function $Rf^{(n)}$. The error is checked in two norms, the first is the L^1 norm which is nothing more than the average of magnitude of $Dq - f$.

$$\|\epsilon\|_{L^1} = \int_\Omega |\epsilon| \tag{5.16}$$

the second is the L^∞ or maximum norm which gives the maximum value of the vector $Dq - f$. Ideally in both these norms $Dq - f$ should be zero, in that case the divergence relation is

satisfied exactly. Fig 5.4 shows then that divergence relation is satisfied, *for any element order*, up to machine accuracy. This result justifies the previously used term "spectral finite volume method". One of the key points of a finite volume method is conservation, we hereby show that this spectral method is also conservative.

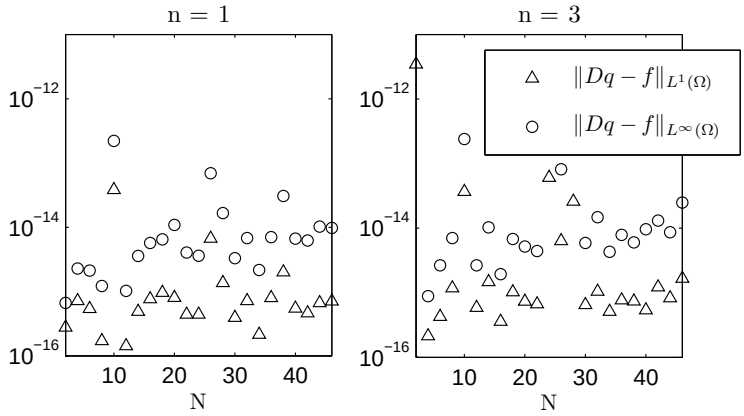


Figure 5.4: The conservation measured as the average value and in the Infinity norm

5.4 Comparison error formulation

To calculate the error we use a formulation that involves the wedge product of two dual k-forms. A more conventional approach is to use the standard error definition using the L_2-norm given in (5.7). This formulation requires less post processing and should, by definition, give similar values for the error. Table 5.1 shows for a given element size N the corresponding

N	$L_2\phi$	$L_2^\wedge\phi$	$L_2\mathbf{u}_{num}$	$L_2^\wedge\mathbf{u}_{num}$
4	0.2477	0.2477	0.2598	0.0583
6	0.0140	0.0140	0.0183	0.0052
8	0.4947e-3	0.4947e-3	0.7473e-3	0.1752e-3
10	0.1135e-4	0.1135e-4	0.6786e-4	0.0363e-4
14	0.2169e-8	0.2169e-8	0.3270e-8	0.0557e-8
18	0.3228e-12	0.1796e-12	0.2137e-12	0.0236e-12

Table 5.1: Table showing the error calculated using convenional method or using wedge product

values for the errors of $\mathbf{u}$ and ϕ, here the symbol L_2 stands for the conventional error (5.7) and $L_2^\wedge$ stands for the error calculation using (5.8). Note that for ϕ it makes no difference. This is because of the way we calculate $\mathcal{R} \star \phi^{(0)}$. Here when we switch from the 0-cochain to the 2-cochain no information is lost and the order of the basis functions used does not change, this is why the two different error formulations yield exactly the same result. For the error of $\mathbf{u}$ (or $\mathbf{q}$) the two different approaches do yield different results. The new formulation has a lower error in the absolute sense. The main reason for this is because the conventional error formulation contains a square[2] the new formulation contains a product of two k-forms

that, though they have very similar values, do not have to be completely equal. Multiplying these two k-forms can cause a local high difference from one k-form to cancel against local low difference of the other k-form. With difference we mean difference from the exact solution to the numerical one. That these two differences are not always equal is shown in Fig. 5.5.

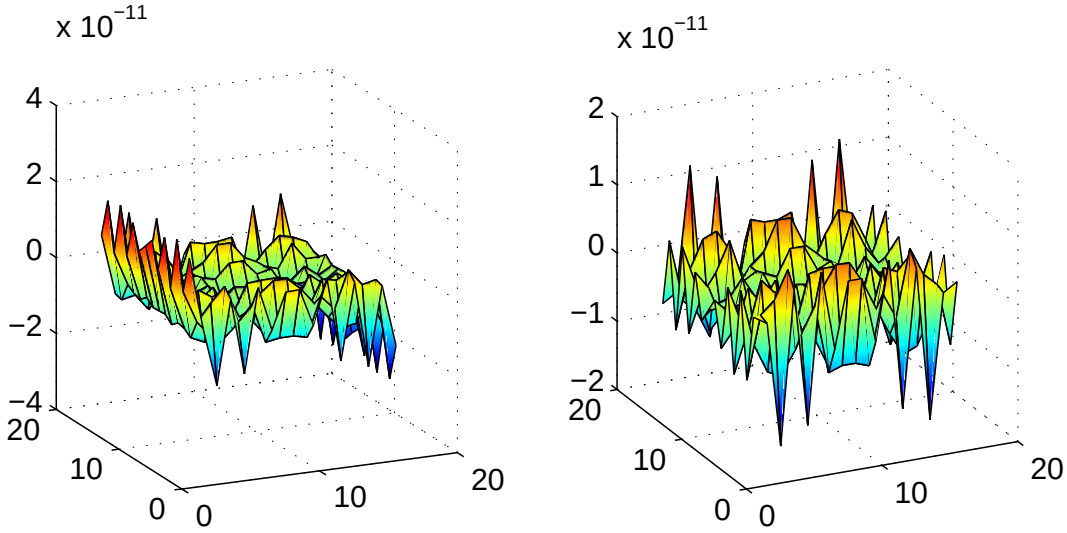


Figure 5.5: (Left) the difference $\left(\mathcal{IR}u^{(1)}_{num} - u^{(1)}_{ex}\right)$ (Right) the difference $\left(\mathcal{IR}q^{(n-1)}_{num} - q^{(n-1)}_{ex}\right)$ are not equal

Chapter 6

Poisson equation on non-orthogonal domains

The compatible spectral element discretization we have presented so far is based on an orthogonal mesh. For any practical application the use of deformed elements is crucial. Therefore we give in this chapter an extension which allows us to solve the Poisson equation on arbitrarily shaped grids. In doing this, we will show that also for deformed grids the discrete system has a purely topological part and a part that contains the information of the shape of the cell-complex (the metric). Using the mapping explained in Chapter 2 we show that the discrete system of equations is only slightly altered when we extend the method from an undeformed orthogonal domain to an arbitrarily shaped curvilinear domain.

6.1 Support operator with mapping

If we now start from (4.2a) and apply the change of variable formula (2.30). Instead of defining the integral (4.2a) in a general curvilinear domain with coordinates (x, y), it is then defined in an orthogonal domain with coordinates (ξ, η) (6.1b). In this domain we perform the integration, this can be done but we need to transform the integrand to this domain too. This is done with the pullback operator.

$$\int_{\Omega} (q^{(n-1)}, \star d\phi^{(0)})\omega^n = \int_{\Omega} q^{(n-1)} \wedge d\phi^{(0)} \tag{6.1a}$$

$$\int_{\Omega'} \Phi^{\star}((q^{(n-1)}, \star d\phi^{(0)})\omega^n) = \int_{\Omega'} \Phi^{\star}(q^{(n-1)} \wedge d\phi^{(0)}). \tag{6.1b}$$

Next we redefine our forms as, ${\Phi^\star}^{-1}\tilde{q}^{(n-1)} = q^{(n-1)}$, ${\Phi^\star}^{-1}\tilde{\phi}^{(0)} = \phi^{(0)}$ and ${\Phi^\star}^{-1}\widetilde{\star d\phi^{(0)}} = \star d\phi$. Where $\tilde{a}^{(k)}$ is the k-form on the domain Ω'. We then insert these into (6.1b), to get

$$\begin{aligned}
\int_{\Omega'} (\ {\Phi^\star}^{-1}\tilde{q}^{(n-1)}, {\Phi^\star}^{-1}\widetilde{\star d\phi^{(0)}}\ )\Phi^\star\omega^n &= \int_{\Omega'} \Phi^\star({\Phi^\star}^{-1}\tilde{q}^{(1)} \wedge d{\Phi^\star}^{-1}\tilde{\phi}^{(0)}) && \text{(6.2a)}\\
&= \int_{\Omega'} \tilde{q}^{(n-1)} \wedge d\tilde{\phi}^{(0)} && \text{(6.2b)}\\
&= \int_{\partial\Omega'} \tilde{\phi}^{(0)} \wedge \tilde{q}^{(1)} - \int_{\Omega'} \tilde{\phi}^{(0)} \wedge d\tilde{q}^{(n-1)}. && \text{(6.2c)}
\end{aligned}$$

The right hand side of (6.2c) is equal to the right hand side of that of the undeformed derivation (4.2c). This means that the right hand side does not change when subject to a mapping. In the integral of the left hand side we see that before we can evaluate the inner product we need to transform our forms to the domain Ω. Another thing is the term $\Phi^\star\omega^n$ which we will show is the determinant of the Jacobian matrix, which appears because the integration region changes shape. In Appendix E some examples of the action of the pullback operator are included.

The derivation (6.2a), (6.2b) and (6.2c) shows that when we define the differential forms on the orthogonal GL square, we can derive a **grad** operator that, combined with the **div** operator, gives the discrete Laplace equation that is defined in a general curvilinear domain. Another interpretation would be that we have transformed our problem defined in Ω to an equivalent problem in Ω'.

6.2 Integral Evaluation

As in the undeformed case we use (6.2c) to find the discrete gradient operator $\mathcal{G}$, based on the discrete divergence, $\mathcal{D}$ such that $\mathcal{G}\phi = \star d\phi$. We have shown in Chapter 4 that the discrete divergence operator can be constructed without referring to the metric of the domain, since this is already incorporated in the integration of $q^{(n-1)}$. It is therefore natural that this part of the formulation does not change when moving to non-orthogonal grids. Here we explain in more detail how the left hand side of (6.2c) is discretized and what the resulting formulation of $\mathcal{G}$ is.

6.2.1 Inner product

Here we give the discretization of the left hand side of (6.2a).

$$\begin{aligned}\int_{\Omega'} (\ {\Phi^\star}^{-1}\tilde{q}^{(1)}, {\Phi^\star}^{-1}\widetilde{\star d\phi^{(0)}}\ )\Phi^\star\omega^n = \\ \sum_{k=0}^{N}\sum_{l=0}^{N}\Bigg[\Bigg\{T^{\xi\xi}_{k,l}\sum_{j=1}^{N}q^{\xi}_{k,j}e_j(\eta_l) + T^{\eta\xi}_{k,l}\sum_{i=1}^{N}q^{\eta}_{k,j}e_i(\eta_l)\Bigg\} \\ \Bigg\{T^{\xi\xi}_{k,l}\sum_{j=1}^{N}(\mathcal{G}\phi)^{\xi}_{k,j}e_j(\eta_l) + T^{\eta\xi}_{k,l}\sum_{i=1}^{N}(\mathcal{G}\phi)^{\eta}_{k,j}e_i(\eta_l)\Bigg\} \quad + \\ \Bigg\{T^{\xi\eta}_{k,l}\sum_{j=1}^{N}q^{\xi}_{k,j}e_j(\eta_l) + T^{\eta\eta}_{k,l}\sum_{i=1}^{N}q^{\eta}_{k,j}e_i(\eta_l)\Bigg\} \\ \Bigg\{T^{\xi\eta}_{k,l}\sum_{j=1}^{N}(\mathcal{G}\phi)^{\xi}_{k,j}e_j(\eta_l) + T^{\eta\eta}_{k,l}\sum_{i=1}^{N}(\mathcal{G}\phi)^{\eta}_{k,j}e_i(\eta_l)\Bigg\}\Bigg]J_{k,l}w^{GL}_k w^{GL}_l,\end{aligned} \tag{6.3}$$

where

$$\begin{aligned}J_{k,l} &= \frac{\partial y}{\partial\eta}\frac{\partial x}{\partial\xi} - \frac{\partial x}{\partial\eta}\frac{\partial y}{\partial\xi}\bigg|_{\xi_k,\eta_l} \\ T^{\xi\xi}_{k,l} &= \frac{\partial x}{\partial\xi}\bigg|_{\xi_k,\eta_l}\frac{1}{J_{k,l}}, \qquad T^{\eta\xi}_{k,l} = \frac{\partial x}{\partial\eta}\bigg|_{\xi_k,\eta_l}\frac{1}{J_{k,l}} \\ T^{\xi\xi}_{k,l} &= \frac{\partial x}{\partial\xi}\bigg|_{\xi_k,\eta_l}\frac{1}{J_{k,l}}, \qquad T^{\eta\xi}_{k,l} = \frac{\partial x}{\partial\eta}\bigg|_{\xi_k,\eta_l}\frac{1}{J_{k,l}}.\end{aligned} \tag{6.4}$$

In this rather large expression we evaluate the integrand in the GL points and integrate using the GL weights. The term $J_{k,l}$ arises from $\Phi^\star\omega^n$, the determinant of the Jacobian. The terms $T^{\cdots}_{k,l}$ come from the transformation of the 1-forms $q^{(n-1)}$ and $\mathcal{G}\phi$. More about the transformation of differential forms is given in Appendix E. Next we write (6.3) in a more dense notation of matrices (6.5).

$$\begin{aligned}\int_{\Omega'} (\ {\Phi^\star}^{-1}\tilde{q}^{(1)}, {\Phi^\star}^{-1}\widetilde{\star d\phi^{(0)}}\ )\Phi^\star\omega^n &= (\mathbf{T}^{-1}I^{GL}\mathbf{q})^T W^{GL}\mathbf{J}(\mathbf{T}^{-1}I^{GL}\mathcal{G}\vec{\phi}) \\ &= \mathbf{q}^T(\underbrace{I^{GL^T}\mathbf{T}^{-T}W^{GL}\mathbf{J}\mathbf{T}^{-1}I^{GL}}_{A}\mathcal{G})\vec{\phi}.\end{aligned} \tag{6.5}$$

The action of these matrices is described in Fig. 6.1, their formulations in App. C. The matrix I^{GL} takes the edge values and interpolates to point values at the GL locations according to (4.15). Matrix T^{-1} takes these point values from Ω' and returns the vector proxy of $q^{(n-1)}$ and $\star d\phi$ in the curvilinear domain Ω. Then the inner product is taken at each of the GL points and integrated over the domain by W^{GL} and the determinant of the Jacobian J. We can now express matrix $\mathcal{G}$ as

$$\mathcal{G} = \mathbf{A}^{-1}\mathcal{D}^T\mathbf{B} \tag{6.6}$$

$$\mathbf{A} = I^{GL^T}\mathbf{T}^{-T}W^{GL}\mathbf{J}\mathbf{T}^{-1}I^{GL}, \quad \mathbf{B} = I^{G^T}W^G. \tag{6.7}$$

As in the case of an orthogonal mesh the final system of equations looks like,

$$\mathcal{D}\mathbf{A}^{-1}\mathcal{D}^T\mathbf{B}\phi = f, \tag{6.8}$$

which is the same as in the undeformed case (4.40b). Only the matrix $\mathbf{A}$ contains two extra matrices that arise from the transformation.

6.2.2 Discretization of $\phi^{(0)}$ and $q^{(1)}$ and $f^{(2)}$

In the previous section we found a new formulation for $\mathcal{G}$ such that we can solve the Poisson equation on non-orthogonal grids. We essentially transform the equation to the orthogonal domain Ω'. In this domain we discretize and solve the equation. To get the final solution we must take this solution and apply the pullback operator, described in Appendix E. So we transform the problem to an analogous one in the domain of Ω', this means we also apply the boundary conditions and forcing in this domain.

$\mathcal{R}\phi^{(0)}$: Since $\phi^{(0)}$ is a 0-form its value does not change when pulled-back or pushed forward from one domain to the other. The discretization of $\phi^{(0)}$ is therefore given by,

$$\phi_{i,j} = R\phi^{(0)} = \phi^{(0)}(\Phi(\xi_i, \eta_j)). \tag{6.9}$$

$Rq^{(n-1)}$: For the discretization of the (n-1)-form q we can use the same definition as in the case where we work with undeformed grids (4.13). So $q^{\xi}_{i,j}$ and $q^{\eta}_{i,j}$ ($\mathcal{R}q^{(n-1)}$) are given by (6.10).

$$\begin{aligned} q^{\xi}_{i,j} &= \int_{\eta_j}^{\eta_{j+1}} q_\xi(\xi_i, \eta)d\eta \quad = \int_{\eta_j}^{\eta_{j+1}} u_x\frac{\partial y}{\partial \eta} - u_y\frac{\partial x}{\partial \eta}\bigg|_{\Phi(\xi_i,\eta)} d\eta \\ q^{\eta}_{i,j} &= \int_{\xi_i}^{\xi_{i+1}} q_\eta(\xi, \eta_i)d\eta \quad = \int_{\xi_i}^{\xi_{i+1}} u_y\frac{\partial x}{\partial \xi} - u_x\frac{\partial y}{\partial \xi}\bigg|_{\Phi(\xi,\eta_i)} d\xi. \end{aligned} \tag{6.10}$$

Note that q_ξ is always normal to the line over which it is integrated. This is how $\mathcal{R}q^{(1)}$ is defined, the integral of the normal flux across the grid lines connecting the Gauss-Lobatto nodes. In Appendix E we show that q_ξ (defined in Ω') can be written in terms of u_x, u_y (that lie in Ω) and the mapping derivatives as indicated in (6.10). Because the $Rq^{(n-1)}$ is an integration of $q^{(n-1)}$ over to the grid lines connecting the GL points, it does not matter in what domain the integral is evaluated. The integrals are invariant under mappings, this is illustrated in Figure 6.2 We will show that by using the integral value of $q^{(n-1)}$ the divergence relation is exactly satisfied. When a boundary can be parameterized, $Rq^{(n-1)}$ can be calculated to any degree of accuracy is desired. This means curves can be followed exactly and do not have to be approximated by straight line segments, the discretization contains the curvature.

$\mathcal{R}f^{(n)}$: The discretization of $f^{(n)}$ is given by (6.11),

$$f_{i,j} = \mathcal{R}f^{(2)} := \iint_{\Phi(\Omega')} {\Phi^\star}^{-1} f^{(2)} = \int_{\xi_{i-1}}^{\xi_i} \int_{\eta_{j-1}}^{\eta_j} f(\Phi(\xi,\eta)) \, \|J\| d\eta d\xi \, , \quad i,j = 1\ldots, N \, . \tag{6.11}$$

Here the determinant of the Jacobian arises again since we integrate the scalar 2-form over a surface. Since $f^{(2)}$ is known in Ω the argument of f is $\Phi(\xi,\eta) = (x,y)$. More on the transformation of 2-forms in Appendix E.

