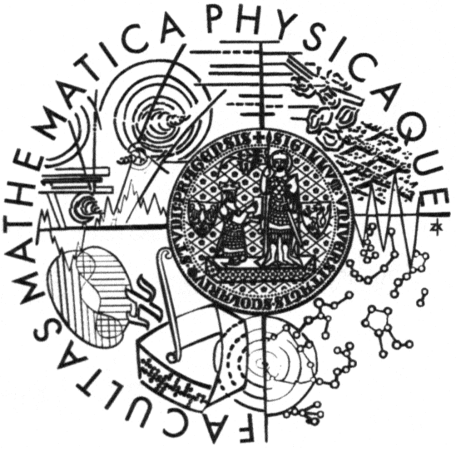
CHARLES UNIVERSITY, PRAGUE
FACULTY OF MATHEMATICS AND PHYSICS

DOCTORAL DISSERTATION



MARTIN ZÍTKA

On Some Aspects of Adaptive Higher-Order Finite Element Method for Three-Dimensional Elliptic Problems

Institute of Mathematics
Academy of Sciences of the Czech Republic

Supervisor: *Prof. RNDr. Karel Segeth, CSc.*
Consultant: *RNDr. Pavel Šolín, PhD.*
Study program: *Scientific and Technical Computing*

2008

Contents

To my wife Svaťa.

Acknowledgments

I would like to express my deep gratitude to all the people who encouraged and supported me throughout my studies.

First, I need to thank my supervisor, Prof. Karel Segeth, whose guidance, understanding, patience and diligent proofreading helped me to improve my thesis to a great degree. I also have to acknowledge his support during my stay at the Mathematical Institute of the Academy of Sciences. I always admired and appreciated his friendly approach to students.

This work would not have been possible without the support and encouragement of Prof. Pavel Šolín, under whose supervision I worked on both my master and doctoral theses. He always provided our research with new interesting topics and novel techniques. I deeply appreciate his support during my studies and I am much obliged for the opportunity he gave me to study and work at the University of Texas at El Paso. During my two year stay at UTEP he not only provided us with interesting research topics, great amount of support and challenging work, but also helped us in many aspects of common life in the United States. It is a great merit of him and his wife, Dagmar Šolínová, that we felt like at home in El Paso.

I also feel greatly indebted to Prof. Ivo Doležel, who gave me the opportunity to work at the Institute of Thermomechanics and supported me during my whole studies. He also supplied us with many interesting physical and engineering problems that brought a lot of inspiration into our research.

I must thank to all the faculty and staff at the Department of Mathematical Sciences at the University of Texas at El Paso that helped us during our stay, especially the chair of the department Prof. Helmut Knaust and our graduate advisor Prof. Piotr Wojciechowski.

Next I would like to thank all my friends and colleagues that helped me

during the work on my thesis. I am grateful to Dr. Tomáš Vejchodský for pointing out many importing theoretical aspects of our work and for moving our research forward with a much faster pace. I would also like to thank my fellow students at the University of Texas at El Paso, Jakub Červený and Martin Lazar, who brought a lot of new thoughts and innovation into the programming part of our research. They also helped make my time in the PhD program more fun and interesting.

Finally, I must thank my whole family for their continuing support and encouragement while pursuing my PhD degree. My deepest thanks belong to my wife Svaťa for her endless support, unceasing patience and love. She has kept me focused and given me encouragement when I needed it most.

CHAPTER 1

Preface

The problem of modelling physical phenomena and their accurate simulation has been driving the progress of mathematics for centuries. The incorporation of computers into this process boosted the development of mathematical tools in the second half of the twentieth century. Undoubtedly, the finite element method (FEM) belongs to the greatest and most widely used theories developed in this period. In this work we deal with the *hp* version of the finite element method (*hp*-FEM), which presents state-of-the-art technology for simulation and approximation of physical, engineering, and mathematical processes.

Higher-order finite element methods are well-established tools for the analysis of partial differential equations. They prove to be very efficient compared to standard finite element methods, especially for large, singular, and multiscale problems. They yield exponential convergence when used appropriately. However, new challenges and problems hindered faster development of higher-order finite element methods in two and three spatial dimensions.

1.1 History of the finite element method

The origins of the finite element method trace back to the 1940s when Richard L. Courant published his paper *Variational methods for the solution of problems of equilibrium and vibration* (see [20]). His work was heavily based on the previous results for variational methods for partial differential equations developed by Rayleigh (see, e.g., [52]), Ritz [54, 45], and Galerkin [29].

Just about a decade after the pioneering Courant's work the finite element method was developed for use in the airframe and structural analysis and gathered momentum at the University of Stuttgart through the work of John Argyris [4, 5]. At the University of Berkeley the finite element method was evolved for use in civil engineering by Ray W. Clough and his collaborators [70, 18, 19]. Another research group that quickly adopted FEM and highly contributed to its wider use among engineers was set up at the University of Wales and lead by Olgierd C. Zienkiewicz [78].

Finite element activity in electrical engineering began in earnest another decade later. In late 1960s, a paper giving the finite-element formulation of the classical hollow-waveguide problem was published by P. Silvester [57]. A concurrent but independent paper published by P. L. Arlett, A. K. Bahrani, and O. C. Zienkiewicz [3] addressed waveguides and cavities. In the following years there was a rapid development in the field. Papers on FEM in magnetostatics, dielectrically loaded waveguides, and other well-known boundary-value problems of electromagnetics were published. The method was quickly applied to integral operators as well.

However, it was not until late 1960s and 1970s when the method was provided with a rigorous mathematical foundation. In the works of Zlámal [81], Strang and Fix [67] and Ciarlet [17] the method finally gained solid mathematical background and attracted wider mathematical audience. The finite element method soon became a strong alternative to the finite difference method, mostly for its ability to naturally deal with domains of various shapes and for better accuracy of the method.

Nowadays, the finite element method is the most widely used approach to the solution of partial differential equations emerging from elasticity, heat transfer, fluid flow, electromagnetics, and other engineering problems.

1.2 Development of different variants of FEM

The finite element method fits nicely into the Galerkin framework. It naturally arises from the weak formulation of the problem, where instead of finding the solution in an infinitely-dimensional space, we restrict ourselves to a subspace of finite dimension. The choice of the subspace is arbitrary but it has a major influence on the approximation error.