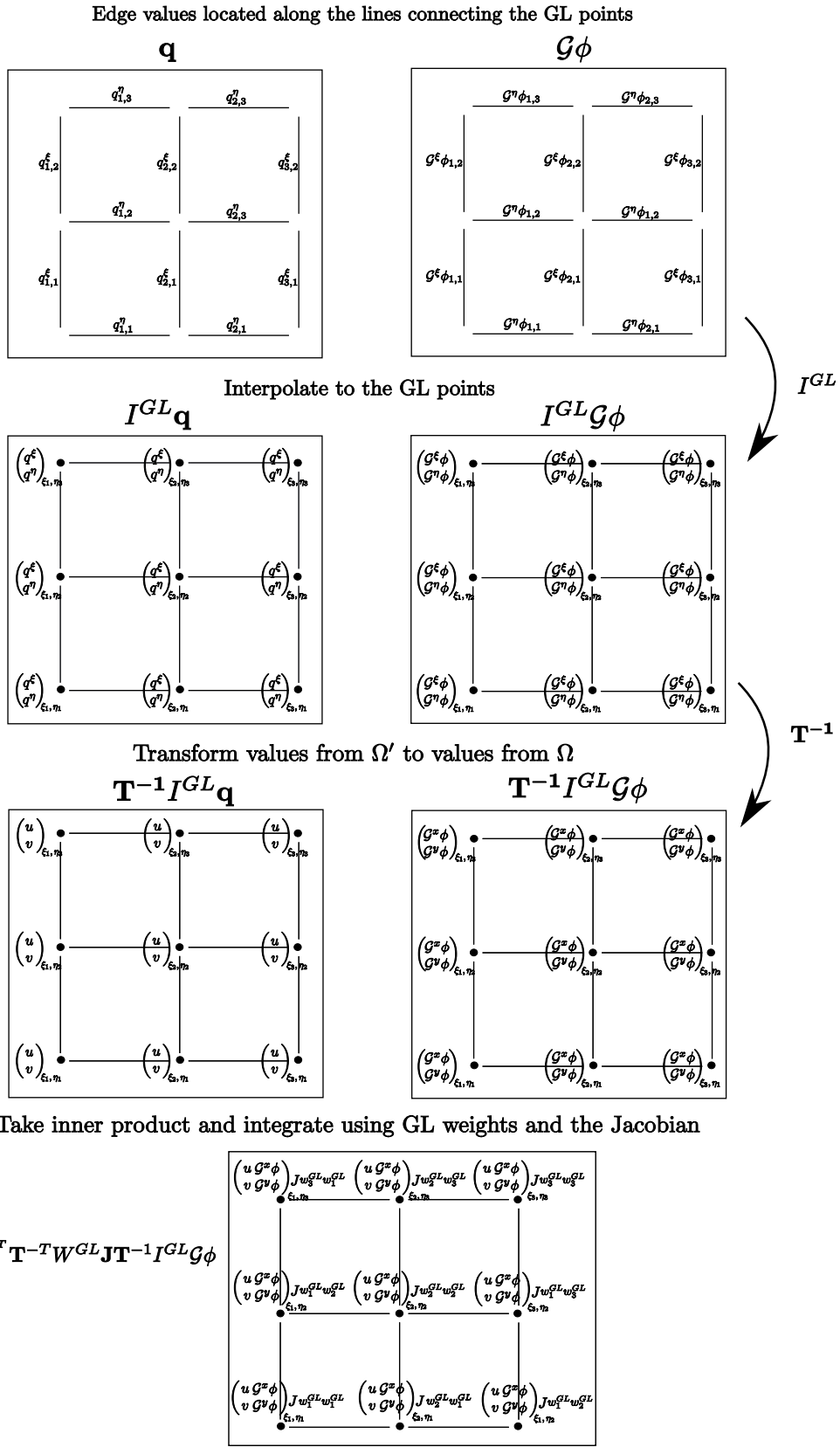


Figure 6.1: Visual representation of matrices, I^{GL} (Interpolation), T^{-1} (Transformation),J(Domain deformation), W^{GL} (Integration

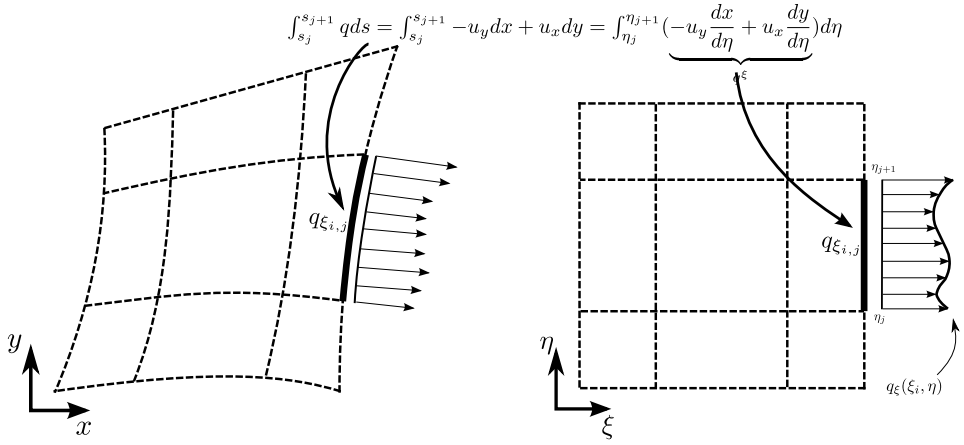


Figure 6.2: The integral edge values remain the same under transformation to different domains

Chapter 7

Results on non-orthogonal domains

For deformed grids, we have shown that it is possible to derive a discrete system of equations that is similar to the system of an undeformed grid. For this we transform the problem to an equivalent one on the orthogonal Gauss-Lobatto grid, on which the k-forms are also defined. In that domain we solve the equations and transform the solution back to the general curvilinear domain. In this chapter we will show a number of results with increasing deformation to show the robustness of the mimetic approach. As an analytical solution we will use (7.1), the same as with testing the undeformed case.

$$\phi^{(0)}(x,y) = \sin(\pi x)\sin(\pi y) \tag{7.1}$$

Note that we refer to the general curvilinear domain with the coordinates (x, y) and when we are in the orthogonal GL domain we use the coordinates (ξ, η). All the mappings that we use take the space of (ξ, η) as a basis and map these coordinates to the coordinates (x, y).

7.1 Errors on deformed domains

In order to determine the accuracy of the solutions obtained, we need to perform an integration of the error in the L_2 norm. When we deal with deformed domains, this means the error definition (5.7) needs to be rewritten. Let us define an error ϵ^2 by,

$$\epsilon^2\omega^n \;=\; (\mathcal{IR}x^{(k)}_{num} - x^{(k)}_{exact}) \wedge \star(\mathcal{IR}x^{(k)}_{num} - x^{(k)}_{exact}). \tag{7.2}$$

The error of the solution $x^{(k)}_{num}$ is then written by,

$$\| \mathbf{x} - \mathbf{x}_{ex} \|_2 \;=\; \int_\Omega \epsilon^2\omega^n. \tag{7.3}$$

When working on deformed elements however we usually want to evaluate this integral in the domain in which the GL is defined, which is not the domain Ω. With the definition of the

smooth map $\Phi : \Omega' \longrightarrow \Omega$ and by making use of (2.32) we rewrite the error definition,

$$\| \mathbf{x} - \mathbf{x}_{ex} \|_2 = \int_{\Omega} \epsilon^2 \omega^n = \int_{\Phi(\Omega')} \epsilon^2 \omega^n = \int_{\Omega'} \Phi^\star(\epsilon^2 \omega^n), \tag{7.4}$$

which is the error definition we will use from now on. According to Frankel (2004) we can say that for any smooth map $\Phi : \Omega' \longrightarrow \Omega$, which might be self intersecting, the integral 7.4 makes sense. We needed this assurance because we will use a self overlapping element to test the properties of the mimetic SEM for deformed elements.

7.2 Case 1: Linear deformation

In this first test case we apply a deformation to the orthogonal GL grid that is the same for every point. The mapping Φ is given by,

$$x(\xi) = \frac{\xi + 1}{2}, \quad y(\xi, \eta) = a\frac{\xi + 1}{2} + \frac{\eta + 1}{2}. \tag{7.5}$$

Here the parameter a determines the amount in which the right boundary is shifted upwards thus changing the amount of skewness in the domain, see Fig (7.1). By taking a uniform

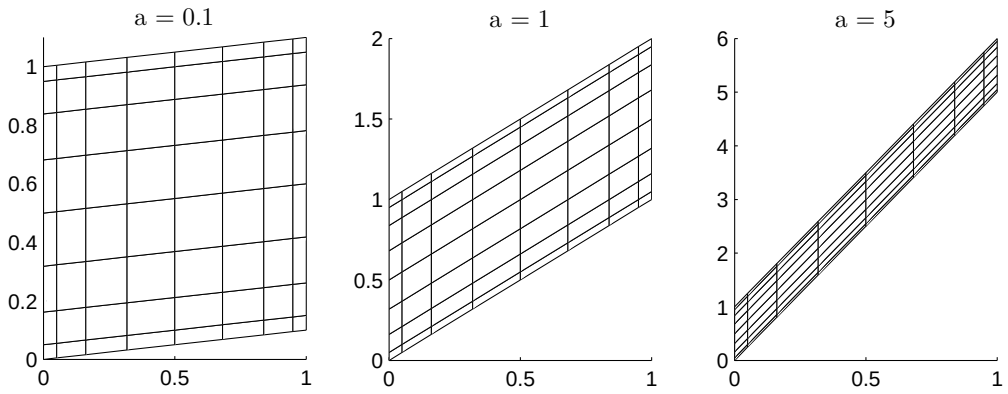


Figure 7.1: Three cases of deformation, $a = 0.1, 1, 5$

deformation the action of the pullback operator is also uniform throughout the field. Which does not introduce any added difficulty in programming of this test case. Using this map the uniform determinant of the Jacobian matrix and the matrix T^{-1} are given by

$$T^{-1} = 2\begin{pmatrix} 1 & 0 \\ a & 1 \end{pmatrix}, \qquad J = \frac{1}{4}. \tag{7.6}$$

The integration of the boundary edge values and the forcing function f, which form the right hand side of our system of equations, is done numerically. Using the same techniques as in the undeformed case we can perform this numerical integration using Gauss integration. The convergence for the three values of a are given in Fig 7.2. As we can see the exponential

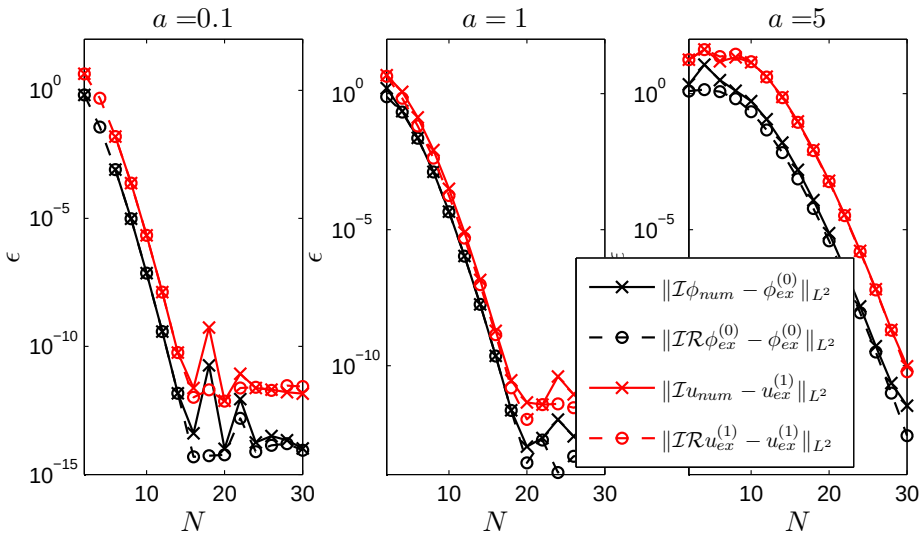


Figure 7.2: The convergence of for the uniformly deformed meshes with shape parameter $a = 0.1$, $a = 1$ and $a = 5$

convergence is retained when working on this uniformly deformed domain. That the convergence plots are nearly linear in this exponential plot means that the convergence rate can be captured with the formula,

$$\epsilon \;=\; \beta e^{-\alpha N} \tag{7.7}$$

To see how the convergence rate responds to the deformation we can determine the value of α in the region of $10^{-6} \leq \epsilon \leq 10^{-10}$ where the convergence is most exponential. The result is that for:

- $a = 0.1$ the convergence value $\alpha = 2.6$
- $a = 1$ the convergence value $\alpha = 2.2$
- $a = 5$ the convergence value $\alpha = 1.6$

This shown that deforming the element more and more brings the rate of convergence down. The more deformation, the higher order polynomials we need to represent our solution. The main cause of this slower convergence is because of the increase in complexity of the solution. If we solve this function on the nearly undeformed grid we need to approximate for 1 peak of a sine wave. However when we stretch the grid, for instance in the case of $a = 5$ we essentially try to approximate the solution through a much larger portion of space, in which a lot more is happening with the function ϕ. So in the case of this uniform deformation we see some deterioration of the convergence speed when increasing the amount of deformation. This is not a fact of deterioration of our numerical scheme but we can ascribe this due to the increase in complexity of the solution that needs to be approximated. This is depicted in Fig 7.3. Fig 7.4 shows how well the divergence relation is satisfied for each of the test cases. Which shows that in all three cases the divergence relation is satisfied up to machine error.

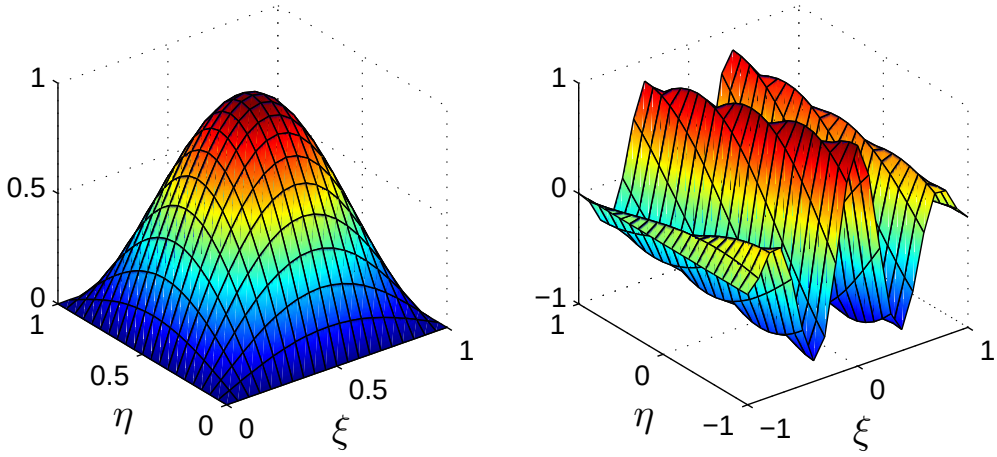


Figure 7.3: (left) the potential for $a = 0$ (right) the potential for $a = 5$

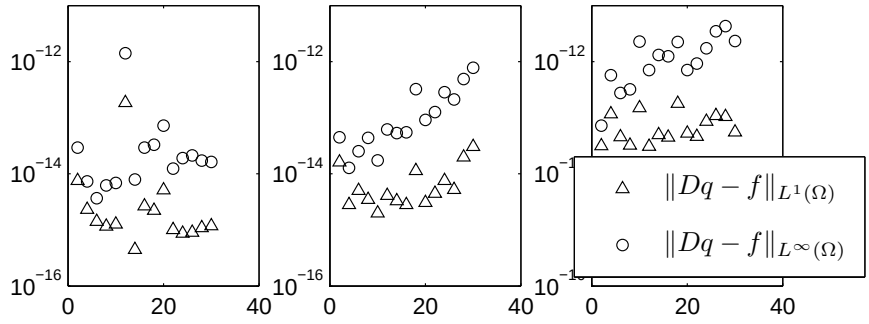


Figure 7.4: The L_1 and the L_∞ norm for $a = 0.1$,$a = 1$ and $a = 5$

7.3 Case 2: Harmonic deformation

The previous test case is not a very challenging one to test the method we are proposing. There is another type of grid which is quite a suitable test case to test the robustness of the discretization. The mapping for this grid is given by,

$$\begin{aligned} x(\xi,\eta) &= \xi + c\sin(\pi\xi)\sin(\pi\eta) \\ y(\xi,\eta) &= \eta + c\sin(\pi\xi)\sin(\pi\eta), \qquad \xi,\eta \in [-1,1]. \end{aligned} \tag{7.8}$$

By increasing the coefficient c we can increase the amount of deformation. Fig 7.5 shows the element shape for three different values of c. For small deformations we see local increases and decreases in the density of the grid, but nothing really serious. One would expect good convergence from this. In the second case there are severe deformations of the grid, some of the cells become non-convex for higher N. In case number three cells are turning in on themselves and overlap, some places have cells lying on top of each other, some of the internal cells reach outside the boundary of the domain.

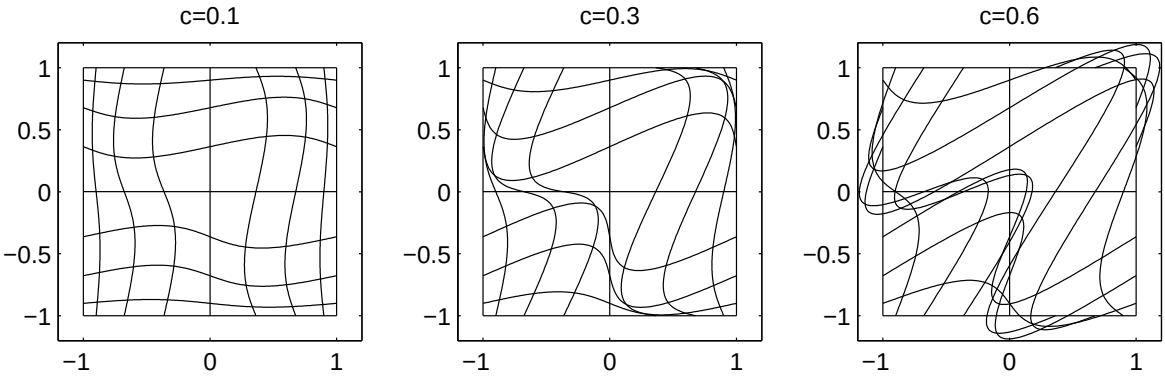


Figure 7.5: Three cases of deformation with shape parameter $c = 0.1$, $c = 0.3$ and $c = 0.6$

When we look at the convergence for these three grids in Fig 7.6 there are a number of things that need attention. The first figure on the left shows two collections of convergence plots. The upper is the one for a shape parameter $c = 0.1$ and the lower the one for $c = 0.0$. The grid that *optically* does not look too bad actually shows convergence that is slower than the undeformed grids convergence. The error of numerical solution ϕ_{num} is equal to that of the expansion using exact coefficients ϕ_{ana}. This means we can ascribe the slower convergence to fact that the ϕ we solve for is less polynomial-like. By distorting the grid the solution becomes harder to approach with polynomial basis functions. The same behavior can be seen for the case where $c = 0.3$, yet to a more extreme extent. In the most most extreme case

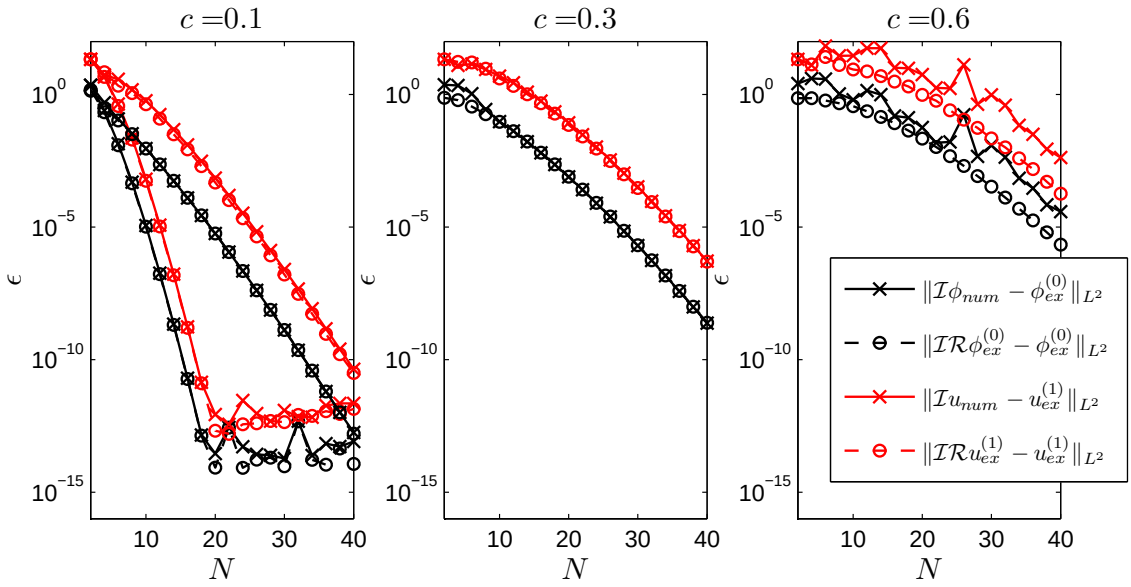


Figure 7.6: Convergence in L_2-norm for deformation shape parameter $c = 0.1$, $c = 0.3$ and $c = 0.6$

where $c = 0.6$ we can see the robustness of this scheme. For this grid we have on average exponential convergence, albeit slow exponential convergence, still $\alpha > 0$. The solution

however is no longer optimal in the sense that the black dotted line (which is the error of $\mathcal{IR}\phi^{(0)}_{exact}$) and the continuous black line which is the error of the numerical solution are no longer on top of each other. It is tempting to think that this must mean that the integration of (6.2c) is no longer exact. Fig 7.7 shows that this proves not to be the case. Fig 7.7 shows three lines, where ϕ_{num2} is the numerical solution in which a higher order integration has been used. It shows that when we increase the accuracy of the integration by adding more integration points, the error does not go down. The solution is actually less accurate for some element sizes. And because the higher order integration points no longer lie on the standard Gauss-Lobatto grid the resulting matrices that need to be inverted become completely full which causes the computation time to increase by a factor of 50. We have stated earlier

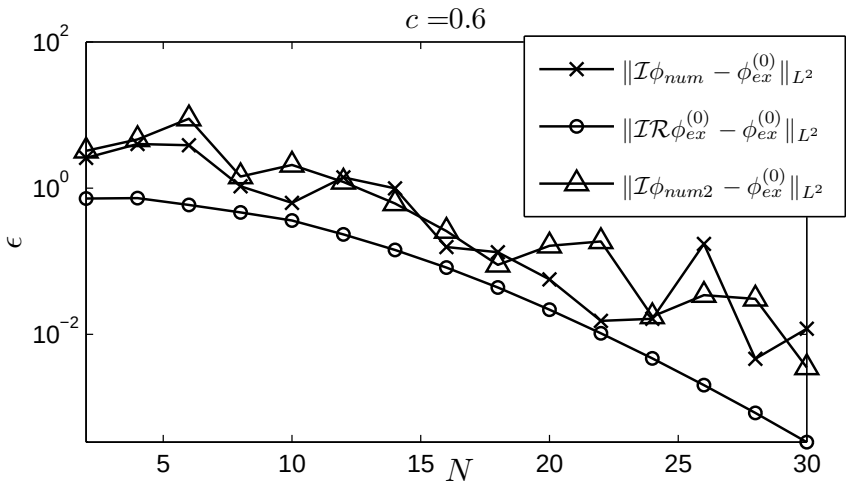


Figure 7.7: Convergence in L_2-norm for deformation shape parameter $c = 0.6$ and higher order integration $(N + 10)$

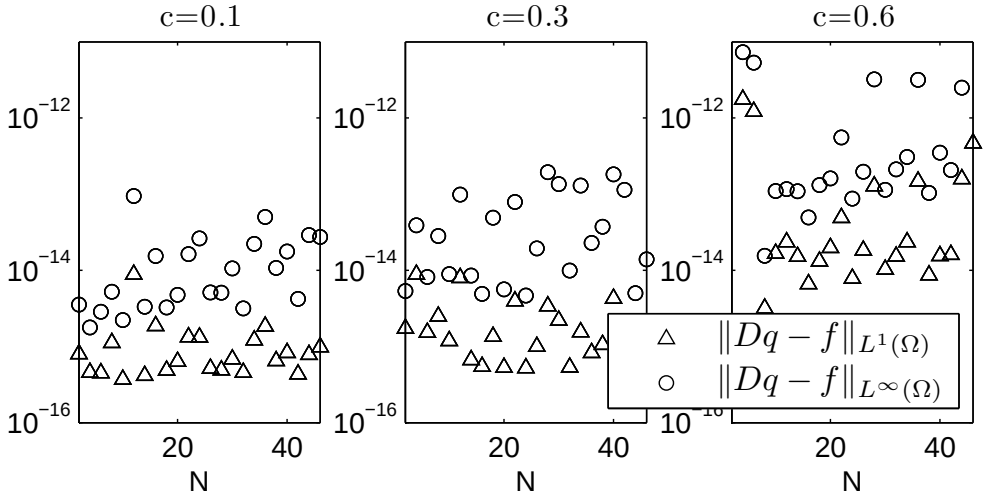


Figure 7.8: The divergence norms for shape parameter $c = 0.1$,$c = 0.3$ and $c = 0.6$

that we exactly satisfy the divergence relation. And indeed we see in Figure 7.8 that in both norms the scheme is conservative up to machine error for all meshes shown in Fig 7.5.

Chapter 8

Multi element

Up to this point we looked at the discretization of the **grad** and **div** operators for one single element. For any physical problem with complex geometries the use of multiple elements is an obvious necessity. The extension to a multiple element method can be done in a variety of ways, and in this chapter we describe our choice. The motivation behind this choice is that we want to reuse as much material developed for the single element case. Our aim is to preserve the property of absolute conservation. The convergence with respect to the characteristic mesh size (h convergence) will not be exponential but will be of an order related to the order of the polynomials used, which is shown with various test cases.

8.1 Cell-complex definition

Being the basis of the single element method we use the integral relation (6.2c) to define implicitly the flux operator $\mathcal{G}$ by the divergence operator $\mathcal{D}$. Based on this integral we propose the following discretization with which we can reuse as much of the single element matrices as possible. Lets first look at the grid for $Rq^{(n-1)}$ (primal cell complex) which is depicted in Fig. 8.1. This is a collection of two separate GL meshes that have one overlapping boundary. Note that these two elements do not share common flux coefficients but each of the elements has flux values q defined at the same location (Compare this with Discontinuous Galerkin or Finite Volume Methods where we have a flux from the left, f_l, and a flux from the right f_r). These extra coefficients have as an advantage that we can discretize inner product on the left hand side of (6.2a) as follows. Let $\langle q, \mathcal{G}\phi \rangle$ denote the inner product from (6.2a). Here we leave out the transformation from the notation since this only complicates the matter and we have seen that the final matrix structure remains the same.

$$\begin{aligned} \int_{\Omega} (q^{(n-1)}, \star d\phi^{(0)}) &= \int_{\Omega_2} (q^{(n-1)}, \star d\phi^{(0)}) + \int_{\Omega_2} (q^{(n-1)}, \star d\phi^{(0)}) \\ [q, A\mathcal{G}\phi] &= [q_1, A_1\mathcal{G}_1\phi_1] + [q_2, A_2\mathcal{G}_2\phi_2]\,. \end{aligned} \tag{8.1}$$

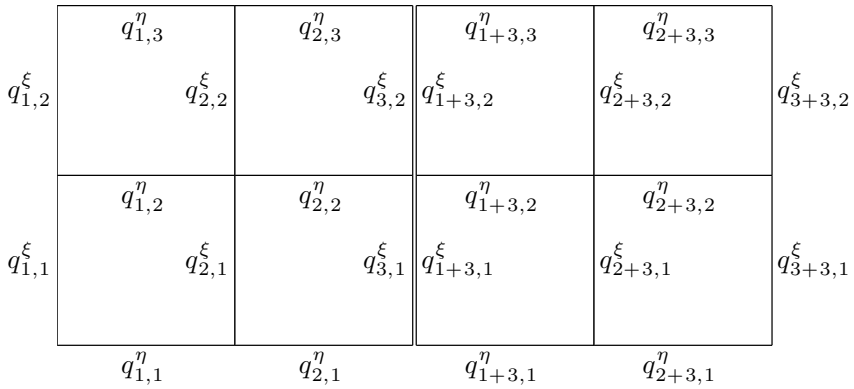


Figure 8.1: The mesh structure for a mesh of 2 by 2 elements

Here the matrix A is just the collection of the single element matrices A_i along the diagonal, so $A = \left(\begin{smallmatrix} A_1 & 0 \\ 0 & A_2 \end{smallmatrix}\right)$. Here A_i is defined as in (6.7).

The locations of the nodes of ϕ are shown in Fig 8.2. Unlike the multiple GL grid, both elements do share common nodes, in the example figure these are nodes $\phi_{3,1}$ and $\phi_{3,2}$. As in

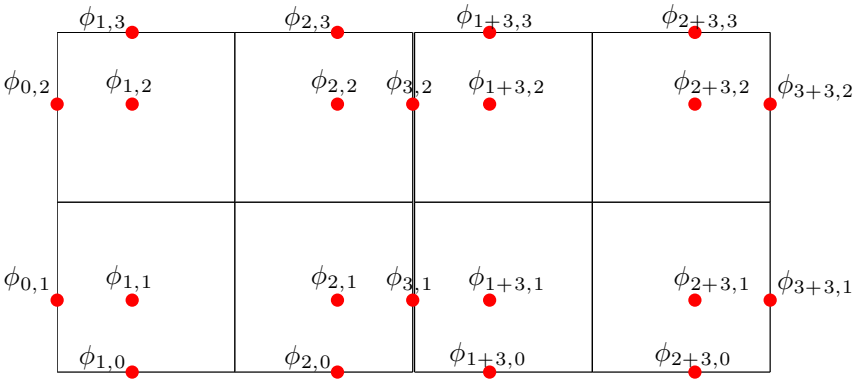


Figure 8.2: The mesh for discretization of ϕ

the single element case we must discretize (6.2a) to find the discretization for the flux operator. We have already worked out the inner product (8.1). What remains is the discretization of,

$$\int_{\partial\Omega} \phi^{(0)} \wedge q^{(1)} - \int_{\Omega} \phi^{(0)} \wedge dq^{(1)}. \tag{8.2}$$

The volume integral over the area is discretized as the sum of the two single element surface integrals.

$$\begin{aligned} \int_{\Omega} \phi^{(0)} \wedge dq^{(1)} &= \int_{\Omega^1} \phi^{(0)} \wedge dq^{(1)} + \int_{\Omega_2} \phi^{(0)} \wedge dq^{(1)} \\ [\phi, \mathcal{D}_i q] &= \left[\phi_1, \mathcal{D}_i^1 q_1\right] + \left[\phi_2, \mathcal{D}_i^2 q_2\right]. \end{aligned} \tag{8.3}$$

Here $\mathcal{D}_i$ is as described in App. C. For the boundary integral we get,

$$\begin{aligned}\int_{\partial\Omega} \phi^{(0)} \wedge q^{(1)} &= \int_{\partial\Omega_1} \phi^{(0)} \wedge q^{(1)} + \int_{\partial\Omega_2} \phi^{(0)} \wedge q^{(1)} \\ [\phi, \mathcal{D}_b q] &= \left[\phi_1, \mathcal{D}_b^1 q_1\right] + \left[\phi_2, \mathcal{D}_b^2 q_2\right].\end{aligned} \tag{8.4}$$

Here the matrices $\mathcal{D}_b^1$ and $\mathcal{D}_b^2$ are constructed as explained in App. C. The interesting thing now is what happens at the intersection of the two elements. So lets look at the boundary integral at the intersection,

$$\int_{\partial\Omega_1} \phi^{(0)} \wedge q^{(1)} + \int_{\partial\Omega_2} \phi^{(0)} \wedge q^{(1)} 0 \quad \text{, for} \qquad \partial\Omega_1 = \partial\Omega_2. \tag{8.5}$$

At the intersection both boundary integrals are of opposite sign, then using the expansions of $\phi^{(0)}$ and $q^{(1)}$ we can write (8.5) as,

$$\underbrace{\sum_{i=1}^{N} \left\{ \phi_{N+1,i} w_i^G \sum_{p=1}^{N+1} q_{N+1,p} e_p(\eta_i) \right\}}_{\int_{\partial\Omega_1} \phi^{(0)} \wedge q^{(1)}} + \underbrace{\sum_{i=1}^{N} \left\{ -\phi_{N+1,i} w_i^G \sum_{p=1}^{N+1} q_{N+1,p} e_p(\eta_i) \right\}}_{\int_{\partial\Omega_2} \phi^{(0)} \wedge q^{(1)}}. \tag{8.6}$$

And by reworking the summations this can be rewritten in

$$\begin{aligned}&\sum_{i=1}^{N} \left\{ \phi_{N+1,i} w_i^G \left[\sum_{p=1}^{N+1} q_{N+1,p} e_p(\eta_i) - \sum_{p=1}^{N+1} q_{N+1+1,p} e_p(\eta_i) \right] \right\} = \\ &\qquad \sum_{i=1}^{N} \left\{ \phi_{N+1,i} w_i^G \sum_{p=1}^{N+1} \left(\underbrace{q_{N+1,p} - q_{N+1+1,p}}_{=0} \right) e_p(\eta_i) \right\} = 0.\end{aligned} \tag{8.7}$$

The result that the summation of the boundary integrals over the element intersection should be zero results naturally from the fact that the flux over the boundary is equal on both sides of the intersection. So we see that the coupling of the two elements is done through the boundary integral. This coupling shows in the metric free divergence operator $\mathcal{D}$ simply as plus or minus unity. Our goal of reusing as much of the single element method as possible is thus accomplished. The matrices that form the system of equations are mostly a collection of the unchanged single element matrices. For the grid which consists of 2 elements of order $N = 2$ Fig .8.3 shows the nonzero elements of the matrices A,$\mathcal{D}$ and B. Note that matrix A shows no coupling, nor does B.

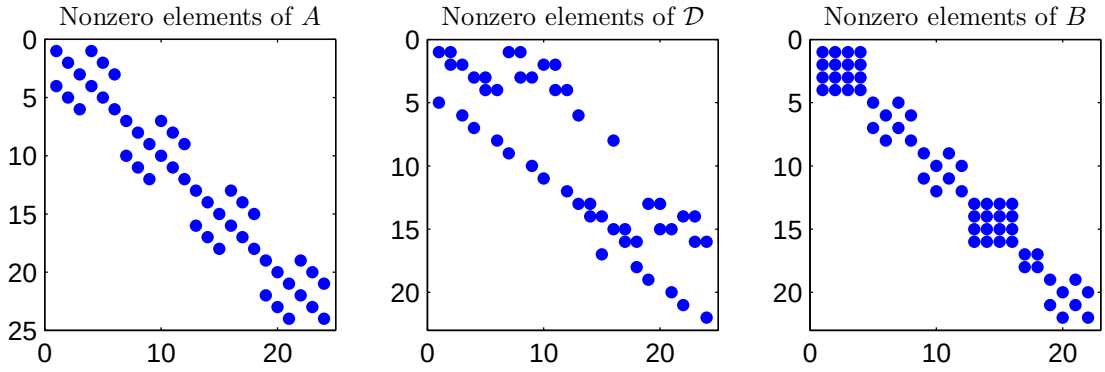


Figure 8.3: The mesh for discretization of ϕ

8.2 Test cases

Here we test 2 types of grid, the first is an undeformed logically rectangular grid. The second is a deformed logically rectangular grid. Both are a collection of $N_x \times N_y$ spectral elements of equal order N an example of these grids is given in Figures 8.4 and 8.5. Both elements have

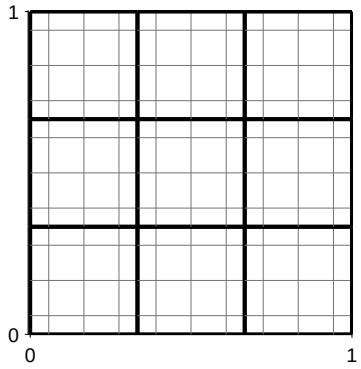


Figure 8.4: Example of multi element mesh ($Nhx = Nhy = 3$) with element order $N = 4$

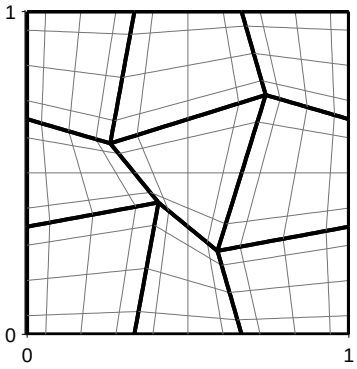


Figure 8.5: Example of deformed multi element mesh ($Nhx = Nhy = 3$) with element order $N = 4$

the size of a unit square, the undeformed grid is created by dividing a unit square in $N_{hx}N_{hy}$ evenly sized elements of order N. To create the deformed element we apply the coordinate transformation (8.8) to the corner points of each element. The deformation of the elements itself is then determined using a transfinite mapping. The deformation we chose is $c = 0.2$. The mapping (8.8) is a slight modification of the deformation we used earlier in the single element deformation (7.8).

$$\begin{aligned} x &= \frac{\xi + 1}{2} + c\sin(\pi(2\xi - 1)\cos(\pi(2\xi - 1)) \\ y &= \frac{\eta + 1}{2} + c\sin(\pi(2\xi - 1)\cos(\pi(2\xi - 1)). \end{aligned} \tag{8.8}$$

In this test case we want to find out if the convergence for increasing N, the p convergence, is still exponential (or spectral). Further we want to find out what the convergence rate is with respect to the element size $h \sim 1/N_x$ (h-convergence). This will no longer be exponential but will be algebraic convergence of the form $\epsilon \sim h^p$ of which we want to determine the order p.

8.2.1 Exponential convergence

To see whether we retain the exponential convergence when working with multiple elements, we test both the deformed and undeformed grid for a fixed number of characteristic grid sizes h ($N_x = N_y$) with the element order (N) increasing. The exact solution for the potential that we test for is given by,

$$\phi^{(0)}(x, y) = \sin(\pi x) \sin(\pi y). \tag{8.9}$$

The gradient of ϕ ($q^{(1)}$), and the forcing ($f^{(2)}$) follow naturally. Looking at Figures 8.6 and 8.7

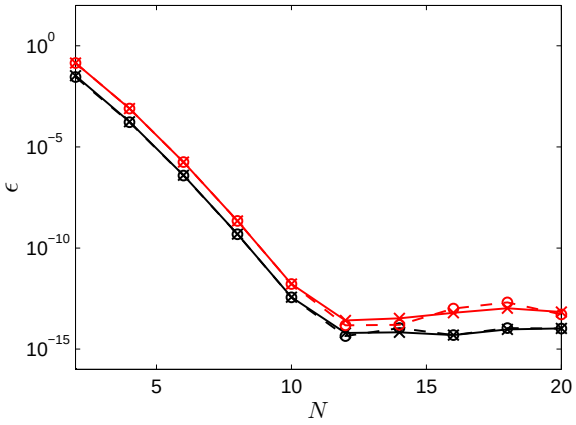


Figure 8.6: Exponential convergence for a fixed ($Nhx = Nhy = 3$) undeformed grid

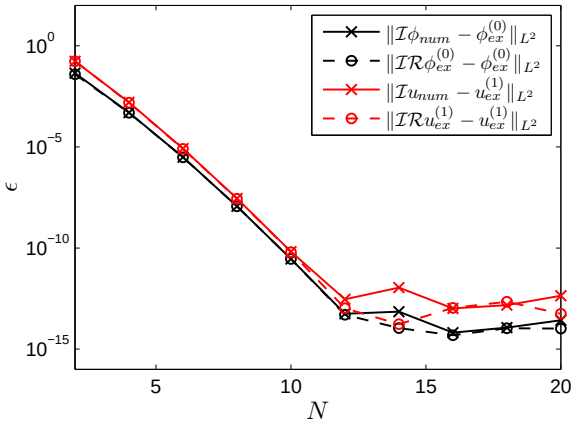


Figure 8.7: Exponential convergence for a fixed ($Nhx = Nhy = 3$) deformed grid $c = 0.2$

we see that the exponential convergence is indeed retained. As for the error in the conservation we can be very short. Also in this multi element case the relation div$q = f$ is satisfied exactly, to machine error that is. The deformed grid does show a lower convergence rate, than the undeformed grid. But the difference is not as profound as with the single element deformation. Mainly due to the fact that in this unit square not much is happening to ϕ, so even though the elements themselves are quite deformed, the solution in those elements is still very smooth and does not have much structure. Another thing is that the deformation is linear, unlike in the single element case.