1.2.1 *h*-FEM

Originally, the conforming finite element methods were using low-order finite elements, e.g., standard first-order elements for H^1 conforming problems, so-called Whitney elements for $H(\mathrm{curl})$ conforming problems, etc. As the popularity of the finite element method increased, it became obvious that the standard FEM treatment is not suitable for some problems, especially for those exhibiting singularities, shock waves, and multiscale behavior. To address such situations, *h*, *p*, and combined *hp* versions of FEM were introduced.

The *h-FEM* resolves the problematic parts of the solution by successively refining the mesh in the parts where the error is largest. It also became obvious that to achieve a reasonably good convergence rates one has to be able to quantify the error of the approximate solution and hence the *h*-adaptive finite element method boosted the development of robust aposteriori error estimation algorithms. Let us cite [8, 9] as the most crucial works on the topic at that time.

1.2.2 *p*-FEM and *hp*-FEM

p-FEM uses a different concept: instead of the spatial refinement one uses the polynomials of higher degree on elements, where the decrease in error is desirable. It turns out that the combination of both *h* and *p* refinement (*hp*-FEM) yields the best results. The pioneering work on this subject was done mainly by B. Szabó and I. Babuška (see [68]). However, it was not until the late 1990s when the *hp*-FEM gained appropriate attention by the FEM community. This was caused mainly by enormous complexity of *hp*-FEM algorithms and by demands imposed on the mathematician/programmer. There are several

recent monographs that give a detailed overview of the *hp*-FEM (see, e.g., [10, 35, 55, 62]).

The *p*- an *hp*-FEM gave rise several classes of higher-order finite element methods.

H^1 conforming higher-order elements

The *p* version of the finite element method together with the concept of hierarchic higher-order shape functions was pioneered by I. Babuška, B. Szabó, and I. N. Katz (see [11]). They introduced the notion of hierarchic H^1 conforming finite elements, allowing to increase polynomial orders on elements by adding suitable higher-degree polynomial functions only.

$H(\mathrm{curl})$ conforming higher-order elements

Soon the finite elements conforming to various functional spaces were developed. The discretization of Maxwell's equations leading to variational problems in $H(\mathrm{curl})$ space lead to the constitution of vector-valued $H(\mathrm{curl})$ conforming finite elements, called *Whitney* or *edge elements* (see [47, 76]). Due to the increasingly widespread interest in the use of higher-order finite element schemes enormous effort was put in the construction of higher-order edge elements. Some higher-order nodal-based edge element schemes were developed (see [1, 75]), but the first hierarchic edge elements have not appeared until recently (see [2, 24, 59]).

Higher-order elements for CFD problems

In the computational fluid dynamics community the need for higher order finite elements was addressed by constitution of *Taylor-Hood elements* [32] for incompressible Navier-Stokes equations. Nowadays the higher-order methods drive the progress in the CFD field especially due to the rise of the *discontinuous Galerkin method*. It was initially introduced by Reed and Hill [53] in 1973 as a technique to solve neutron transport problems. Lesaint and Raviart [44] presented the first numerical analysis of the method. However, the technique

has only recently become popular as a method for solving fluid dynamics problems. The attractivity of this method lies in the fact that it is able to capture discontinuities at a very high resolution and it is relatively easy to implement and parallelizable. Since the support of all the basis functions in the discontinuous Galerkin finite element space is one element only, the construction of hierarchic basis is straightforward. There are many journal articles and monographs dealing with the *hp*-Discontinuous Galerkin, among which let us cite, e.g., [27, 33, 34, 25].

Spectral element method

The *spectral element method* is a higher-order finite element technique that combines the geometric flexibility of the FEM with the high accuracy of the spectral methods. The method was developed by Anthony T. Patera at MIT and Yvon Maday at Paris-VI (see [48, 46]). Later the method was adapted for use with various physical problems (see, e.g., [36, 35]). It exhibits several favorable computational properties, such as the use of tensor products and naturally diagonal mass matrices. There are two main features of the spectral element method. First, the method is implemented by using Lagrange interpolant shape functions within each element, where the nodes of the shape functions are placed at the zeros of Legendre polynomials (Gauss-Lobatto points) mapped from the reference domain to each element. Next, the efficiency of the method is achieved by using the Gauss quadrature for evaluating integrals. The quadrature points reside at the nodal points which enables fast techniques to be used for iterative solution methods. Unfortunately, the spectral element method imposes a serious drawback. It is generally very difficult to combine elements of different polynomial order in the same computational mesh.

1.3 *hp*-FEM software

Higher-order finite element codes and *hp*-adaptive strategies began to appear during the 1990s just after the immense development in the field of computer technology and programming languages in the previous decades. More afford-

able workstations and developer-friendly languages and libraries allowed the mathematicians to write complex *hp*-FEM codes and verify theoretical results. Let us mention the most important software projects dealing with *hp*-FEM.

The research group of Leszek Demkowicz at the University of Texas at Austin is working on several *hp*-FEM software packages solving physical problems on one-, two-, and three-dimensional domains using higher order H^1- and $H(\text{curl})$-conforming elements. They have implemented fully automatic *hp*-adaptivity algorithms on 1-irregular meshes. See [23, 22] for further reference.

Another higher-order finite element code, called "deal.II"[1], emerged from work of the Numerical Methods Group at the University of Heidelberg. Its main authors, Wolfgang Bangerth, Ralf Hartmann, and Guido Kanschat, currently work at the Texas A&M University and the Institute of Aerodynamics and Flow Technology of the German Aerospace Center (DLR) in Braunschweig. It supports computations in one, two, and three dimensions, handling of locally refined grids, including different adaptive strategies based on local error indicators and error estimators. All *h*, *p*, and *hp* refinement are fully supported for continuous and discontinuous elements. See [13] for more information on this project.