8.2.2 h-convergence

Next we look at the h-convergence, for this we solve for ϕ on the unit square. This time we vary the characteristic grid size h or $1/N_x$ and keep the order of the elements (N) constant. We do this for multiple values of N. The h-convergence we are looking for is characterized by $\epsilon_{L2} \sim h^{\beta}$, so by decreasing the characteristic element size by a factor of two, the error in the L_2 norm decreases a factor of $1/2^{\beta}$. Now it is interesting to see how this order β relates to the element order N. Figure 8.8 shows the convergence in a double logarithmic plot for four

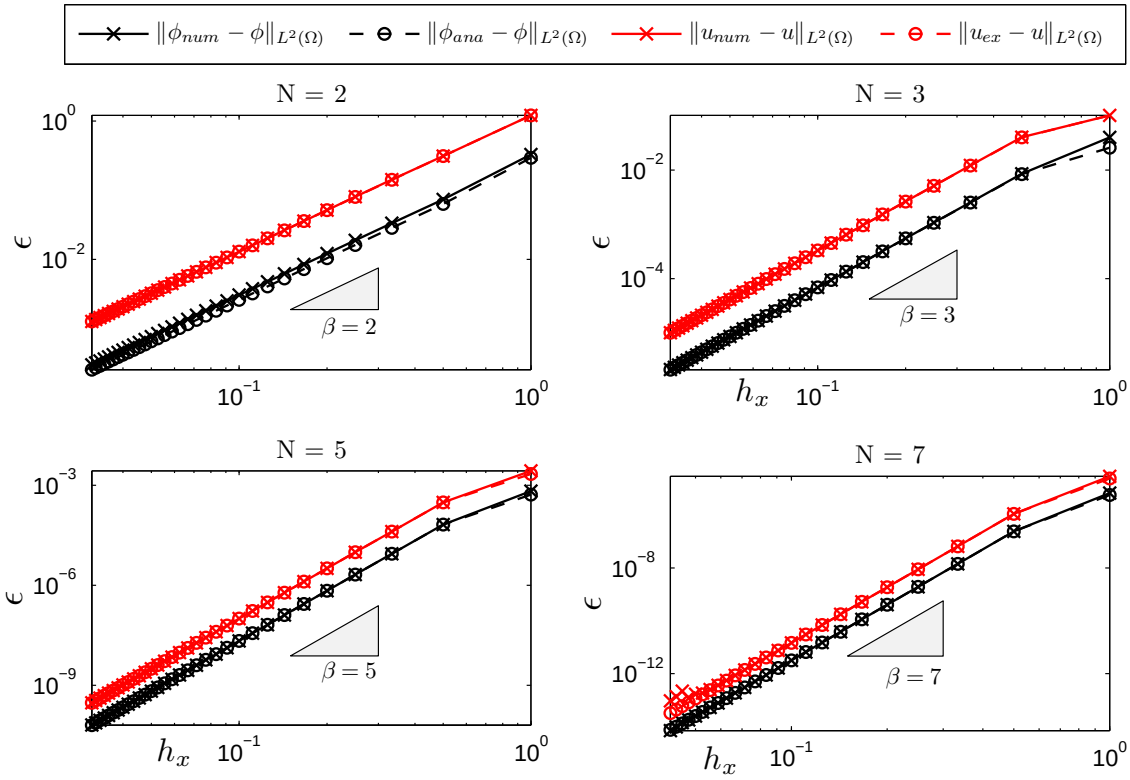


Figure 8.8: Logarithmic plot of error vs the grid size h_x

values of N, we also indicate the corresponding order β. From this follows quite nicely that, for a rectangular grid, the order with which the error goes down is quite consistently equal to N, so $\beta = N$. Lets call polynomial order of the element $\mathcal{P}$, then we have that $\mathcal{P} = N - 1$. So the polynomial order is one lower than the element order. The order of convergence β is equal to $\mathcal{P} + 1$. To test the h-convergence of the method when working with deformed grids we use the grid with the shape indicated by Fig 8.7. So for a given constant order N we decrease the element size. The convergence for this type is grid is given in Fig 8.9. The value of the convergence order β is calculated using data that shows a more or less constant order β, this is the part where h_x is smallest. The rates of convergence are slightly lower than those of the undeformed grid but are still in the order of $\mathcal{P} + 1$.

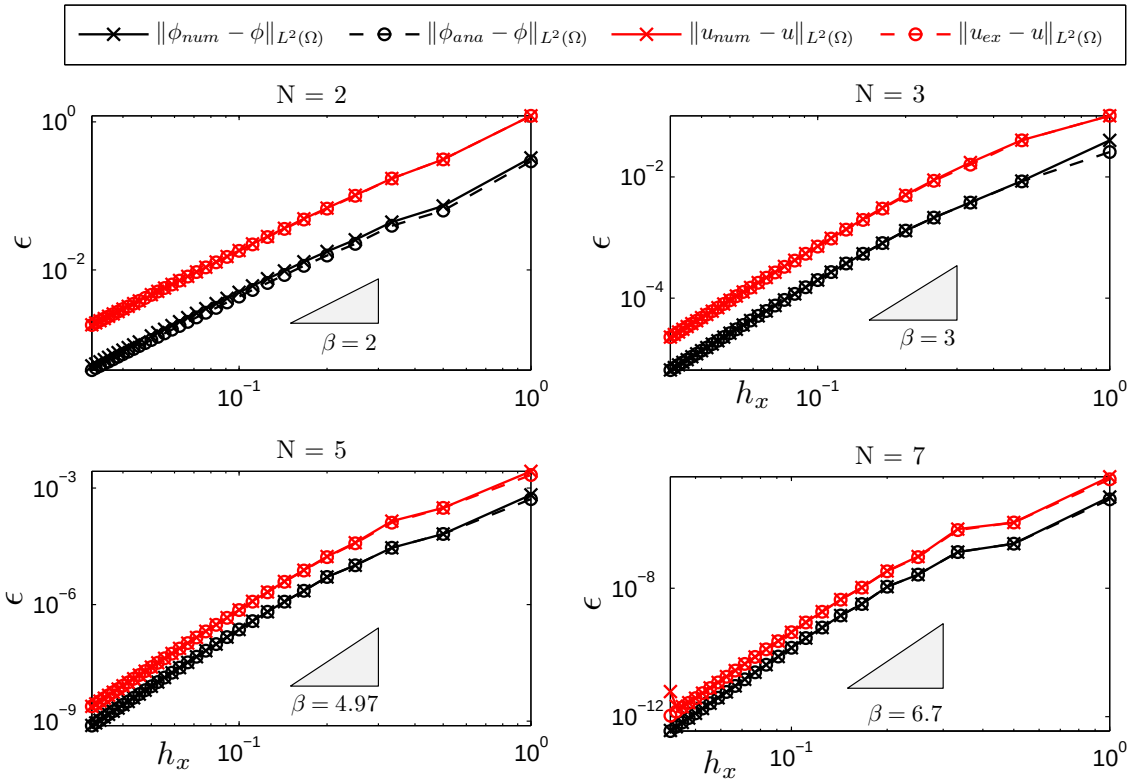


Figure 8.9: Logarithmic plot of error vs the grid size h_x for a deformed grid

8.3 Flow around cylinder

The following test problem is a bit more interesting from a physical perspective. As we will use the Poisson equation to calculate the potential flow around a cylinder. When we assume frictionless, irrotational, incompressible and stationary flow, we are left with just the equation of mass in the form.

$$\frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} = 0. \tag{8.10}$$

Where u_x and u_y are the velocities in Cartesian x and y directions. Now by introducing a velocity potential ϕ and saying that $u_x = \frac{\partial \phi}{\partial x}$ and $u_y = \frac{\partial \phi}{\partial y}$ we end up with the Poisson equation.

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = 0. \tag{8.11}$$

Which we have shown,we can write in terms of differential geometry

$$dq^{(n-1)} = 0, \quad q^{(n-1)} = \star d\phi^{(0)}. \tag{8.12}$$

The flow of a non rotating cylinder in an infinite domain contains a horizontal symmetry plane. This plane cuts though the cylinder. Because of this symmetry one only needs to

model half of the domain. Since the grid that we use to discretize the problem cannot be infinite, we model the problem on a $[W \times H] = [6 \times 3]$ domain. An example domain is shown in Fig 8.10, the radius (R) of the cylinder is 0.5. For this problem an exact solution exists,

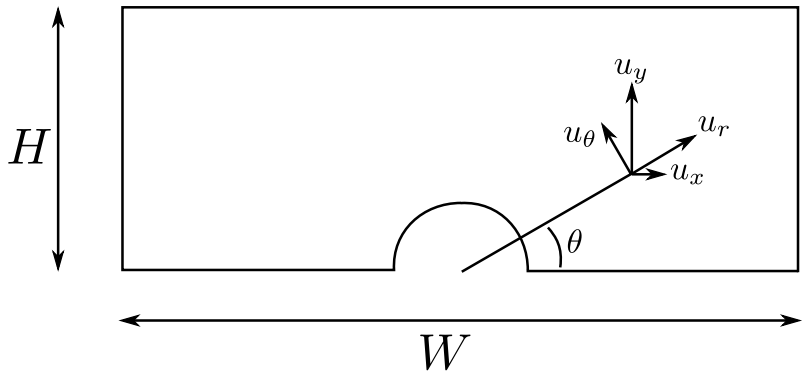


Figure 8.10: Grid on which the potential flow around cylinder is solved (N=3)

which means that we can impose exact boundary conditions for this problem. Therefore we do not have to introduce an error in trying to model the infinite domain. The exact solution for the radial and axial velocity in an infinite domain is given by,

$$\begin{aligned} v_r = &\quad U_\infty(1 - \frac{R^2}{r^2})\cos(\theta) \\ v_\theta = &\quad -U_\infty(1 + \frac{R^2}{r^2})\sin(\theta). \end{aligned} \tag{8.13}$$

Here U_∞ is the free stream velocity, r is the distance to the centre of the circle and θ is the angle starting at the right side of the semi circle (or 15:00) rotating counter clockwise. We can rewrite this to Cartesian velocity components,

$$\begin{aligned} u_x &= v_\theta \sin(\theta) - v_r \cos(\theta) \\ u_y &= v_\theta \cos(\theta) + v_r \sin(\theta). \end{aligned} \tag{8.14}$$

With $x = r\cos(\theta)$ and $y = r\sin(\theta)$ we can calculate the Cartesian velocity components in the whole domain. These equations hold for the cylinder placed in a domain without boundaries (infinite).

The grid on which the cylinder problem is solved is depicted in Fig 8.11. The Neumann boundary conditions that are enforced are calculated using the known exact values, which involves discrete integration using Gaussian quadrature. As for the grid depicted in the figure, it is a logically rectangular grid of which the boundary is deformed such that way obtain the semi circle. This transformation is done using the transfinite mapping which is discussed in Gordon and Hall (1973). For accuracy it might be better though to use local mesh refinement near the stagnation points of the cylinder. However this comes along with a lot of book keeping, and being a proof of concept rather than a study of potential flow we stick to this logically rectangular grid. In the convergence study we keep the amount of cells and increase the element order N. Fig 8.12 shows the results, which are the error of the u_{num} and the divergence Dq. It is evident that the solution is completely divergence free and that it converges to the exact solution exponentially fast.

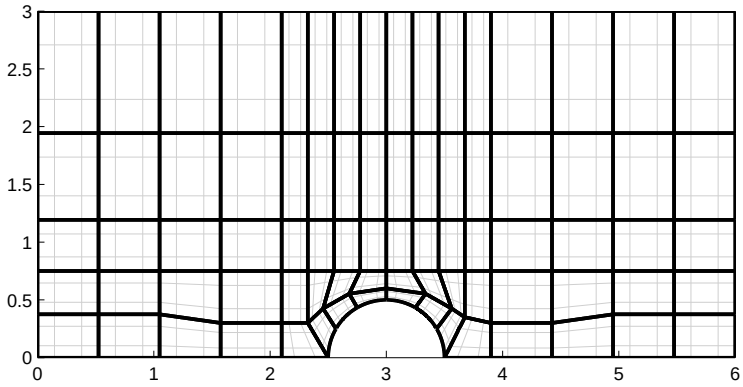


Figure 8.11: Grid on which the potential flow around cylinder is solved (N=3)

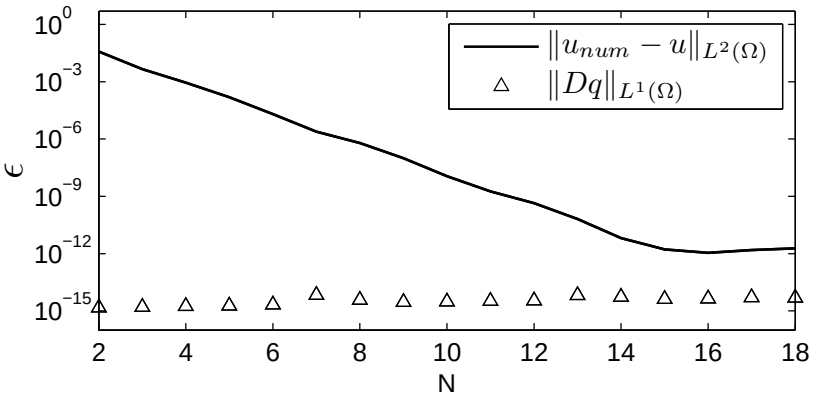


Figure 8.12: Convergence for the cylinder potential problem

For this problem we van calculate the potential deficiency which is given by $\phi_{num} - U_\infty x$. From Fig 8.13 we can see that the characteristic shape of the potential deficiency is captured accurately for the linear elements ($N = 2$) and hardly changes shape when N increases.

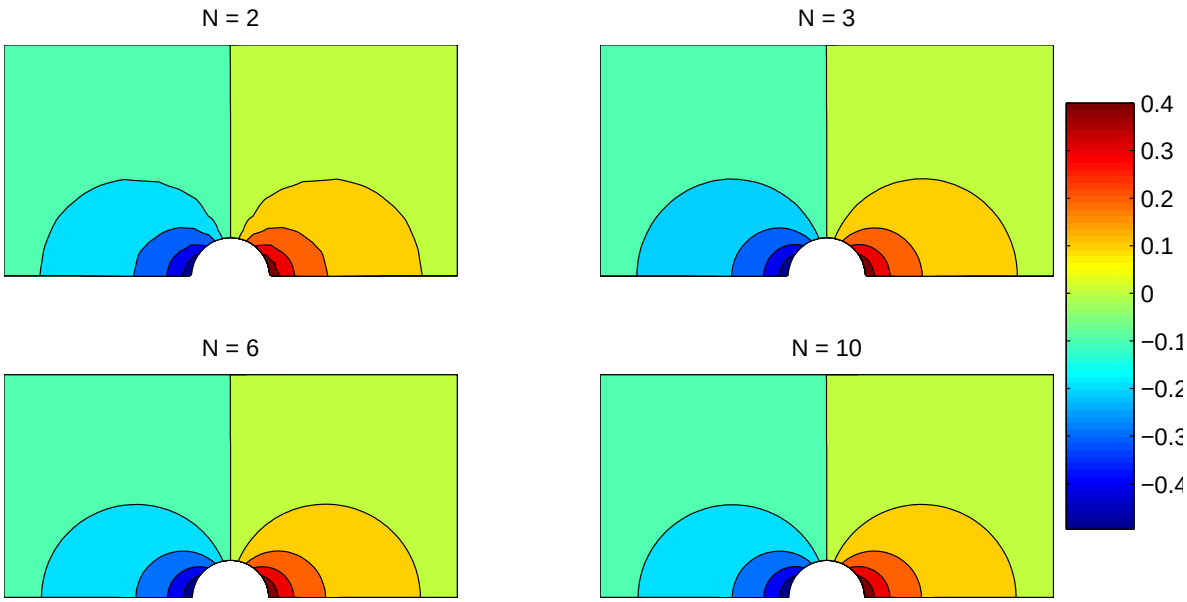


Figure 8.13: Potential deficiency with respect to undisturbed flow

Chapter 9

Anisotropic diffusion

9.1 Symmetric tensor $\mathbb{K}$

We have shown that by using the divergence operator as a support for the gradient operator we get a very robust discretization. When working with transformed grids the inner product upon which the method is based changes slightly. Along the same line we can extend the method such that we can include a symmetric positive definite tensor which allows us solve the slightly broader class anisotropic diffusion problems.

9.1.1 Introducing $\mathbb{K}$

Consider the system,

$$u^{(1)} = d\phi^{(0)} \ , \quad dq^{(1)} = f^{(n)} \ , \quad q^{(n-1)} = \star\mathbb{K}u^{(1)}. \tag{9.1}$$

Here $\mathbb{K}$ is a symmetric tensor of the form $\left(\begin{smallmatrix} k_{xx} & k_{xy} \\ k_{xy} & k_{yy} \end{smallmatrix}\right)$. One of the problems that can be solved with this system is the stationary anisotropic heat equation and the pressure (and gravity) driven Darcy flow in porous media (Neuman (1977)). In Darcy flow, $\mathbb{K}$ is called the anisotropic permeability tensor and represents the material's preferred direction of flow when subject to a pressure gradient ($d\phi^{(0)}$). One could argue that since we have a good representation of **div** and **grad** we could simply insert the tensor $\mathbb{K}$. However it is not that simple. One reason for this is that the **grad** operator that we construct, takes values on points and maps these onto the space of lines. And we cannot apply a tensor, which normally acts on point values, to lines. Mathematically this makes no sense. But as proposed in Hyman (1997), $\mathbb{K}$ can be introduced by changing the inner product which is the basis of the derivation of the method in the earlier chapters.

9.1.2 Flux operator

For solution purposes we also want the discretization of (9.1) to be symmetric on arbitrarily shaped domains. And by including the tensor $\mathbb{K}$ in the gradient operator we can in fact do so. So lets define $\mathcal{G}\phi = \star\mathbb{K}d\phi^{(0)}$ where $\mathcal{G}$ is the new discrete anisotropic gradient matrix operating on the values of ϕ located on the EG mesh. A better name for this matrix is the flux operator. For the sake of clarity we use the standard undeformed GL element to show how to find the mimetic discretization of $\mathcal{G}$. We include the inverse of the symmetric positive definite tensor $\mathbb{K}$, $\mathbb{K}^{-1}$, in the definition of the Hodge.

$$\begin{aligned}\int_\Omega (q^{(1)}, \star\mathbb{K}\star^{-1}\mathcal{G}\phi)\omega^n &= \int_\Omega (q^{(1)}, \star\underbrace{\mathbb{K}^{-1}\star^{-1}\star\mathbb{K}}_{1} d\phi^{(0)})\omega^n \\ \int_\Omega (q^{(1)}, \star d\phi^{(0)})\omega^n &= \int_\Omega q^{(1)}\wedge d\phi^{(0)}.\end{aligned} \tag{9.2}$$

So in the inner product we pre-multiply the $\mathcal{G}$ with the inverse of the tensor $\mathbb{K}$. This might be confusing, but since we defined the gradient to already include $\mathbb{K}$, these two cancel each other out and we are back at the starting point of the derivation we have made earlier (4.2c). Here we use an undeformed element but this can be naturally extended to arbitrarily shaped curvilinear grids. In that case the integral looks like

$$\int_{\Omega'} (\star\mathbb{K}^{-1}\star^{-1}\Phi^{\star^{-1}}\mathcal{G}\phi, \Phi^{\star^{-1}}q^{(1)})\Phi^\star\omega^n = \int_{\partial\Omega}\phi^{(0)}\wedge q^{(1)} - \int_\Omega \phi^{(0)}\wedge dq^{(1)}. \tag{9.3}$$

In App. F we show how to evaluate the inner product using summations. But again for clarity we prefer to use the matrix notation. It shows also that including the tensor $\mathbb{K}$ is a rather small modification. Written out in terms of matrices (9.3) equals

$$(\mathbf{T}^{-1}\mathcal{I}^{GL}\tilde{\mathbf{q}})^T W^{GL}\mathbf{J}\mathbf{K}^{-1}(\mathbf{T}^{-1}\mathcal{I}^{GL}\mathcal{G}\vec{\phi}). \tag{9.4}$$

Here we see that the only difference to (6.5) is the matrix $\mathbf{K}^{-1}$. So even though formally we had to introduce the Hodge $\star$ operator and its inverse, these do not appear in the numerical evaluation (9.3) of the inner product. Lets restate what (9.4) means. Operating on the vector $\star\mathcal{G}\vec{\phi}$, we have the interpolation matrix $\mathcal{I}^{GL}$. This interpolation matrix takes all the edge values (cochains) contained in that vector and returns (reconstructs) the values $\binom{\star\mathcal{G}_\xi\phi^{(0)}}{\star\mathcal{G}_\eta\phi^{(0)}}$, at each of the Gauss-Lobatto points. The transformation matrix $\mathbf{T}^{-1}$ operating on $\binom{\star\mathcal{G}_\xi\phi^{(0)}}{\star\mathcal{G}_\eta\phi^{(0)}}$ returns $\binom{\mathcal{G}_x\phi^{(0)}}{\mathcal{G}_y\phi^{(0)}}$. Notice that the Hodge $\star$ operators are no longer there. This means that we can just multiply with the matrix $\mathbb{K}^{-1}$. The complete set of matrices that make up the gradient operator is given by,

$$\mathcal{G} = \mathbf{A}^{-1}\mathcal{D}^T\mathbf{B} \tag{9.5}$$

$$\mathbf{A} = \mathcal{I}^{GL^T}\mathbf{T}^{-T}W^{GL}\mathbf{J}\mathbf{K}^{-1}\mathbf{T}^{-1}\mathcal{I}^{GL}, \quad \mathbf{B} = \mathcal{I}^{G^T}W^G. \tag{9.6}$$

9.2 Anisotropic diffusion

The example that we discuss here is a problem with *anisotropic* diffusion, meaning that the symmetric tensor $\mathbb{K}$ varies throughout the domain of computation. The following problem is from Lipnikov and Gyra (2008), the anisotropic in the domain [0,1] is given by,

$$\mathbb{K}(x,y) = \begin{cases} \alpha = 10^{-3}, & for \;\; x < 0.5 \\ \alpha = 1, & for \;\; x > 0.5. \end{cases} \tag{9.7}$$

which has as exact solution,

$$\phi(x,y) = \begin{cases} y(y-1)x^2 & for \;\; x < 0.5 \\ y(y-1)(1-x)(\alpha - x(1+2\alpha)), & for \;\; x > 0.5. \end{cases} \tag{9.8}$$

Fig 9.1 shows the potential function defined by (9.8). Since the solution is a collection of two polynomials, the use of two second order elements is sufficient to get an exact solution. This is only when the two elements together span the whole domain, share the boundary at $x = 0.5$ and are not internally deformed. It is interesting to see then what happens when using multiple deformed elements that do not have a boundary at $x = 0.5$. This means the value of tensor $\mathbb{K}$ jumps within an element. Discontinuous functions however are not well suited for integration using Gaussian quadrature nor for polynomial expansion.