The software developed during the work on this thesis is a part of the HERMES project of the *hp*-FEM group[2] at the University of Texas at El Paso. The aim of the project is to develop *hp*-adaptive algorithms for use in multi-physics and coupled problems. The first step was the development of a 2D code solving elliptic problems and time-harmonic Maxwell's equations using H^1 and $H(\text{curl})$ conforming finite elements up to the tenth order (see [65, 72]). Later the automatic *hp*-adaptive algorithms allowing computation on meshes with arbitrary-level hanging nodes were implemented by J. Červený for H^1 and $H(\text{curl})$ conforming elements and tested with outstanding results (see [16, 60, 61]). A similar adaptive strategy with arbitrary-level hanging nodes in three dimensions was successfully implemented by P. Kús (see [42, 43]). The main focus of the project now is targeted on multi-physics and time-dependent problems in 3D. The emphasis is put on *hp*-adaptive algorithms

[1] `http://dealii.org`
[2] `http://hpfem.math.utep.edu`

using the method of arbitrary-level hanging nodes and moving meshes.

1.4 Objectives of this work

Theoretical and practical results for one-dimensional *hp*-FEM are well-covered in the literature of the last two decades of the twentieth century and show a lot of optimism for future research. Further studies in two- and especially in three dimensions revealed several problems that needed to be addressed:

- some theoretical aspects and algorithms could not be applied directly and needed to be redesigned for purely higher-dimensional problems,
- the complexity of implementation imposed greater demands on the mathematician/programmer.

The main aim of this work was to address these problems.

In particular, it is well-known that linear problems obtained by the *hp*-FEM are usually much smaller than the ones produced by standard methods. Unfortunately, as the problems get smaller, the conditioning properties of the problem are still ill-conditioned and sometimes even cause iterative solvers to fail. We thoroughly investigate this issue and define new sets of shape functions with better conditioning properties.

Unfortunately, as the properties of the linear problem strongly depend on the underlying differential operator, we must define different sets of shape functions for different operators to get some sort of optimality. We present several methods of shape function optimization together with comparison to standard functions.

As mentioned above, the *hp*-FEM usually produces much smaller linear systems while preserving the accuracy of the solution. However, when performing larger computations (e.g., in 3D), one can see that the computational costs may exceed the capabilities of a regular computer if implemented inoptimally. The optimality of implementation of the *hp*-FEM in 3D is another major focus of this work.

Original contribution of the author is presented in Chapters 3, 4, and 5. These are mainly results obtained during the work on the HERMES project

at the University of Texas at El Paso and at the Academy of Sciences of the Czech Republic under the supervision of professors Pavel Šolín, Ivo Doležel and Karel Segeth. Most of the results were obtained during the collaboration with my supervisors and my colleagues, mainly Tomáš Vejchodský from the Mathematical Institute of the Academy of Sciences, and Francisco Ávila from the University of Texas at El Paso. The results which this thesis is based on were also published in various journals and conference proceedings. The work concerning shape function optimization and non-affine concept of FEM was presented in [64, 80]. A significant part of the research was devoted to the implementation of the 2D and 3D *hp*-FEM code. The associated results were published in [65, 66, 72, 73].

CHAPTER 2

Introduction to higher-order finite element method

In this introductory chapter we give a brief introduction to the *hp*-FEM framework. We present our model problem — a linear second order elliptic partial differential equation on a polytopic domain. We then show the discretization strategy used by *hp*-FEM and we present the main reasons which lead us to this method. Finally, we must mention the major challenges and problems the method imposes.

2.1 Model problem

Throughout this work we stick to a linear second order elliptic partial differential equation on a polytopic domain as our model problem.

2.1.1 Classical formulation

Consider a polytope $\Omega \subset \mathbb{R}^d$ such that $\Gamma_D, \Gamma_N \subset \partial\Omega$ are relatively open in $\partial\Omega$, $\Gamma_D \neq \emptyset$, $\Gamma_D \cap \Gamma_N = \emptyset$, and $\overline{\Gamma_D} \cup \overline{\Gamma_N} = \partial\Omega$. Next, let $a_1 \in C^1(\Omega)$, $g_D \in C(\partial\Omega)$ and $g_N \in C(\partial\Omega)$. The problem is to find $u \in V = C^2(\Omega) \cap C(\overline{\Omega})$ such that

$$-\nabla \cdot (a_1(\boldsymbol{x})\, \nabla u) = f \quad \text{in } \Omega, \tag{2.1}$$

$$u|_{\Gamma_D} = g_D \quad \text{on } \Gamma_D, \tag{2.2}$$

$$\frac{\partial u}{\partial \boldsymbol{\nu}} = g_N \quad \text{on } \Gamma_N. \tag{2.3}$$

Here $a_1(\boldsymbol{x})$ is a space-dependent coefficient, f represents the so called *load function*, $\boldsymbol{\nu}$ is the outward unit normal to $\partial\Omega$, (2.2) and (2.3) stand for the *Dirichlet* and *Neumann* boundary condition, respectively.

We assume that all functions mentioned in the classical formulation lie in such functional spaces that all operations make sense.

Remark. Despite not being the most general, the formulation (2.1) still describes several physical phenomena, for example the stationary heat transfer, distribution of electrostatic and gravitational potential, transverse deflection of a cable, axial deformation of a bar, to name the most important, and others.

2.1.2 Weak formulation

To be able to formulate the problem in terms of vector spaces, we need to introduce the so-called *Dirichlet lift* $G \in C^2(\Omega) \cap C(\overline{\Omega})$ such that

$$G = g_D \text{ on } \Gamma_D. \tag{2.4}$$

Then we can introduce a new unknown function $\bar{u} \in \{v \in C^2(\Omega),\ v|_{\Gamma_D} = 0\}$, so that $u = \bar{u} + G$. We reformulate (2.1), (2.2) and (2.3) as

$$-\nabla \cdot \big(a_1(\boldsymbol{x})\, \nabla(\bar{u} + G)\big) = f \quad \text{in } \Omega, \tag{2.5}$$

$$\bar{u}|_{\Gamma_D} = 0 \quad \text{on } \Gamma_D, \tag{2.6}$$

$$\frac{\partial(\hat{u} + G)}{\partial \boldsymbol{\nu}} = g_N \quad \text{on } \Gamma_N. \tag{2.7}$$

The Dirichlet lift G may be chosen arbitrarily to satisfy condition (2.4) as we will show later that the solution u is invariant under its concrete choice.

We use the standard procedure to obtain the weak formulation: we multiply the equation (2.5) by a smooth test function v, integrate over the domain Ω and use the Green's theorem to reduce the order of differentiation. Finally we can weaken the smoothness requirements of all the functions involved so that the integrals still make sense.

The weak formulation of this problem then reads: find $u \in H^1(\Omega)$ such that $u = \bar{u} + G$, where $\bar{u} \in V = \{v \in H^1(\Omega), v|_{\Gamma_D} = 0\}$, $G \in H^1(\Omega)$, $G|_{\Gamma_D} = g_D$ in the sense of traces, $f \in L^2(\Omega)$, $a_1 \in L^\infty(\Omega)$, and

$$\int_\Omega a_1(\boldsymbol{x})\, \nabla\bar{u} \cdot \nabla v \, \mathrm{d}\boldsymbol{x} = \int_\Omega f v \, \mathrm{d}\boldsymbol{x} - \int_\Omega a_1(\boldsymbol{x})\, \nabla G \cdot \nabla v \, \mathrm{d}\boldsymbol{x} + \int_{\Gamma_N} a_1(\boldsymbol{x})\, g_N v \, \mathrm{d}S \quad \forall v \in V. \tag{2.8}$$

Let us introduce the following notation:

$$a(u, v) := \int_\Omega a_1(\boldsymbol{x})\, \nabla u \cdot \nabla v \, \mathrm{d}\boldsymbol{x}, \tag{2.9}$$

$$l(v) := \int_\Omega f v \, \mathrm{d}\boldsymbol{x} - \int_\Omega a_1(\boldsymbol{x})\, \nabla G \cdot \nabla v \, \mathrm{d}\boldsymbol{x} + \int_{\Gamma_N} a_1(\boldsymbol{x})\, g_N v \, \mathrm{d}S. \tag{2.10}$$

Hence, it is our task to find a function $\bar{u} \in V$ such that

$$a(\bar{u}, v) = l(v) \quad \forall v \in V. \tag{2.11}$$

Let us assume that $a_1(\boldsymbol{x}) \geq C_{\min} > 0$ almost everywhere in Ω and that all the functions lie in aforementioned functional spaces. Then it can be shown easily (see [58]) that the bilinear form $a(\cdot, \cdot)$ is V-elliptic and bounded and l is a bounded linear form. It has several positive consequences:

- The function $|||v|||_e^2 = a(v, v)$ defines a norm in the space V that is equivalent with the norm $\|\cdot\|_V = \|\cdot\|_{H^1(\Omega)}$,
- the bilinear form a and the linear form l satisfy the assumptions of the Lax-Milgram lemma and thus $\bar{u}$ exists and is unique.

Independence of the solution u on the Dirichlet lift G

Assume that there are two weak solutions $u_1 = \bar{u}_1 + G_1$ and $u_2 = \bar{u}_2 + G_2$. We write (2.8) for both u_1 and u_2 and subtract them. Then it is straightforward to conclude that

$$\int_\Omega a_1(\boldsymbol{x})\, \nabla(u_1 - u_2) \cdot \nabla v \, \mathrm{d}\boldsymbol{x} = 0 \tag{2.12}$$

for all $v \in V$. Since $u_1 - u_2 \in V$, we can take the difference $u_1 - u_2$ as our test function v. Hence,

$$\int_\Omega a_1(\boldsymbol{x}) \left|\nabla(u_1 - u_2)\right|^2 = 0, \tag{2.13}$$

which implies $||u_1 - u_2||_e^2 = 0$, or, in other words, $u_1 = u_2$ almost everywhere.

2.1.3 *hp*-FEM formulation

Definition 2.1. Finite element mesh $\mathcal{T}_{h,p} = \{K_1, K_2, \ldots, K_M\}$ over a polytope $\Omega \subset \mathbb{R}^d$ is a geometrical division of Ω into a finite number of nonoverlapping open polytopic cells (*elements*) K_i such that

$$\overline{\Omega} = \bigcup_{i=1}^{M} \overline{K_i}. \tag{2.14}$$

Each cell $K_i, i = 1, \ldots, M$, is equipped with a polynomial degree $1 \leq p(K_i) = p_i$.

Remark. To stay in a more realistic setting, from now on we consider our problem to live in a three dimensional space, i. e., $d = 3$. The word "cell" may have different meanings in $\mathbb{R}^3$: it can represent a tetrahedron, triangular prism, hexahedron, pyramid, etc.

Definition 2.2. A mesh is called *regular*[1] if for two arbitrary elements $K_i, K_j \in \mathcal{T}_{h,p}$, $i \neq j$, exactly one of the following alternatives holds:

- $\overline{K_i} \cap \overline{K_j}$ is empty,

[1] In some publications the denomination *face-to-face* or *conforming mesh* is used instead. However, we use the adjective "conforming" in a different meaning. Thus we will stick to the attribute *regular* to avoid ambiguity.

- $\overline{K_i} \cap \overline{K_j}$ is a single common vertex,
- $\overline{K_i} \cap \overline{K_j}$ is a single common edge,
- $\overline{K_i} \cap \overline{K_j}$ is a single common face.

Remark. Although one can perform discretizations using irregular meshes, i. e., meshes containing the so called *hanging nodes*, they substantially complicate the discretization procedure. It is also desirable to require regularity at least for initial (non-refined) meshes. However, schemes have been developed that allow for computation with 1-irregular meshes (see, e. g., [22, 23]). Recently it was also shown that it is possible to allow for irregularity of arbitrary degree for 2D problems (see [16, 60]). Despite its complexity, this novel concept was generalized to 3D problems most recently and proved to be very successful (see [41, 42, 43]).

We have polynomial degrees assigned to each element in the mesh. To be able to formulate our FEM problem in terms of linear spaces, we need to assign polynomial degrees to faces and edges of the mesh. That is why we need to employ the *minimum rule*:

- Let s_i be a face shared by two neighboring elements, i. e., $s_i = \overline{K_j} \cap \overline{K_l}$. The polynomial degree assigned to the face is defined as $p_{s_i} = \min(p_j, p_l)$.
- Let e_i be an edge shared by n neighbors, i. e., $e_i = \overline{K_{i_1}} \cap \overline{K_{i_2}} \cap \cdots \cap \overline{K_{i_n}}$. The polynomial degree assigned to the edge is defined as $p_{e_i} = \min(p_{i_1}, p_{i_2}, \ldots, p_{i_n})$.
- If an element K lies on the boundary, the polynomial degree on $K \cap \partial\Omega$ is dictated by the polynomial degree associated to the element interior.