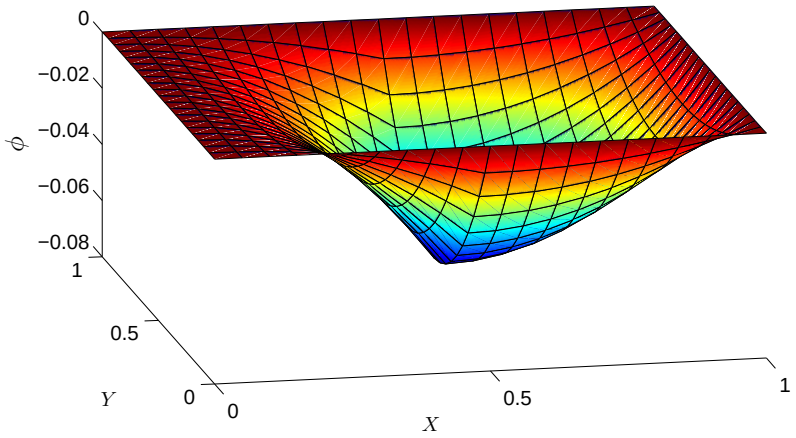


Figure 9.1: Potential defined in (9.8)

9.2.1 Discontinuous $\mathbb{K}$ in element

To see what effect a discontinuous tensor $\mathbb{K}$ has when this discontinuity does not lie on the boundary of an element, we use the grid indicated by Fig 8.5. Fig 9.2 shows the contour plots of the analytic solution and three solutions for the problem of (9.7). Clearly the numerical solutions show a very distorted and discontinuous potential whereas the exact potential shows

a slight kink in the derivative at $x = 0.5$. The fact that the discontinuity of $\mathbb{K}$ runs through the elements in the centre of the domain causes the solution to be highly oscillatory in the elements that this discontinuity runs through. The result of this is that the exponential convergence is lost as we can see from Fig 9.3 which is a double logarithmic plot of the error of the solution. That fact that convergence shows as a straight line in this double logarithmic plot indicates that the convergence is algebraic.

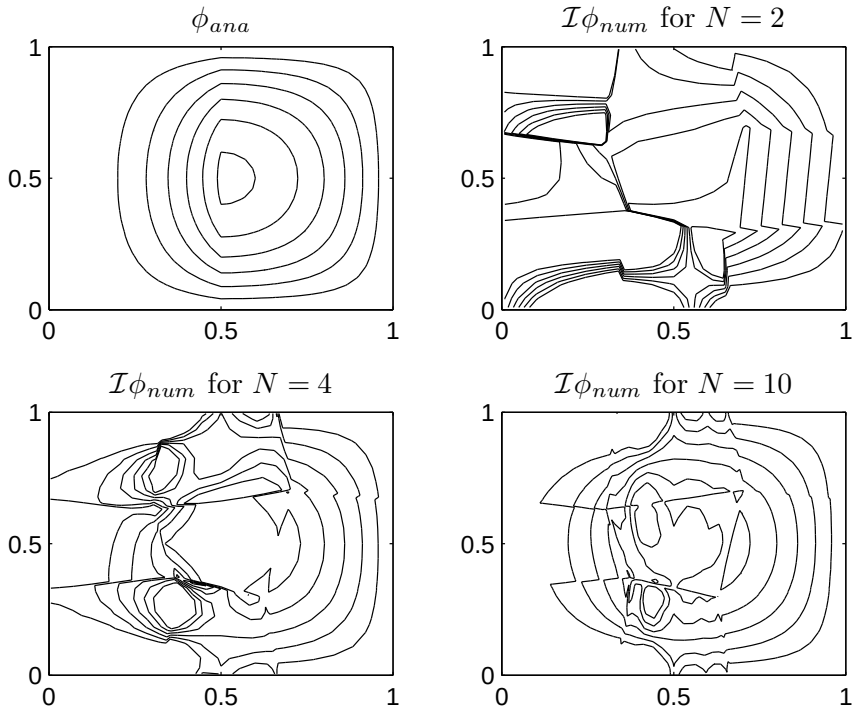


Figure 9.2: Potential contour plots for $\mathbb{K}$ discontinuity through element

9.2.2 Discontinuous $\mathbb{K}$ on boundary

Where in the previous example the discontinuity was positioned inside the elements we use in this example a grid that has element boundaries along $x = 0.5$. The grid that we use is depicted in Fig 9.4, it is a distorted mesh but the discontinuity at $x = 0.5$ is on the boundary of the elements. Fig 9.5 shows the contour plots of ϕ and it shows that the solutions for each N agree quite well with the exact solution. None of the oscillatory behaviour that is present in the example where the discontinuity lies within an element. This shows also in the convergence plot in Fig 9.6. From order three the solution for $u^{(1)}$ becomes exact and from order four that of $\phi^{(0)}$. This is because the mapping is a first order polynomial and the solution itself is a second or first order polynomial as well. Together this makes that the solution is exact from a very low order on.

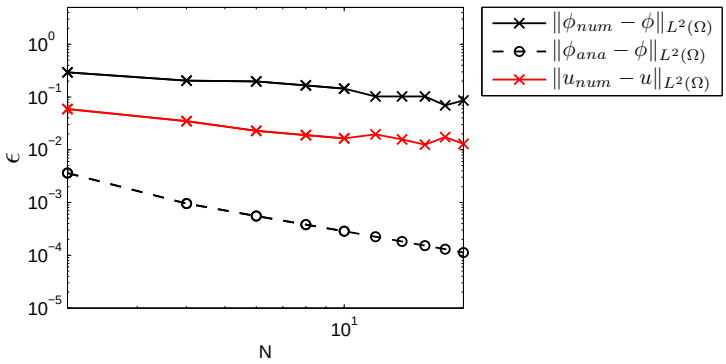


Figure 9.3: Double logaritmic plot of error verses element order N for a 3×3 distorted grid

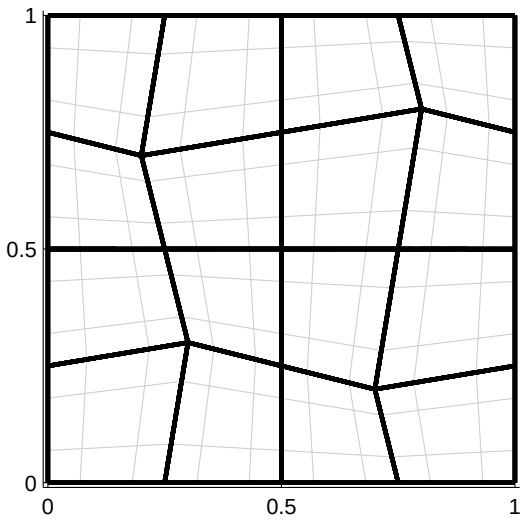


Figure 9.4: 4×4 grid used to test discontinuous $\mathbb{K}$ on boundary of element

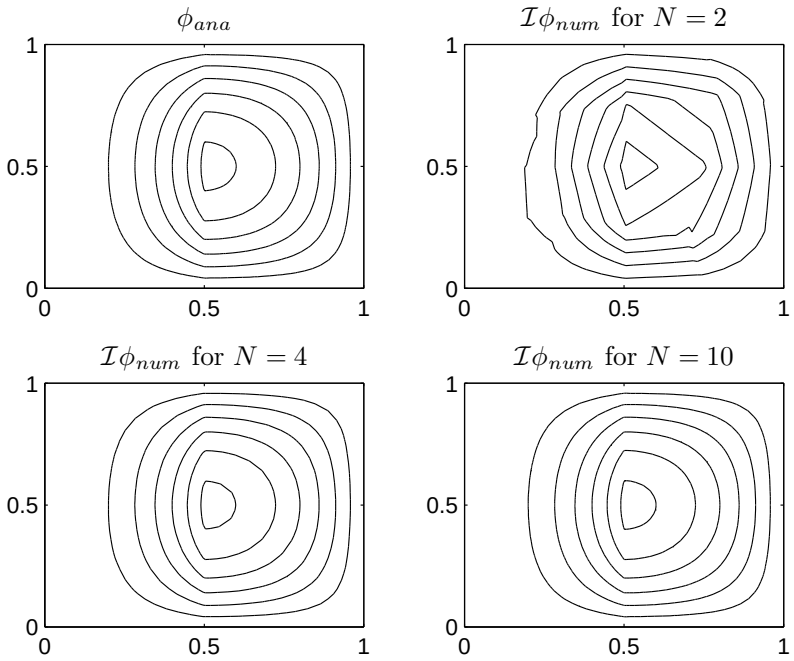


Figure 9.5: Potential contour plots for $\mathbb{K}$ discontinuity through element

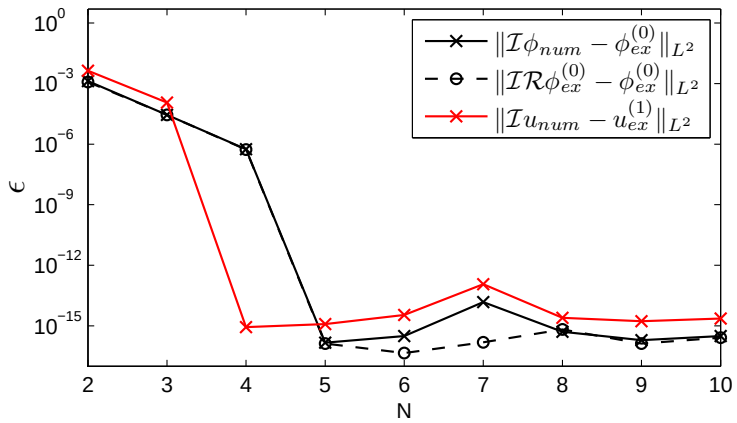


Figure 9.6: Semi logaritmic plot of error v.s. element order N for a 4×4 distorted grid

Chapter 10

Conclusions and recommendations

In the first chapters of this report we focussed on how the Poisson equation can be rewritten in term of differential geometry. By using differential geometry and the link to its discrete counterpart Algebraic topology we derived a mimetic discretization of the Poisson equation. This means that the discrete operators (**div** and **grad**) that make up the Poisson equation mimic discretely the analytical relations that connect them.

10.1 Conclusions

In chapters 5 and 7 we tested the resulting system of equations from which the following conclusions can be drawn:

- In creating a mimetic formulation we had to create an element that consists of two separate staggered elements. One standard Legendre-Gauss-Lobatto element and one Extended-Gauss grid. This staggered configuration enables us to incorporate the boundary conditions in the mimetic formulation, see Section 4.6
- For both deformed and undeformed elements the discretization results in spectral convergence that is theoretically to be expected. Although for severely deformed self overlapping elements the convergence is no longer equal to the theoretically expected convergence, see Sections 5.3 and 7.3
- Because of the introduction of the edge functions, both the deformed and undeformed element's solution show absolute conservation of the conserved quantity, see Sections, see Sections 5.3 and 7.3
- A low order finite volume operator L^{FV} serves as a good preconditioner for the Spectral Element Method operator L^{SEM}, this results in a condition number of $O(1)$ which is independent of the element order, see Section 4.6

In chapters 8 and 9 we extend the method to multiple elements and a more general elliptical equation, the anisotropic diffusion problem. From these chapters we can conclude the following:

- By equating flux quantities at the boundaries of elements the multiple element case also shows absolute conservation for both deformed and undeformed elements, see Section 8.2.1.
- The so called h-convergence for both deformed and undeformed elements is what is to be theoretically expected, see Section 8.2.2
- By constructing a flux operator that incorporates the anisotropic diffusion tensor we have the same converging and conservation for anisotropic diffusion problems as for the Poisson equation, see Section 9.2.2

10.2 Recommendations

The mimetic method shows that we can combine high order, curved grid and absolute conservation in one. It might be interesting to look into the following points.

- Conduct research in use of non conforming grids such that local increases in element order can be used where the solution demands for it.
- The method might be suitable to calculate solutions for the full potential equation.
- For the grid we make use of the Legendre-Gauss-Lobatto points which are non analytical. The use of the Chebyshev-nodes might give more accurate results.
- Conduct research in finding a mimetic formulation for the Stokes equation.

Bibliography

P.B. Bochev. A discourse on variational and geometric aspects of stability of discretizations. 33rd Computational Fluid Dynamics Lecture Series, 2003.

P.B. Bochev and J.M. Hyman. Principles of mimetic discretizations of differential equations. IMA Volume 142, Springer Verlag, 2006.

H. Flanders. *Differential forms with applications to the physical sciences*. Dover publications, inc. New York, 1962.

T. Frankel. *The Geometry of Physics an introduction*. Cambridge university press, 2004.

M.I. Gerritsma. Edge functions for spectral elements. proceedings ICOSAHOM, 2009.

W. J. Gordon and C. A. Hall. Construction of curvilinear co-ordinate systems and their applications to mesh generation. *Int. J. Numer. Meth. Eng.*, 1973.

J. Hyman. The numerical solution of diffusion problems in strongly heterogeneous non-isotropic materials. Journal of Computational Physics 132, 130148, 1997, 1997.

K. Lipnikov and V. Gyra. High-order mimetic finite difference method for diffusion problems on polygonal meshes. Elsevier, 2008.

C. Mattiussi. The Finite Volume, Finite Difference, and Finite Elements Methods as Numerical Methods for Physical Field Problems. *Advances in Imaging and Electron Physics*, 113, 2000.

S. P.: Neuman. Theoretical derivation of Darcy's law. *Acta Mechanica*, Volume 25(3-4), 1977.

S.J. Sherwin and G.M.: Karniadakis. *Spectral hp element methods for cfd*. Oxford university press, 1999.

E. Tonti. On the mathematical structure of a large class of physical theories. Accademia Nazionale dei Lincei, estratto dai Rendiconti della Classe di Scienze fisiche, matematiche e naturali, Serie VIII, Vol. LII, fasc. 1, Gennaio, 1972., 1972.

Appendix A

Lagrangian polynomials

The unique Lagrangian polynomial of order N with zeros at x_p $(0 \leq p \leq N+1)$ is given by (A.1).

$$h_p(x) = \frac{\prod_{q=0,q\neq p}^{N}(x - x_q)}{\prod_{q=0,q\neq p}^{N}(x_p - x_q)}. \tag{A.1}$$

Though if we denote by $g(x)$ the polynomial of order $(N+1)$ with zeros at the $(N+1)$ nodal points x_p, then $h_p(x)$ can be written in a more compact form A.2

$$h_p(x) = \frac{g(x)}{g'(x_p)(x - x_p)}. \tag{A.2}$$

So we see that when we have a polynomial that has its zeros on points x_p we can easily construct functions that have a unit value on one node and the value zero on all the others. The Lagrangian functions that are treated in this appendix are used as basis functions to expand other mostly unknown functions making use of relation (A.3)

$$u(x) = \sum_{p=0}^{N} \hat{u}_p h_p(x), \quad \hat{u}_p = u(x_p). \tag{A.3}$$

Because of the orthogonality of the basis functions, the coefficients with which we multiply the basis functions are equal to the function values at the nodes. This means that the approximation is exact in the nodal points. For polynomials up to order $N-1$ where N is the number of nodes we have that this expansion is exact. For other non polynomial smooth and differentiable functions the error that we make tends to go down exponentially with increasing order N. Another property that makes these Lagrangian functions suitable to be basis functions, is that when we define weights w_p to be integral of the basis function $h_p(x)$ (A.4), we can use these weights to approximate the integral of any function that we are using the expansion for (A.5).

$$w_p = \int_{-1}^{1} h_p(x)dx, \quad 0 \leq p \leq N \tag{A.4}$$

$$\int_{-1}^{1} u(x)dx = \sum_{p=0}^{N} w_p u(x_p). \tag{A.5}$$

The integration accuracy differs per type of Lagrangian, we will see that Gauss integration is has a higher accuracy than the Gauss-Lobatto integration.

A.1 Legendre polynomials

A particular class of polynomials that can be used as a basis for constructing Lagrangians are the Legendre polynomials. The Legendre polynomials $L_k(x), k = 0, 1, ...,$ are the eigenfunctions of the singular Sturm-Liouville problem (A.6)

$$((1 - x^2)L_k'(x))' + k(k + 1)L_k(x) = 0. \tag{A.6}$$

When $L_k(x)$ is normalised such that $L_k(1) = 1$ the Legendre polynomials satisfy the recurrence relation (A.7), where $L_0(x) = 1$ and $L_1(x) = x$.

$$L_{k+1}(x) = \frac{2k + 1}{k + 1} x L_k(x) - \frac{k}{k + 1} L_{k-1}(x). \tag{A.7}$$

Amongst other reasons it is this recurrence relation that makes this particular class of polynomials interesting to work with in spectral element methods. They allow for efficient calculation of the Legendre polynomials.

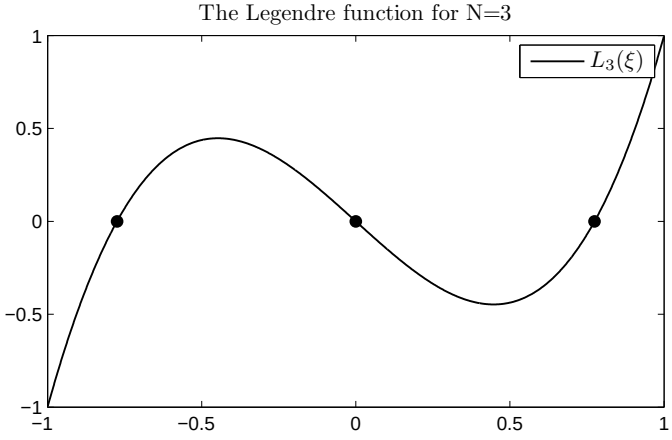


Figure A.1: The shape of the Legendre function for $N = 3$

A.2 Gauss-Lobatto

One typical case of a Lagrangian polynomial is the one with zeros on the Gauss-Lobatto nodes. These nodes are the zeros of the function $g(x) = (1 - x^2)L_N'(x)$ in which $L_N(x)$ is

the N-th order Legendre polynomial. Now in order to use this function as a base to form the Lagrangian using equation (A.2) we need to calculate the zeros of $g(x)$ and to calculate $(1-x^2)L'_N(x)$. Fortunately we do not have to perform the actual differentiation of $L_N(x)$, due to relation (A.8) which allows us to rewrite $g(x)$ as a functions of the easy to calculate Legendre polynomials only.

$$(1-x^2)L'_N(x) = NL_{N-1}(x) - NxL_N(x). \tag{A.8}$$

In order to calculate the zeros of $g(x)$ we can use the iterative Newton-Raphson method in which equation (A.9) is used to calculate the new approximate zero locations till the accuracy that is desired.

$$x_{n+1} = x_n - \frac{g(x_n)}{g'(x_n)}. \tag{A.9}$$

In order for such an iterative method to work we need a good estimate for the initial zero locations, for this one can use the GL-Chebyshev nodes which are located at (A.10)

$$x_k = \cos\left(\frac{\pi k}{N}\right), \quad k = 0, 1..., N. \tag{A.10}$$

Due to the definition of the Legendre polynomials (A.6) we can write $g'(x) = -N(N+1)L_N(x)$. Combining the relations (A.2),(A.8) and the above relation for $g'(x)$ we can write the Gauss-Lobatto Lagrangian polynomials as A.11.

$$h_p(x) = \frac{NL_{N-1}(x) - NxL_N(x)}{(-N(N+1)L_N(x))(x-x_p)}. \tag{A.11}$$

Figure (A.2) shows the locations of the Gauss-Lobatto points and the form of the associated Lagrangian polynomials for $N = 4$.