Remark. It turns out that the application of the minimum rule is the only way we can make the finite element space independent of a concrete choice of basis functions (see [62] for further details).

For the sake of simplicity, let us define the following sets:

- Let $\mathcal{S} = \{s_i, i = 1, \ldots, M_s\}$ be the set of all faces in the mesh $\mathcal{T}_{h,p}$, where each face s_i is equipped by its polynomial degree p_{s_i}.

- Let $\mathcal{E} = \{e_i, i = 1, \ldots, M_e\}$ be the set of all edges in the mesh $\mathcal{T}_{h,p}$, where each edge e_i is equipped by its polynomial degree p_{e_i}.

Now we are ready to define the *finite element space* as

$$V_{h,p} = \{v \in V; v|_{K_i} \in P^{p_i}(K_i),\ K_i \in \mathcal{T}_{h,p};\\ v|_{s_j} \in P^{p_{s_j}}(s_j),\ s_j \in \mathcal{S}; v|_{e_k} \in P^{p_{e_k}}(e_k),\ e_k \in \mathcal{E}\}, \quad (2.15)$$

where

$$P^{p_i}(K_i) = \{v \text{ is polynomial of degree at most } p_i \text{ on } K_i\}. \quad (2.16)$$

Analogous definition is used for $P^{p_{s_j}}(s_j)$ and $P^{p_{e_k}}(e_k)$.

Remark. Note that the previous definition of the polynomial space $P^{p_i}(K_i)$ corresponding to an element K_i is formulated very vaguely. First, the meaning of the local order of approximation p_i depends on the type of underlying element; we consider a single order of approximation $p_{t,i}$ for tetrahedral elements. On the other hand, due to the Cartesian product geometry nature of hexahedral elements we may prescribe an anisotropic local order of approximation as a triplet

$$p_{h,i} = \{p_{h,i,1}, p_{h,i,2}, p_{h,i,3}\}. \quad (2.17)$$

Each of the numbers $p_{h,i,j}$, $j = 1, 2, 3$, specifies an internal directional order of approximation.

Next we can formulate the FEM problem as to find $u_{h,p} \in V_{h,p}$ such that

$$a(u_{h,p}, v_{h,p}) = l(v_{h,p}) \qquad \text{for all } v_{h,p} \in V_{h,p}. \quad (2.18)$$

Since $V_{h,p}$ is a finite-dimensional linear space, we can consider a finite basis $\mathcal{B} = \{v_i,\ i = 1, \ldots, N\}$, where $N = \dim V_{h,p}$. Then the solution can be expanded in terms of v_i as

$$u_{h,p} = \sum_{i=1}^{N} y_i v_i, \quad (2.19)$$

where y_i, $i = 1, \ldots, N$, are unknown coefficients. Now the finite element formulation (2.18) can be rewritten as

$$\sum_{i=1}^{N} y_i\ a(v_i, v_j) = l(v_j) \qquad \text{for all } j = 1, \ldots, N. \quad (2.20)$$

Let us use the following notation:

- $A = \{a_{ij}\}_{i,j=1}^{N}$ is the so-called *stiffness matrix* with entries

$$a_{ij} = a(v_j, v_i),\ i,j = 1,\ldots,N,$$

- $\boldsymbol{y} = (y_1,\ldots,y_N)^T$ is the vector of unknown coefficients,
- $\boldsymbol{l} = (l_1,\ldots,l_N)^T$ is the so-called *load vector* with the entries

$$l_j = l(v_j),\ j = 1,\ldots,N.$$

Now it is easy to reformulate the problem as a system of N linear algebraic equations

$$A\boldsymbol{y} = \boldsymbol{l}. \tag{2.21}$$

There are two parts of the solution process that consume most of the CPU time:

- assembly of the stiffness matrix A and the load vector $\boldsymbol{l}$,
- numerical solution of the linear system (2.21).

We will address both these problems in the following chapters.

Remark. For standard finite element methods the computational cost of stiffness matrix and load vector assembly is usually insignificant compared to the cost of the linear system solution process. However, for higher-order methods the stiffness matrix assembly tends to take more time than for standard FEM and we need to take this fact into consideration.

Problems discretized using *hp*-FEM usually lead to much smaller linear problems compared to standard FEM discretizations of comparable precision, but there are several aspects that constantly deteriorate the process of numerical solution:

- the stiffness matrix is still sparse, but the relative number of nonzero entries is usually greater than in the case of lower-order FEM,
- the conditioning properties of the stiffness matrix are worse (relative to its size) than in the case of low-order FEM and can seriously influence the performance of numerical solvers.

2.2 Affine concept of the finite element method

Modern implementations of the finite element method usually follow the so-called *affine concept:* Every mesh element K_i is mapped onto its reference counterpart $\hat{K}$ via an affine reference mapping $\boldsymbol{x}_{K_i} : \hat{K} \to K_i$. The reference domain $\hat{K}$ is then equipped with a set of hierarchic shape functions. The global basis functions that have their support in the element K_i are defined to be the images of these shape functions through the reference map.

The affine concept of FEM is the standard way to construct the basis of the space $V_{h,p}$. We explain this approach thoroughly in this section on an example of tetrahedral and hexahedral finite elements.

Remark. The affine concept also poses some limitations. E.g., it may be an obstacle in the process of finding the optimal basis for the *hp*-FEM, as we will see in Chapter 3.

Although the basis $\mathcal{B}$ of the space $V_{h,p}$ can be constructed arbitrarily, we usually require the following standard properties to facilitate the FEM solution process:

- The basis functions should have *small supports*. Due to the nature of the stiffness matrix assembly process, this requirement results in the sparse structure of the stiffness matrix.
- The basis should be *hierarchic*. Consider a finite element space P of functions which are continuous on Ω. Moreover, on each element $K_i \in \mathcal{T}_{h,p}$ they are polynomial of order at most p. Let $\mathcal{B}^p$ be a basis of such a space. The basis is called hierarchic if $\mathcal{B}^p \subset \mathcal{B}^{p+1}$ for all $p \geq 1$. In other words, increasing the polynomial degree in the definition of the space results in adding new basis functions only.

These requirements have several positive consequences. First, the basis of the finite element space can be constructed easily by adding basis functions to elements, faces, and edges, up to the required polynomial degree. Next, as we will see later, hierarchic basis can be easily set up by putting together various types of functions with small supports.

2.2.1 Hierarchic basis of $V_{h,p}$ on 3D meshes

Let us define the following types of functions:

- *Vertex functions* associated with mesh vertices. The support of a vertex function is a patch $S(v)$ of elements sharing the common vertex v. The function is nonzero at the common vertex, zero at the other vertices of the patch, and piecewise linear. It is zero outside of $S(v)$.
- *Edge functions* associated with mesh edges. The support of an edge function is a patch $S(e)$ of elements sharing the common edge e. The function is nonzero on the common edge, zero on the other edges of the patch and piecewise polynomial. It is zero outside of $S(e)$.
- *Face functions* associated with mesh faces. The support of a face function consists of two neighboring elements sharing the common face. The function is nonzero on the common face, zero on the other faces of the two elements and piecewise polynomial. It is zero outside of this two-element patch.
- *Bubble functions* associated with element interiors. The support of a bubble function lies inside the element to which it belongs. It is zero on all faces of the element and polynomial in the interior of the element. It is zero outside of the element.

Now we are ready to describe a hierarchic basis of $V_{h,p}$. Let $\mathcal{T}_{h,p}$ be a mesh covering a polyhedral domain Ω as defined in Section 2.1.1. The hierarchic basis $\mathcal{B}^p$ of the space $V_{h,p}$ can be constructed using the following basis functions:

- Vertex functions associated with vertices which do not lie on $\overline{\Gamma}_D$.
- Edge functions associated with edges which do not lie on $\overline{\Gamma}_D$. To each edge e_i, $i = 1, \ldots, M_e$, we assign as many edge functions as required by the polynomial degree p_{e_i}.
- Face functions associated with the faces which do not lie on $\overline{\Gamma}_D$. To each face s_i, $i = 1, \ldots, M_s$, we assign as many face functions as required by the polynomial degree p_{s_i}.

- Bubble functions: to each element K_i we assign as many bubble functions as required by the polynomial degree p_i.

Remark. Note that the previous text was a mere description of the hierarchic basis. The exact mathematical process to define it will be presented in Section 2.2.9. We can see that the nature of the hierarchic basis depends on the choice of the polynomial degrees associated with mesh edges, faces, and element interiors. Although we used just one polynomial degree for each of them, note that in practice we need to specify different number of polynomial degrees for different types of mesh entities, as we will see in the later sections of this chapter. For example, there is only one polynomial degree associated with every triangular face of the mesh or with every tetrahedral element interior. On the other hand, we must specify two directional polynomial degrees for every quadrilateral face and three polynomial degrees to every hexahedral element interior. Details will be discussed later in this chapter.

Remark. During the 1960s and 1970s, only spaces of piecewise linear or quadratic functions have been commonly used for FEM discretization. However, these spaces proved to have inoptimal approximation properties for problems with singular or multiscale character. That is why higher-order methods are becoming increasingly popular. First, the p-version of FEM was introduced to improve the properties of the finite element space. However, this method may require an excessive number of *degrees of freedom*.

Further research revealed that the polynomial degree only should be raised in regions where the solution is smooth while spatial refinement is more appropriate near singularities. The requirement of saving as many degrees of freedom as possible leads to two important techniques in hp-FEM:

- p-refinement – raising the polynomial degree where effective, thus we need to allow for different polynomial degrees on different mesh elements,
- h-refinement – spatial refinement of elements, thus we need to develop algorithms for computation on irregular meshes.

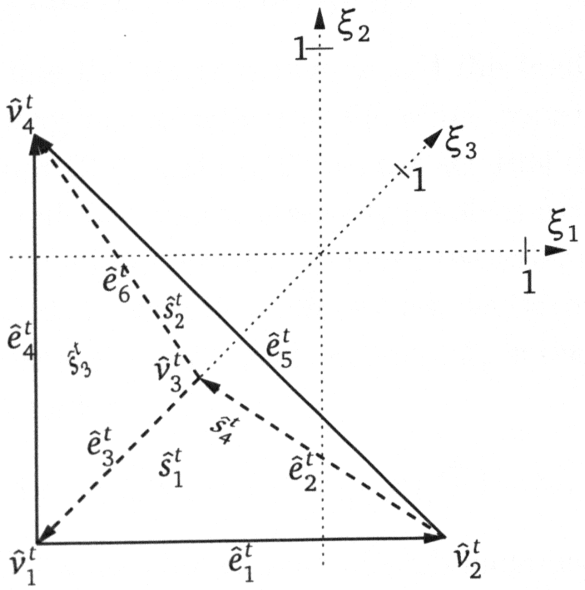


Figure 2.1: Reference tetrahedron $\hat{K}_t$.

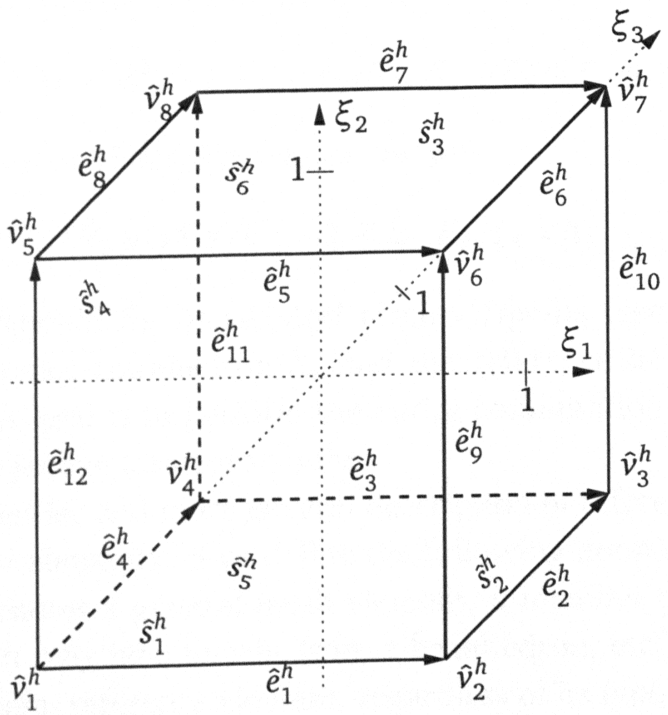


Figure 2.2: Reference hexahedron $\hat{K}_h$.