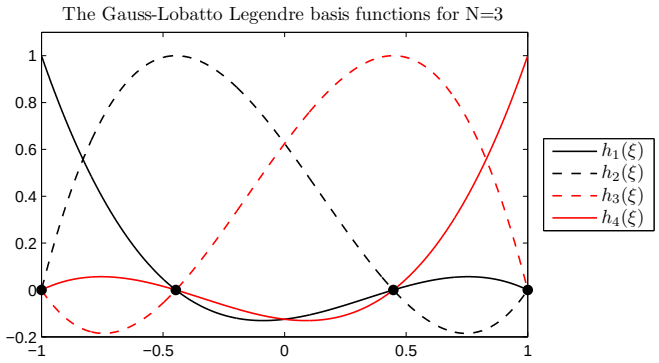


Figure A.2: The shape of the GL Lagrangians for $N = 3$

The weight corresponding to each basis function can be calculated using (A.12)

$$w_p = \frac{2}{N(N+1)}\frac{1}{[L_N(x_p)]^2}. \tag{A.12}$$

When the polynomial order of a Gauss-Lobatto Lagrangian is given by N, the Gauss-Lobatto integration is exact for polynomials up to $2N-1$.

A.3 Gauss

The second type of Lagrangian that we make use of, is the Lagrangian which has as zeros the Gauss points, see figure A.1l. These points are given to be the zeros of L_N. The first thing we need to do in order to construct the Gauss Lagrangian is to find the zeros of $g(x) = L_N$ for which we can again use (A.9). Now $g(x)$ can be calculated using the recurrence relation (A.7) and $g'(x)$ can be found when recognizing that (A.6) can be rewritten into (A.13)

$$L'_N(x) = \frac{NL_{N-1}(x) - NxL_N(x)}{(1-x^2)}. \tag{A.13}$$

This time suitable starting values for x_n are given by.

$$x_p = cos\left(\frac{2p+1}{2N+2}\right), \quad 0 \leq p \leq N \tag{A.14}$$

Using the information above, we can formulate the expression for the Lagrange polynomial with zeros on the Gauss nodes (A.15).

$$\tilde{h}_p(x) = \frac{L_N(x)}{(x-\tilde{x}_p)\frac{NL_{N-1}(\tilde{x}_p)}{1-\tilde{x}_p^2}}. \tag{A.15}$$

The weights of the function are given by (A.16)

$$w_p = \frac{2}{(1-x_p^2)[L'_N(x_p)]^2}. \tag{A.16}$$

When the polynomial order of a Gauss Lagrangian is given by N, the Gauss integration is exact for polynomials up to $2N+1$.

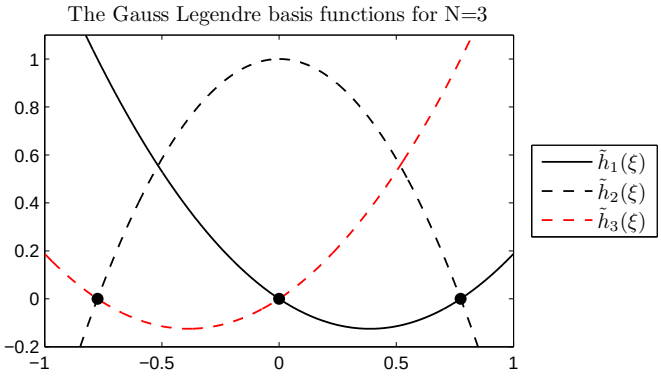


Figure A.3: The shape of the GL Lagrangians for $N = 3$

A.4 Extended Gauss

Another type of polynomial that we are using are the Extended Gauss (EG) polynomials. These polynomials are slightly modified Gauss polynomials to include also the endpoints of

the domain, -1 and 1. The introduction of these endpoints is needed for our derivation of the discrete symmetric Laplace operator. The EG polynomials can be constructed using the Gauss polynomials and extending them in the following manner (A.17).

$$\begin{aligned} \overline{h}_0(\xi) &= \tilde{h}_1(\xi)\cdot(1-\xi)\cdot(\xi-\xi_1)\frac{1}{\tilde{h}_1(-1)\cdot 2\cdot(-1-\xi_1)} \\ \overline{h}_i(\xi) &= \tilde{h}_i\cdot\frac{1-\xi^2}{1-\xi_i^2} \\ \overline{h}_0(\xi) &= \tilde{h}_N(\xi)\cdot(1+\xi)\cdot(\xi-\xi_1)\frac{1}{\tilde{h}_N(1)\cdot 2\cdot(1-\xi_1)}. \end{aligned} \tag{A.17}$$

As can be seen, they are mainly an expansion of the original Gauss elements or functions. For the most outer and inner Extended Gauss function, the function that is nearest are used. This means that for the construction of $\overline{h}_0(\xi)$ we use the function $\tilde{h}_1(\xi)$. Though we could use any other of the Gauss functions, since these type of polynomials are all Lagrangians, and as we have said earlier the Lagrangian is unique. The form of the polynomial is dictated by the location of its zeros, and therefore it does not matter which of the Gauss functions we take, the result is the same. By introducing new functions we might ask ourselves what the integration weights are for this particular form. The definition of our new weights is given by (A.18). Here w_{b_j} is the j-th Bouman weight and w_{g_i} the i-th Gauss weight. ξ_{g_i} is the i-th Gauss point.

$$\int_{-1}^{1}\overline{h}_j(\xi)d\xi = w_{b_j} = \sum_{i=1}^{N} w_{g_i}\overline{h}_j(\xi_{g_i}) \quad for \quad N\geq 3, \quad j = 1..N+1. \tag{A.18}$$

Looking a bit closer at (A.18) and figure (A.4), we see that the weight of $\overline{h}_0$ must be zero. Since the function $\overline{h}_0$ is per definition zero in the Gauss points its contribution to the line integral (its weight) is always zero, the same holds for $\overline{h}_{N+1}$. The weights of the inner points remain exactly the Gauss weights. Some tests where done to see whether this is also true when a higher order polynomial is used for the integration. And indeed the weights do not change, as they shouldn't.

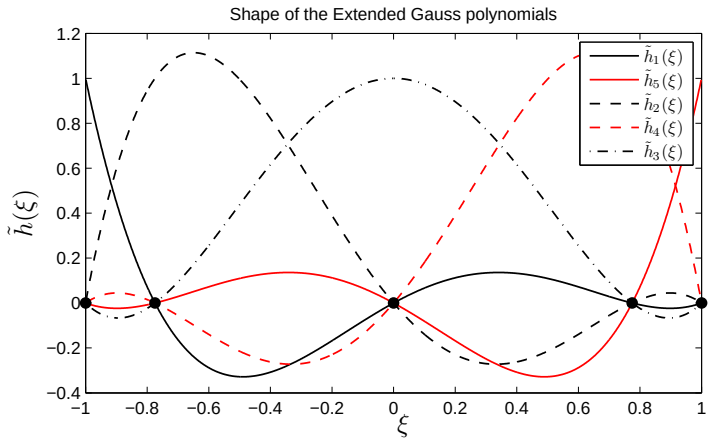


Figure A.4: The shape of the EG Lagrangians for $N = 3$

A.5 Gauss-Lobatto-Legendre edge function

In the derivation of our numerical approximation of the external derivative we make a statement that with a recombination one can write

$$\sum_{i=0}^{N}\sum_{j=1}^{N} q^{\xi}_{i,j} dh_i(\xi)e_j(\eta) = \sum_{i=1}^{N}\sum_{j=1}^{N}\left(q^{\xi}_{i,j} - q^{\xi}_{i-1,j}\right) e_i(\xi)e_j(\eta) \tag{A.19}$$

Lets prove this for a small N, say 2. Here we can exclude the summation over j by writing,

$$\sum_{i=0}^{N}\sum_{j=1}^{N} q^{\xi}_{i,j} dh_i(\xi)e_j(\eta) = \sum_{i=0}^{N} a_i dh_i(\xi), \quad a_i = \sum_{j=1}^{N} q^{\xi}_{i,j} e_j(\eta). \tag{A.20}$$

Expanding this sum for $N = 2$

$$\sum_{i=1}^{2} a_i dh_i(\xi) = a_0 dh_0(\xi) + a_1 dh_1(\xi) + a_2 dh_2(\xi). \tag{A.21}$$

Now we can do the same for the right hand side of (A.19) using the definition of the edge interpolants $e_i(\xi) = -\sum_{k=0}^{i} \frac{dh_i(\xi)}{d\xi}$. We also use the fact that the sum of all the basis functions is equal to 1, and as a result the sum of the derivatives of the basis functions equals 0, $\sum_{i=0}^{N} h_i(\xi) = 1$ thus $\sum_{i=0}^{N} \frac{dh_i(\xi)}{d\xi} = \frac{d1}{d\eta} = 0$

$$\begin{aligned}
\sum_{i=1}^{2} (a_i - a_{i-1})\, e_i(\xi) &= -(a_1 - a_0)dh_0(\xi) - (a_2 - a_1)(dh_1(\xi) + dh_0(\xi)) \qquad (A.22)\\
&= a_0 dh_0(\xi) \underbrace{-a_1 dh_0(\xi) + a_1 dh_0(\xi)}_{=0} + a_1 dh_1(\xi) - a_2 \underbrace{(dh_0(\xi) + dh_1(\xi))}_{-dh_2(\xi)}\\
&= a_0 dh_0(\xi) + a_1 dh_1(\xi) + a_2 dh_2(\xi)
\end{aligned}$$

We can see that both have the same result, this can be repeated for arbitrary N.

$$\sum_{i=1}^{2} (a_i - a_{i-1})\, e_i(\xi) = \sum_{i=0}^{N} a_i dh_i(\xi) \tag{A.23}$$

Appendix B

Extended Gauss Interpolation

Here we will look a bit closer at the expansion of ϕ where the nodes $\phi_{i,j}$ are on the Extended Gauss nodes. Say we define a 1-d EG element that has the following collection of nodes.

$$\begin{cases} \overline{\xi}_0 = -1, \\ \overline{\xi}_i = \tilde{\xi}_i, \\ \overline{\xi}_{N+1} = 1, \end{cases} \tag{B.1}$$

Here $\tilde{\xi}_i, \quad i = 1 \dots N$ are the Gauss points that are the zeros of the N Legendre polynomial $L(\tilde{\xi}_i) = 0$. The corresponding basis functions are an extension of the Gauss basis functions and are given by (A.17). Note that nodes or basis functions of the Extended Gauss mesh are referred to with a bar ($\overline{h}$), and that nodes and basis functions of the Gauss mesh are referred to with a tilde ($\tilde{h}$).

The EG mesh and the G mesh share their internal nodes, (the Gauss mesh has only internal nodes). To expand a solution $\phi(\xi)$ using the EG basis functions is given by,

$$\phi(\xi) = \sum_{i=0}^{N+1} \phi_i \overline{h}_i(\xi). \tag{B.2}$$

If we were to lose information of the endpoints of the 1-d domain, (ϕ_0 and/or ϕ_{N+1}), we can still use the Gauss expansion.

$$\phi(\xi) = \sum_{i=1}^{N} \phi_i \tilde{h}_i(\xi) \tag{B.3}$$

This expansion can be used because the internal nodes of the Gauss and the EG mesh are on the same locations. We mention this because the loss of information of the endpoints, is something that happens in our 2-d extended Gauss domain. If use the 1-d extended Gauss and copy that to 2 dimensions the resulting element looks like Fig. B.1. Here the notation

$\overline{h}_i\overline{h}_j$ stands for $\overline{h}_i(\xi)\overline{h}_j(\eta)$. Since this is just an extension of the 1-d EG mesh to a 2-d EG mesh we write the expansion of $\phi(\xi,\eta)$ grid as,

$$\phi(\xi,\eta) = \sum_{i=0}^{N+1}\sum_{j=0}^{N+1} \phi_{i,j}\overline{h}_i(\xi)\overline{h}_j(\eta) \tag{B.4}$$

Where we took the Extended Gauss polynomials in two directions. In the case of the actual

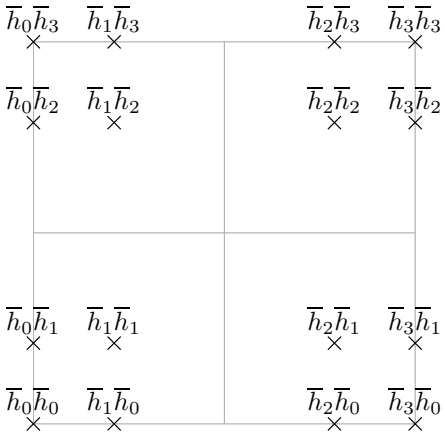


Figure B.1: A direct extension of 1-d to 2-d EG mesh

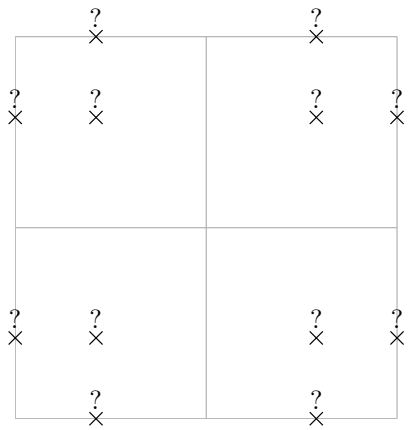


Figure B.2: The real 2-d extended Gauss grid

Extended Gauss grid that we use to solve the PDE's in this thesis, the grid is not just an extension of the 1-d EG mesh. This is because we leave out the corner points. These can/must be left out of the discretization since they neither contribute in a boundary integral nor in a volume/surface integral. This means that we cannot just use the EG polynomials for the reconstruction of the field ϕ. As explained in the 1-d case we can use just the internal nodes for the reconstruction of ϕ which requires the use of the Gauss polynomials. In doing so we do not use the information of the boundary nodes in the reconstruction, which is a waste. Now here we will describe two candidates such that might allow us to include the boundary nodes in the reconstruction of ϕ. These candidates are illustrated in Figures B.3 and B.4. Where Fig. B.3 depicts the discretization given by,

$$\begin{aligned}\phi(\xi,\eta) = \sum_{i=1}^{N}\sum_{j=1}^{N} \phi_{i,j}\overline{h}_i(\xi)\overline{h}_j(\eta) + \sum_{j=1}^{N}\left[\phi_{0,j}\overline{h}_0(\xi) + \phi_{N+1,j}\overline{h}_{N+1}(\xi)\right]\tilde{h}_j(\eta) + \\ \sum_{i=1}^{N}\left[\phi_{i,0}\overline{h}_0(\eta) + \phi_{i,N+1}\overline{h}_{N+1}(\eta)\right]\tilde{h}_i(\xi)\end{aligned} \tag{B.5}$$

and Fig. B.4 depicts the discretization given by

$$\begin{aligned}\phi(\xi,\eta) = \sum_{i=1}^{N}\sum_{j=1}^{N} \phi_{i,j}\overline{h}_i(\xi)\overline{h}_j(\eta) + \sum_{j=1}^{N}\left[\phi_{0,j}\overline{h}_0(\xi) + \phi_{N+1,j}\overline{h}_{N+1}(\xi)\right]\tilde{h}_i(\eta) + \\ \sum_{i=1}^{N}\left[\phi_{i,0}\overline{h}_0(\eta) + \phi_{i,N+1}\overline{h}_{N+1}(\eta)\right]\overline{h}_i(\xi)\end{aligned} \tag{B.6}$$

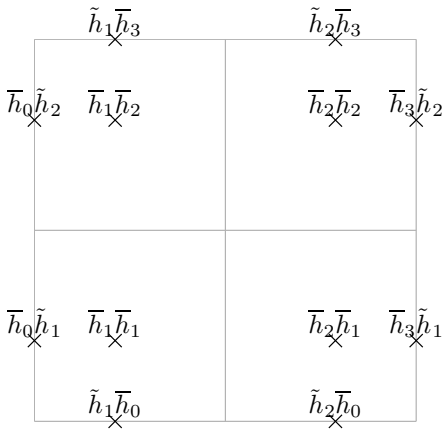


Figure B.3: Whole boundary EG/G polynomials combination

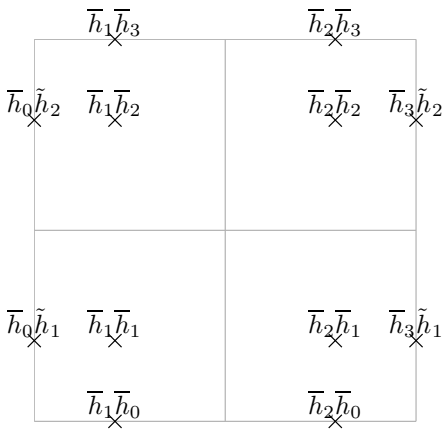


Figure B.4: Half the boundary EG/G polynomials combination

There is one difference between these cases, which is that in the first case the nodes on the boundary are multiplied by a combination of a Gauss and an EG basis functions. In the second case only half of the boundary is expanded using a combination of Gauss and EG basis functions, and the other half of the boundary using EG basis functions only. The reason for this second choice will become clear after we have tested the first choice.