2.2.2 Reference elements and reference mappings

To be able to assemble the stiffness matrix and the load vector in an algorithmically feasible way, one usually uses the affine concept of FEM. Looking briefly at the equations (2.9) and (2.10) we can see that the actual process of the stiffness matrix and load vector assembly involves a lot of integration, as it is usual in the FEM solution process. To make the assembly process feasible from mathematical and algorithmic point of view, we introduce a special *reference* (or *master*) element $\hat{K}$ and for every element K_i in the mesh we construct a smooth bijective mapping

$$\boldsymbol{x}_{K_i} : \xi \in \mathbb{R}^3 \to \boldsymbol{x} \in \mathbb{R}^3, \quad \boldsymbol{x}_{K_i}(\hat{K}) = K_i, \tag{2.22}$$

(see Figures 2.3 and 2.4 for visualization of such mappings for a tetrahedron and a hexahedron, respectively). More specifically, for each type of cell present in $\mathcal{T}_{h,p}$ we need to present a reference equivalent with a set of integration rules (numerical quadrature).

For tetrahedral mesh elements we use the set

$$\hat{K}_t = \{\xi \in \mathbb{R}^3;\ -1 < \xi_1, \xi_2, \xi_3;\ \xi_1 + \xi_2 + \xi_3 < -1\} \tag{2.23}$$

as a reference domain. Similarly, we use the set

$$\hat{K}_h = \{\xi \in \mathbb{R}^3;\ -1 < \xi_1, \xi_2, \xi_3 < 1\} \tag{2.24}$$

as a reference domain for hexahedral mesh elements (see Figures 2.1 and 2.2 for the reference tetrahedron $\hat{K}_t$ and the reference hexahedron $\hat{K}_h$, respectively). Our goal is to perform the entire computation on the reference domains using bijective transformations.

To allow for easier and more general description of reference mappings for different types of elements let us define the following notation: Unless stated otherwise, K_i denotes a general mesh element, a member of $\mathcal{T}_{h,p}$, regardless of its type (i.e., it may be a tetrahedron, a hexahedron, etc.). Similarly, by $\hat{K}$ we denote a general reference element, regardless of its type (e.g., a reference tetrahedron $\hat{K}_t$, a reference hexahedron $\hat{K}_h$). First, let R_i denote the total number of vertices of K_i (i.e., $R_i = 4$ if K_i is a tetrahedron, or $R_i = 8$ if K_i is hexahedron). Next, let E_i denote the total number of edges of K_i (i.e.,

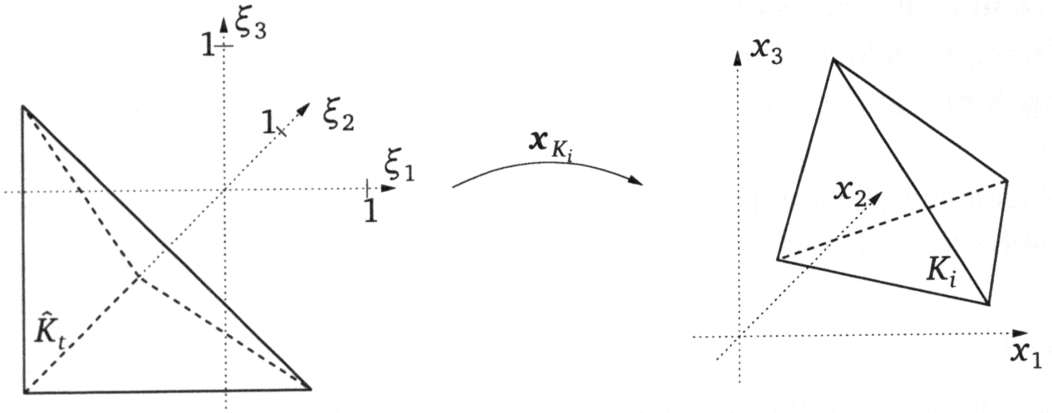


Figure 2.3: Affine mapping for a tetrahedral element K_i.

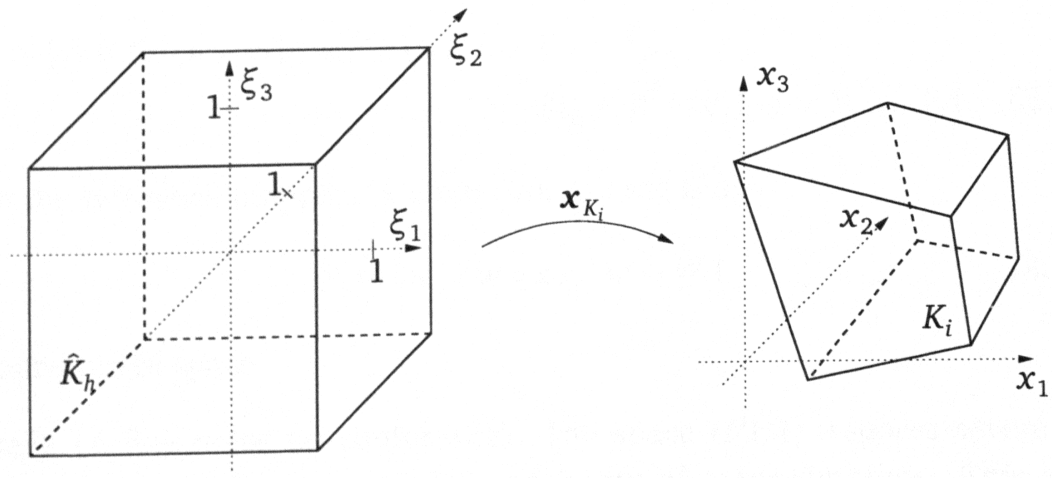


Figure 2.4: Trilinear mapping for a hexahedral element K_i.