B.1 Testing first candidate

To test the choice of basis functions there is one obvious test case. In that case we make use that of the fact that the sum of all the basis functions should result in a field that has a uniform value, which is 1. Since summing up all basis functions is as the same as taking $\phi_{i,j} = 1$. This results in Fig. B.5 we show the result for three different values of element size N. Here we see that the summation of the basis functions does not result in a plane with height 1. When the order increases we see an improvement but it is clear that (B.5) is not a correct expansion for ϕ. From Fig. B.5 we can see however that the corner points get the value 2. This was to be expected, for this we calculate the value in such a corner point using (B.5) and evaluate at $(\xi, \eta) = (-1, -1)$.

$$\begin{aligned}\phi(-1,-1) = \underbrace{\sum_{i=1}^{N}\sum_{j=1}^{N}\overline{h}_i(-1)\overline{h}_j(-1)}_{=0} + \underbrace{\sum_{j=1}^{N}\left[\underbrace{\overline{h}_0(-1)}_{=1}+\underbrace{\overline{h}_{N+1}(-1)}_{=0}\right]\tilde{h}_j(-1)}_{=1} + \ldots \\ \underbrace{\sum_{i=1}^{N}\left[\overline{h}_0(-1)+\overline{h}_{N+1}(-1)\right]\tilde{h}_i(-1)}_{=1} = 2\end{aligned} \tag{B.7}$$

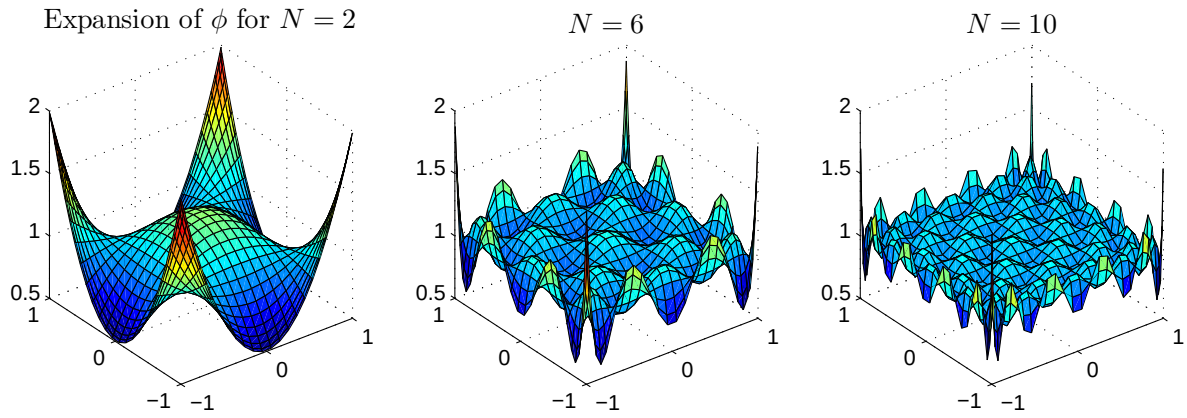


Figure B.5: Surface plot of $\phi(\xi, \eta)$ using (B.5)

B.2 Testing second candidate

Inspired by the fact that with the first candidate the Interpolation does not yield a satisfying result, where the approximation of ϕ in the corner points of the domain is the worst. We propose also the interpolation given by (B.6). By using a combination of Gauss-EG polynomials on two opposite boundaries of the domain and a combination of EG-EG polynomials on the other two opposite boundaries of the domain we hope to find a better interpolation. The result for taking $\phi_{i,j} = 1$ is given for multiple values of N in Fig. It is obvious that this is the desired result and that (B.6) is also a suitable interpolation for ϕ. Now lets test a more challenging function. For this we take,

$$\phi(\xi, \eta) = \sin(\pi\xi)\sin(\pi\eta), \qquad -1 \leq \xi \leq 1, -1 \leq \eta \leq 1 \tag{B.8}$$

To be able to compare we present two interpolations. Namely the mixed Extended Gauss-Gauss interpolation given by (B.6) and the standard Gauss interpolation given by,

$$\sum_{i=1}^{N}\sum_{j=1}^{N} \phi_{i,j}\tilde{h}_i(\xi)\tilde{h}_j(\eta) \tag{B.9}$$

. In Fig. B.7 we show surface plots of the interpolated ϕ minus the exact value of ϕ, $(\mathcal{I}\phi - \phi(\xi, \eta))$. What clearly shows is that the accuracy is about two orders higher when we include the boundary points. When we look at an example where the boundary is non-zero (Fig. (B.8)) we see a much smaller advantage of the inclusion of the boundary nodes. This is mainly due to the lower quality interpolation near the corner nodes.

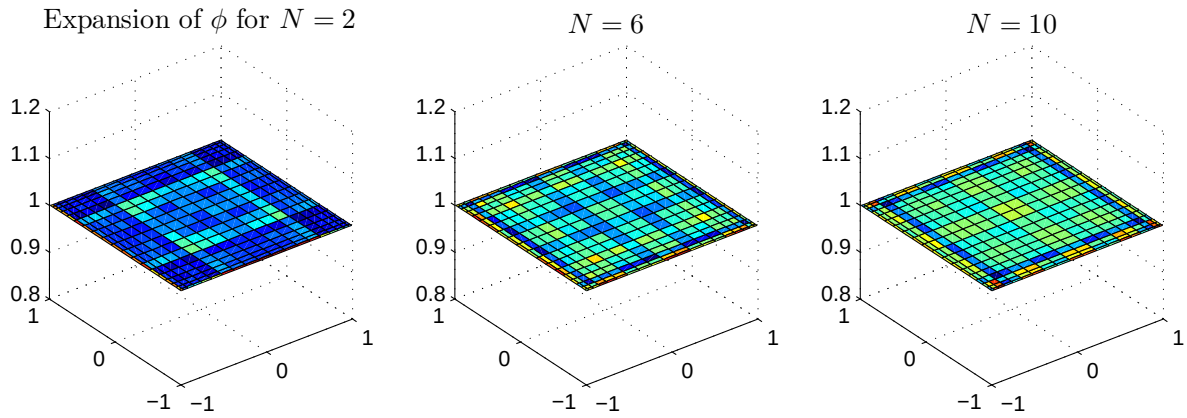


Figure B.6: Surface plot of $\phi(\xi, \eta)$ using (B.6)

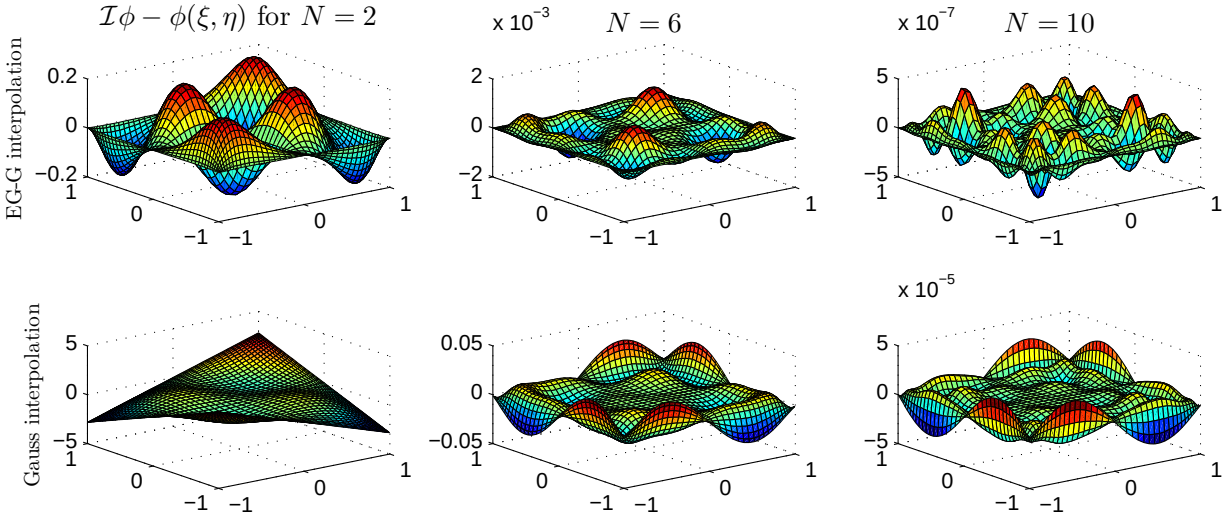


Figure B.7: Surface plot of $(\mathcal{I}\phi - \phi(\xi, \eta))$ using expansions (B.6) and (B.9)

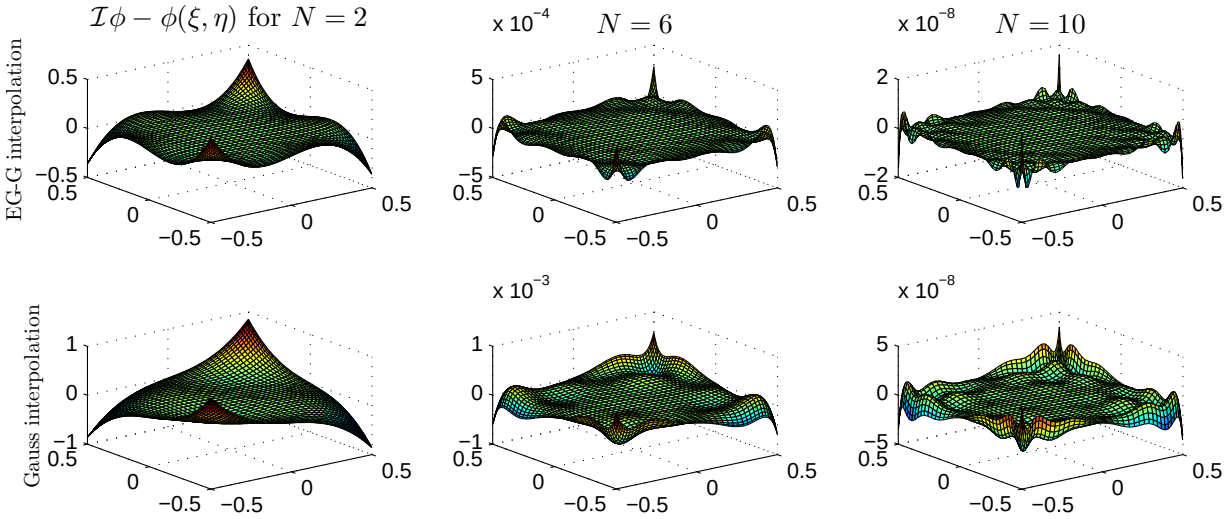


Figure B.8: Surface plot of $(\mathcal{I}\phi - \phi(\xi,\eta))$ using expansions (B.6) and (B.9)

B.3 Final Candidate

As a final possibility we propose to use use the standard EG interpolation. But for this we need values of ϕ at the corner points of the domain, and these are not calculated in our numerical scheme. We obtain these values by using the boundary nodes and Gauss interpolation. We say that the value of the corner nodes are equal to the average of the interpolated values of the 2 adjacent boundaries. So the nodal values at the corner points are given by,

$$\begin{aligned}
\phi_{0,0} &= \phi(-1,-1) = 1/2\left(\sum_{i=1}^{N}\phi_{i,0}\tilde{h}_i(\xi=-1) \;+\; \sum_{j=1}^{N}\phi_{0,j}\tilde{h}_j(\eta=-1)\right)\\
\phi_{N+1,0} &= \phi(1,-1) = 1/2\left(\sum_{i=1}^{N}\phi_{i,0}\tilde{h}_i(1) \;+\; \sum_{j=1}^{N}\phi_{N+1,j}\tilde{h}_j(-1)\right)\\
\phi_{N+1,N+1} &= \phi(1,1) = 1/2\left(\sum_{i=1}^{N}\phi_{i,N+1}\tilde{h}_i(1) \;+\; \sum_{j=1}^{N}\phi_{N+1,j}\tilde{h}_j(1)\right)\\
\phi_{0,N+1} &= \phi(-1,1) = 1/2\left(\sum_{i=1}^{N}\phi_{i,N+1}\tilde{h}_i(-1) \;+\; \sum_{j=1}^{N}\phi_{0,j}\tilde{h}_j(1)\right)
\end{aligned} \tag{B.10}$$

This discretization is not biased, as in the case of (B.6) where we have to choose in what direction we use Gauss or Extended Gauss interpolation. For solutions that are not symmetrical in the corner points, averaging both interpolations in the corner nodes yields a slightly better interpolation. This is shown in Fig. B.9, it does show already in the scaling of the axes. The difference when computing the error in the L_2 norm gives a difference of a factor of 2. So the EG interpolation using the averaged value in the corner points yields an interpolation that

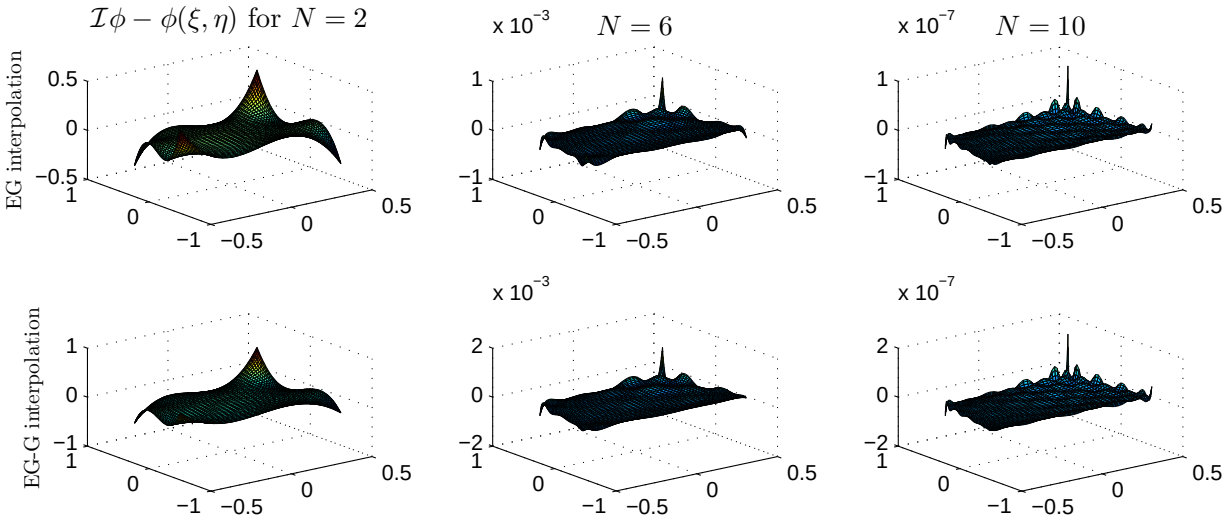


Figure B.9: Surface plot of $(\mathcal{I}\phi - \phi(\xi,\eta))$ on a non-symmetrically deformed grid, using expansions (B.6) and (B.9)

is two times more accurate than the mixed EG-G interpolation, in this non-symmetrically deformed grid test case. Some tests have been done on other functions than the sine function, and the results show that for these functions, the averaged corner point method yields better results than the mixed EG-G interpolation. How much better depends on the asymmetry of the solution.

Appendix C

Matrix construction

Throughout this thesis we use a multitude of matrices to describe actions like interpolation, transformation and integration. In this appendix all the important matrices discussed.

$\mathcal{I}^{GL}$: This matrix is the interpolation matrix for the (n-1)-cochains (discrete integral values) that are connected to the primal cell complex. Using an index notation the matrix $\mathcal{I}^{GL}$ is given by,

$$\begin{aligned}
&\mathcal{I}^{GL}(2r-1,c) = h_k(\xi_i)e_l(\eta_j) \\
&\mathcal{I}^{GL}(2r,c+N(N+1)) = e_l(\xi_i)h_k(\eta_j) \\
&r = i+(j-1)(N+1),\ c = k+(l-1)(N+1),\ 1 \leq i,j,k \leq N+1,\ 1 \leq l \leq N;
\end{aligned} \tag{C.1}$$

This interpolation matrix appears to be completely full, yet this is far from the truth. Since $h_k(\xi_i)$ only has a nonzero value for $k = i$ the structure is sparse as Fig C.1 shows. Starting from the first row of this matrix any two consecutive rows correspond to the values $\begin{pmatrix} q^\xi \\ q^\eta \end{pmatrix}$ which are the proxies of the (n-1)-form $\mathbf{q}^{(n-1)} = -q^\eta dx + q^\xi dy$ in the GL point (ξ_i, η_j).

W^{GL} This matrix is the weight matrix that contains the integration weights from the GL points. It is used to calculate an inner product integrated over the GL square. It is given by,

$$\begin{aligned}
&W^{GL}(2r-1,2r-1) = w_i^{GL}w_j^{GL} \\
&W^{GL}(2r,2r) = w_i^{GL}w_j^{GL} \\
&r = i+(j-1)(N+1),\ 1 \leq i,j \leq N+1
\end{aligned} \tag{C.2}$$

The GL weights themselves are described in Appendix A. The matrix W^{GL} is a diagonal matrix of size $2(N+1)^2$, which is two times the number of GL points. Two times since $\mathbf{q}^{n-1}$ contains two vector proxies.

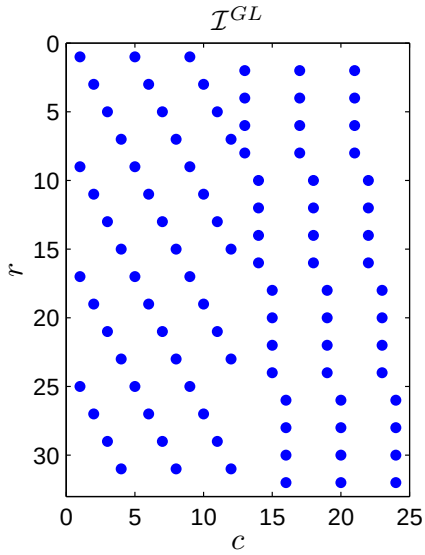


Figure C.1: Sparse structure of the matrix $\mathcal{I}^{GL}$ for $N = 3$

$\mathcal{D}$: This matrix is called the discrete divergence matrix and it consists of two parts, the first is a part for the inside of the domain. The second part operates on the boundary. The inner part we name $\mathcal{D}_i$ and is given by,

$$\begin{aligned}
&\mathcal{D}_i(r, c1) = 1 \quad \mathcal{D}_i(r, c1+1) = -1 \\
&\mathcal{D}_i(r, c2) = 1 \quad \mathcal{D}_i(r, c2+1) = -1 \\
&r = i + (j-1)N,\ c1 = i + (j-1)(N+1),\ c2 = j + (i-1)(N+1) \\
&1 \leq i, j \leq N+1
\end{aligned} \tag{C.3}$$

This is nothing else than the incidence matrix that represents the coboundary operator acting on the cochain $\mathbf{q}$. The boundary part of $\mathcal{D}$ we call $\mathcal{D}_b$ and is given by,

$$\begin{aligned}
&\mathcal{D}_b(2i-1, 1+(i-1)(N+1)) = -1 \\
&\mathcal{D}_b(2i, i(N+1)) = 1 \\
&\mathcal{D}_b(2i-1+2N, 1+(i-1)(N+1)+N(N+1)) = -1 \\
&\mathcal{D}_b(2i+2N, i(N+1)+N(N+1)) = 1 \\
&1 \leq i \leq N
\end{aligned} \tag{C.4}$$

Now concatenating these matrices as $\mathcal{D} = \begin{pmatrix} -\mathcal{D}_i \\ \mathcal{D}_b \end{pmatrix}$ we can show the shape of the discrete divergence matrix which is depicted in Fig C.2.

W^{GG} : This matrix consists of two parts one for the inner part of the domain and one for the boundary. The contain the Gauss quadrature weights, the for the inner part of the domain we name W_i^{GG} and is given by,

$$W^{GG}(r, r) = w_i^G w_j^G, \quad r = i + (j-1)N, \quad 1 \leq i, j \leq N \tag{C.5}$$

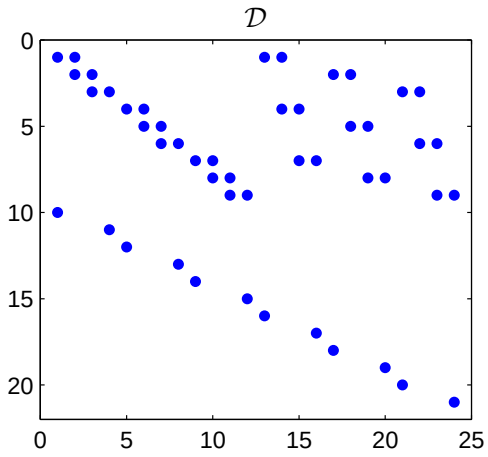


Figure C.2: Sparse structure of the matrix $\mathcal{D}$ for $N = 3$

Here we use the Gauss quadrature weights which are described in appendix A. The boundary part of W^{GG} is called W_b^G where we use only one G to indicate that this involves integration along just one dimension. The matrix W_b^G is given by,

$$\begin{aligned} &W_b^G(2i-1, 1+2(i-1)) = w_i^G \\ &W_b^G(2i, 2i) = w_i^G \\ &W_b^G(2i-1+2N, 1+2(i-1)+2N) = w_i^G \\ &W_b^G(2i+2N, 2i+2N) = w_i^G \\ &1 \leq i \leq N \end{aligned} \tag{C.6}$$

The inner and outer part are concatenated as $W^{GG} = \begin{pmatrix} -W_i^G \\ W_b^G \end{pmatrix}$ which results in a diagonal matrix with size $N^2 + 4N$.

$\mathcal{I}_G$: The matrix $\mathcal{I}^G$ is a concatenation of two parts $\mathcal{I}^G = \begin{pmatrix} \mathcal{I}_i^G & 0 \\ 0 & \mathcal{I}_b^G \end{pmatrix}$. The first is $\mathcal{I}_i^G$ it is an interpolation matrix for the 2-cochains on the primal GL cell complex which returns point values in the Gauss nodes. The action of the matrix $\mathcal{I}_i^G$ is depicted in Fig C.3. The matrix is constructed following,

$$\begin{aligned} &\mathcal{I}_i^G(r, c) = e_k(\xi_i) e_l(\eta_j) \\ &r = i + (j-1)(N+1),\ c = k + (l-1)N,\ 1 \leq i, j, k, l \leq N \end{aligned} \tag{C.7}$$

The matrix $\mathcal{I}_b^G$ is an interpolation matrix for the 1-cochains on the boundary of the primal cell complex that reconstructs the value of $q^\xi(\xi, \eta)$ or $q^\eta(\xi, \eta)$. The matrix's action is depicted

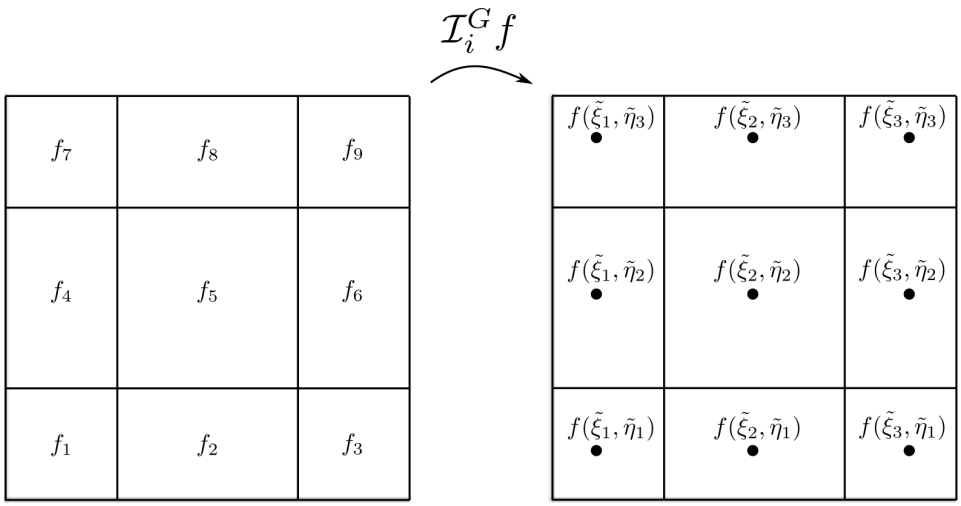


Figure C.3: Action of matrix $\mathcal{I}_i^G$ for $N = 3$

in Fig C.4 and is constructed following,

$$\begin{aligned} \mathcal{I}_b^G(2r-1, 2c-1) &= e_c(\xi_r) \\ \mathcal{I}_b^G(2r, 2c) &= e_c(\xi_r) \\ \mathcal{I}_b^G(2r-1+2N, 2c-1+2N) &= e_c(\eta_r) \\ \mathcal{I}_b^G(2r+2N, 2c+2N) &= e_c(\eta_r) \\ 1 \leq r, c \leq N & \end{aligned} \tag{C.8}$$

The final structure of the matrix $\mathcal{I}^G$ is shown in Fig. C.5

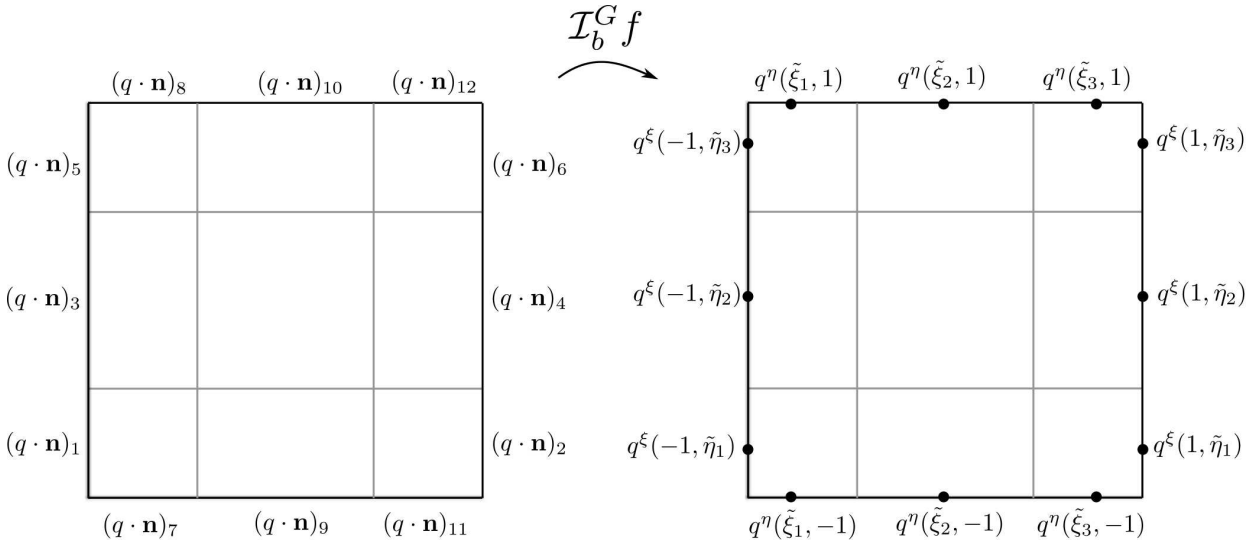


Figure C.4: Action of matrix $\mathcal{I}_b^G$ for $N = 3$

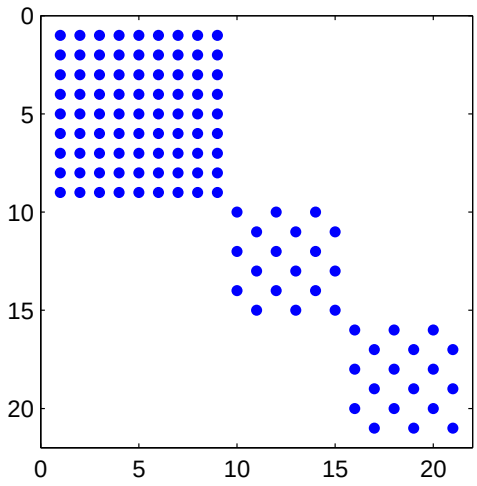


Figure C.5: Structure of matrix $\mathcal{I}_b^G$ for $N = 3$

Appendix D

Numerical integration of line segments and surfaces

When solving for the Poisson equation indicated by (4.1c) we often need to include non-zero boundary conditions. This means we need to calculate $\mathcal{R}q^{(n-1)}$ and $\mathcal{R}f^{(n)}$ which involves integration. While for some cases analytical integration can be done, this is not very practical especially when working with deformed grids. Therefore we explain here how to evaluate these integrals numerically. First we treat the evaluation of $\mathcal{R}q^{(n-1)}$. The term $q^{\xi}_{i,j}$ according to the numbering of Fig. 4.2 is given by,

$$q^{\xi}_{i,j} = \int_{\eta_j}^{\eta_{j+1}} q_\xi(\xi_i, \eta) d\eta. \tag{D.1}$$

Now we will evaluate this integral using Gauss quadrature. Therefore we need to parametrise the coordinate η. This gives,

$$q^{\xi}_{i,j} = \int_{-1}^{1} q_\xi(\xi_i, \eta(s)) \frac{d\eta}{ds} ds, \quad \text{with} \quad \eta(s) = \frac{s+1}{2}(\eta_{j+1} - \eta_j) + \eta_j. \tag{D.2}$$

With a number of Q Gauss quadrature points this integral is evaluated as,

$$q^{\xi}_{i,j} = \sum_{m=1}^{Q} q_\xi(\xi_i, \eta(\tilde{s}_m)) \frac{d\eta}{ds} w^G_m, \tag{D.3}$$

where $\tilde{s}_m$ are the Gauss quadrature points and w^G_m are the Gauss weights. Now the exact same procedure for $q^{\eta}_{i,j}$ results in the equation,

$$q^{\eta}_{i,j} = \sum_{m=1}^{Q} q_\xi(\xi(\tilde{s}_m), \eta_j)) \frac{d\xi}{ds} w^G_m, \quad \text{with} \quad \xi(s) = \frac{s+1}{2}(\xi_{j+1} - \xi_j) + \xi_j. \tag{D.4}$$

For deformed elements the formula is just the same except for the fact that in that case we have that,

$$q_\xi(x(\xi,\eta(s)),y(\xi,\eta(s))) = u_x\frac{\partial y}{\partial \eta} - u_y\frac{\partial x}{\partial \eta} \tag{D.5}$$

$$q_\eta(x(\xi(s),\eta),y(\xi(s),\eta)) = u_y\frac{\partial x}{\partial \xi} - u_x\frac{\partial y}{\partial \xi}, \tag{D.6}$$

In which the coordinates (x, y) depend on the coordinates (ξ, η) which can separately depend on the integration coordinate (s).

The forcing $f^{(n)}$ also involves the evaluation of an integral. But here we need to work with integration in two directions. We evaluate $\mathcal{R}f^{(n)}$ using,

$$\int_{\eta_j}^{\eta_j+1}\int_{\xi_i}^{\xi_i+1} f(\xi,\eta)d\xi d\eta = \int_{-1}^{1}\int_{-1}^{1} f(\xi(s),\eta(r))\frac{\partial \xi}{\partial s}\frac{\partial \eta}{\partial r}dsdr, \tag{D.7}$$

with,

$$\xi = \frac{s+1}{2}(\xi_{i+1}-\xi_i)+\xi_i \quad \text{and} \quad \eta = \frac{r+1}{2}(\eta_{j+1}-\eta_j)+\eta_j. \tag{D.8}$$

This means the integral is evaluated as,

$$\sum_{m=1}^{Q}\sum_{n=1}^{Q} f(\xi(\tilde{s}_m),\eta(\tilde{r}_n))\frac{\partial \xi}{\partial s}\frac{\partial \eta}{\partial r}w_m^G w_n^G \tag{D.9}$$

When working on deformed domains we have that,

$$f^{(n)} = f(x,y)\left(\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}\right)d\xi d\eta. \tag{D.10}$$

Here we needed to include the determinant of the Jacobian matrix.

It usually suffices to have an integration order Q between 4 and 8. Any higher does not give more accurate solutions and only adds in the computation time.

Appendix E

Mappings and Forms

When a domain is subject to a mapping (Φ), like explained in chapter 6, the k-forms can be transformed between these to domains by the Pullback operator. For each k-form this operation is different, here we will give an example of how this works in practice.

E.1 0-forms

Say we have the 0-form $\phi^{(0)}_{x,y}$, which is a point variable, that is on the curvilinear domain indicated by $\Omega(x, y)$ then the pullback operator brings this 0-form to $\Omega'(\xi, \eta)$ in the following way.

$$\tilde{\phi}^{(0)}_{\xi,\eta} \;=\; \Phi^{\star}(\phi^{(0)}_{x,y}) \;=\; \phi^{(0)}_{\Phi(\xi,\eta)} \tag{E.1}$$

This says that the values of ϕ simply mapped onto the other domain. This is illustrated by Figure E.1.

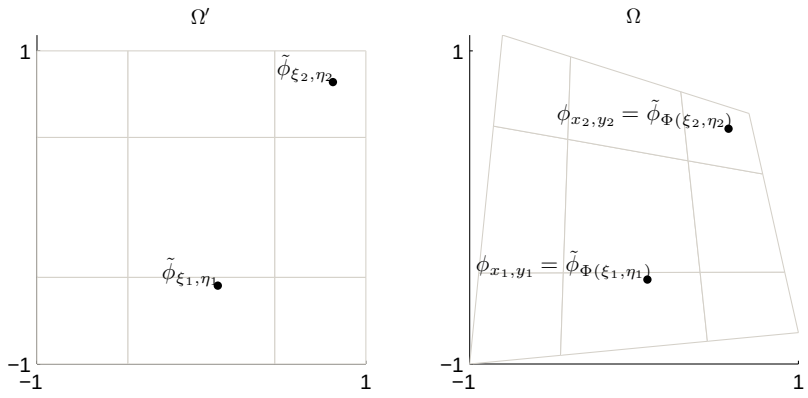


Figure E.1: The action of the pullback operator on 0-form is just a one to one mapping of values from domain Ω' to Ω

E.2 1-forms

The pullback operator working on a 1-form is a bit more complex since we have to take into account the direction that the 1-form (or vector) is pointing in. Now say we want to apply the pullback operator $\Phi^\star$ to the 1-form $\star u^{(1)}$. We start with (E.2), which is the definition of the 1-form u.

$$u^{(1)} = u_x dx + u_y dy \tag{E.2}$$

Next we apply the hodge to this 1-form after which we change the coordinates to get,

$$\begin{aligned} \star\, u^{(1)} = \; q \;&=\; -u_y dx + u_x dy \\ &=\; -u_y \left(\frac{dx}{d\xi}d\xi + \frac{dx}{d\eta}d\eta\right) + u_x \left(\frac{dy}{d\xi}d\xi + \frac{dy}{d\eta}d\eta\right) \\ &=\; \underbrace{\left(-u_y\frac{dx}{d\xi} + u_x\frac{dy}{d\xi}\right)}_{-q_\eta} d\xi + \underbrace{\left(-u_y\frac{dx}{d\eta} + u_x\frac{dy}{d\eta}\right)}_{q_\xi} d\eta \end{aligned} \tag{E.3}$$

Here we write the minus sign at $\left(-u_y\frac{dx}{d\xi} + u_x\frac{dy}{d\xi}\right) = -q_\eta$, this motivated by the fact that when we let the mapping to be $(\xi, \eta) = (x, y)$ we will get $q_\eta = u_y$. We could also write this into matrix notation then we get,

$$\Phi^\star(\star\mathbf{u}^{(1)}_{x,y}) = \underbrace{\begin{pmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial x}{d\eta} \\ -\frac{\partial y}{\partial \xi} & \frac{\partial x}{d\xi} \end{pmatrix}}_{T} \begin{pmatrix} u_x \\ u_y \end{pmatrix}_{\Phi(\xi,\eta)} = \begin{pmatrix} q_\xi \\ q_\eta \end{pmatrix}_{\xi,\eta} \tag{E.4}$$

This equation describes the pullback operator applied to $\star u$ evaluated at a point (x, y). The matrix T operates on the vector proxies (u, v) (in Ω) and gives us the vector proxies (q_ξ, q_η)

of **q** (in Ω'). This can also be done the other way around, when we have the vector proxies (q_ξ, q_η) (in Ω') we can apply the inverse of the transformation matrix T^{-1} to get the vector proxies (u, v) (in Ω). This is important to know since we need this to evaluate an inner product in Ω while our 1-forms are expanded in Ω'.

E.3 2-forms

Since the dimension of our problem is 2 the highest form we have is a 2-form. We treat here f^2 which is also the forcing term in the Poisson problem. In Ω the 2-form $f^{(2)}$ is defined by,

$$f^{(2)} = f_{x,y} dx \wedge dy \tag{E.5}$$

We apply the transformation to dx and dy,

$$\begin{aligned} f_{x,y} dx \wedge dy &= f_{\Phi(\xi,\eta)} \left(\frac{dx}{d\xi} d\xi + \frac{dx}{d\eta} d\eta \right) \wedge \left(\frac{dy}{d\xi} d\xi + \frac{dy}{d\eta} d\eta \right) \\ &= \tilde{f}_{\Phi(\xi,\eta)} \left(\frac{dx}{d\xi} \frac{dy}{d\eta} - \frac{dx}{d\eta} \frac{dy}{d\xi} \right) d\xi \wedge d\eta \end{aligned} \tag{E.6}$$

Here we see that by introducing the coordinate transformation we need to introduce the determinant of the Jacobian matrix to correct the value for f. We have have to make use of the fact that $u \wedge u = 0$ and that $u \wedge v = -v \wedge u$. Then $\tilde{f}$ on Ω' is given by,

$$\tilde{f}_{\xi,\eta} d\xi \wedge d\eta = f_{\Phi(\xi,\eta)} \|J\| d\xi \wedge d\eta \tag{E.7}$$

Appendix F

Large equations

The evaluation of the integral that makes the inclusion of $\mathbb{K}$ possible.

$$
\begin{aligned}
&\int_{\Omega'} (\star\mathbb{K}^{-1} \star^{-1} \Phi^{\star^{-1}} \mathcal{G}\phi^{(0)}, \Phi^{\star^{-1}} q^{(1)}) \Phi^{\star} \omega^n = \\
&\sum_{k=0}^{N}\sum_{l=0}^{N} \Bigg[\underbrace{\left(T^{\xi\xi}_{k,l} \sum_{j=1}^{N} q^{\xi}_{k,j} e_j(\eta_l) + T^{\eta\xi}_{k,l} \sum_{i=1}^{N} q^{\eta}_{k,j} e_i(\eta_l) \right)}_{q^x} \\
&\quad \Bigg(K^{-1}_{11} \underbrace{\left\{ T^{\xi\xi}_{k,l} \sum_{j=1}^{N} (\mathcal{G}\phi)^{\xi}_{k,j} e_j(\eta_l) + T^{\eta\xi}_{k,l} \sum_{i=1}^{N} (\mathcal{G}\phi)^{\eta}_{k,j} e_i(\eta_l) \right\}}_{(\mathcal{G}\phi)^x} + \\
&\qquad K^{-1}_{21} \underbrace{\left\{ T^{\xi\eta}_{k,l} \sum_{j=1}^{N} (\mathcal{G}\phi)^{\xi}_{k,j} e_j(\eta_l) + T^{\eta\eta}_{k,l} \sum_{i=1}^{N} (\mathcal{G}\phi)^{\eta}_{k,j} e_i(\eta_l) \right\}}_{(\mathcal{G}\phi)^y} \Bigg) + \dots \\
&\underbrace{\left(T^{\xi\eta}_{k,l} \sum_{j=1}^{N} q^{\xi}_{k,j} e_j(\eta_l) + T^{\eta\eta}_{k,l} \sum_{i=1}^{N} q^{\eta}_{k,j} e_i(\eta_l) \right)}_{q^y} \\
&\quad \Bigg(K^{-1}_{12} \underbrace{\left\{ T^{\xi\xi}_{k,l} \sum_{j=1}^{N} (\mathcal{G}\phi)^{\xi}_{k,j} e_j(\eta_l) + T^{\eta\xi}_{k,l} \sum_{i=1}^{N} (\mathcal{G}\phi)^{\eta}_{k,j} e_i(\eta_l) \right\}}_{(\mathcal{G}\phi)^x} + \\
&\qquad K^{-1}_{22} \underbrace{\left\{ T^{\xi\eta}_{k,l} \sum_{j=1}^{N} (\mathcal{G}\phi)^{\xi}_{k,j} e_j(\eta_l) + T^{\eta\eta}_{k,l} \sum_{i=1}^{N} (\mathcal{G}\phi)^{\eta}_{k,j} e_i(\eta_l) \right\}}_{(\mathcal{G}\phi)^y} \Bigg) \Bigg] J_{k,l} w^{GL}_k w^{GL}_l
\end{aligned}
\tag{F.1}
$$

$$
\begin{aligned}
J_{k,l} &= \frac{\partial y}{\partial \eta}\frac{\partial x}{\partial \xi} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}\bigg|_{\xi_k,\eta_l} \\
T^{\xi\xi}_{k,l} &= \frac{\partial x}{\partial \xi}\bigg|_{\xi_k,\eta_l}\frac{1}{J_{k,l}}, \qquad T^{\eta\xi}_{k,l} = \frac{\partial x}{\partial \eta}\bigg|_{\xi_k,\eta_l}\frac{1}{J_{k,l}} \\
T^{\xi\xi}_{k,l} &= \frac{\partial x}{\partial \xi}\bigg|_{\xi_k,\eta_l}\frac{1}{J_{k,l}}, \qquad T^{\eta\xi}_{k,l} = \frac{\partial x}{\partial \eta}\bigg|_{\xi_k,\eta_l}\frac{1}{J_{k,l}} \\
K^{-1}_{11} &= \frac{1}{k_{xx}k_{yy}-k^2_{xy}}k_{yy}, \quad K^{-1}_{22} = \frac{1}{k_{xx}k_{yy}-k^2_{xy}}k_{xx}, \\
K^{-1}_{12} &= K^{-1}_{21} = \frac{-1}{k_{xx}k_{yy}-k^2_{xy}}k_{xy}
\end{aligned}
\tag{F.2}
$$