$E_i = 6$ if K_i is a tetrahedron, or $E_i = 12$ if K_i is a hexahedron). Next, let S_i denote the number of faces of K_i (i.e., $S_i = 4$ if K_i is a tetrahedron, $S_i = 6$ if K_i is a hexahedron, etc.). Finally, let us denote the edges and faces of K_i by e_{i_j}, $j = 1, \ldots, E_i$, and s_{i_j}, $j = 1, \ldots, S_i$, respectively. We use similar notation for reference elements using accented letters $\hat{E}$, $\hat{S}$, $\hat{e}_j$, $\hat{s}_j$, etc., as seen in Figures 2.1 and 2.2.

Consider an element $K_i \in \mathcal{T}_{h,p}$ and the corresponding reference element $\hat{K}$. To every edge e_{i_j}, $j = 1, \ldots, E_i$, there is a unique reference element edge $\hat{e}_j$, such that

$$\boldsymbol{x}_{K_i}(\hat{e}_j) = e_{i_j}, \quad j = 1, \ldots, E_i. \tag{2.25}$$

Analogously, to each face s_{i_l}, $l = 1, \ldots, S_i$, we can find a reference element face $\hat{s}_l$.

It is clear that the solution $u_{h,p}$ restricted to the element K_i satisfies

$$u_{h,p}|_{K_i} \in W_i = \{w \in P^{p_i}(K_i),\ w|_{s_{i_l}} \in P^{p_{s_{i_l}}}(s_{i_l}),\ l = 1, \ldots, S_i,\\ w|_{e_{i_j}} \in P^{p_{e_{i_j}}}(e_{i_j}),\ j = 1, \ldots, E_i\}. \tag{2.26}$$

This motivates us to define the local finite element space $\hat{W}_i$,

$$\hat{W}_i = \{\hat{w} \in P^{p_i}(\hat{K}),\ \hat{w}|_{\hat{s}_l} \in P^{p_{s_{i_l}}}(\hat{s}_l),\ l = 1, \ldots, S_i,\\ \hat{w}|_{\hat{e}_j} \in P^{p_{e_{i_j}}}(\hat{e}_j),\ j = 1, \ldots, E_i\}. \tag{2.27}$$

Since the reference mapping is affine, we can see that

$$\hat{W}_i = \{\hat{w} = w \circ \boldsymbol{x}_{K_i},\ w \in W_i\} \tag{2.28}$$

is a polynomial space.

Remark (A few notes on conformity). The space $H^1(\Omega)$ imposes severe requirements on conformity – global continuity of approximation. This constraints the values of approximation at element vertices, edges and faces. Only functions that vanish entirely on the element boundary are unconstrained.

Remark. In the previous sections we have defined the *global* finite element space $V_{h,p}$ and then the corresponding *local* finite element spaces $\hat{W}_i$. Note

that in practice the process is reversed: we start with definition of local spaces and the global space is then constructed using these local spaces and reference mappings. Of course, one has to be particularly cautious to comply with the conformity requirements of the space V, i.e., we care about local and global orientations of edges and faces and we have to adjust the shape functions accordingly. Details will be presented later in this chapter.

2.2.3 Hierarchic shape functions

As mentioned in the previous remark, we need to provide the space $\hat{W}_i$ with a hierarchic basis $\hat{\mathcal{B}}_i$. This basis has the same structure as the global basis, i.e., we can distinguish between the *vertex*, *edge*, *face*, and *bubble* functions (see Figures 2.5, 2.6, 2.7, and 2.8). Basis functions on a reference element are usually referred to as *shape functions*. We construct the global finite element space using the reference mappings in such a way that if $\hat{v} \in \hat{\mathcal{B}}_i$, then for the function $v = \hat{v} \circ \boldsymbol{x}_{K_i}^{-1}$ we can find a basis function $w \in \mathcal{B}$ such that $w|_{K_i} = v$.

When defining the shape functions we will follow the standard approach taken in [62]. We construct the so-called Lobatto-based hierarchic basis of shape functions.

2.2.4 Tetrahedral reference element

For the sake of easier description of the Lobatto-based hierarchic basis, let us define the *affine coordinates* on the tetrahedral reference element.

Definition 2.3 (Affine coordinates). Let $\hat{K}_t$ be the tetrahedral reference element defined in (2.23). Affine coordinates on the tetrahedral reference element have the form

$$\begin{aligned}
\lambda_{1,t}(\xi_1,\xi_2,\xi_3) &= \frac{\xi_2+1}{2},\\
\lambda_{2,t}(\xi_1,\xi_2,\xi_3) &= -\frac{\xi_1+\xi_2+\xi_3+1}{2},\\
\lambda_{3,t}(\xi_1,\xi_2,\xi_3) &= \frac{\xi_1+1}{2},\\
\lambda_{4,t}(\xi_1,\xi_2,\xi_3) &= \frac{\xi_3+1}{2}.
\end{aligned} \tag{2.29}$$

The affine coordinates are affine functions. Each of them is equal to 1 at one vertex and zero at all the other vertices. These are actually the properties we impose on vertex functions. However, using affine coordinates we can also define the edge, face, and bubble functions, e.g., as monomial products of the affine coordinates (Peano shape functions). Unfortunately, these shape functions are renowned for their extremely bad conditioning properties. That is why we follow a more modern approach to incorporate the so-called *Lobatto polynomials* into the shape functions.

Definition 2.4 (Legendre polynomials). We define the Legendre polynomial of degree k as

$$\mathrm{L}_k(x) = \frac{1}{2^k k!} \frac{\mathrm{d}^k}{\mathrm{d}x^k}(x^2-1)^k, \quad k = 0,1,2,\ldots. \tag{2.30}$$

Definition 2.5 (Lobatto functions and kernel functions). Let us define functions $l_k(x) : [-1,1] \to \mathbb{R}$, $k = 0,1,2,\ldots$,

$$\begin{aligned} l_0(x) &= \frac{1-x}{2}, \quad l_1(x) = \frac{x+1}{2}, \\ l_k(x) &= \frac{1}{\|\mathrm{L}_{k-1}\|_2} \int_{-1}^{x} \mathrm{L}_{k-1}(\xi)\ \mathrm{d}\xi, \quad 2 \le k. \end{aligned} \tag{2.31}$$

Since all functions $l_k(x)$, $2 \le k$, vanish at ± 1, we can define *kernel functions* ϕ_j, $j = 0,1,2,\ldots$, by decomposing l_k into

$$l_k(x) = l_0(x) l_1(x) \phi_{k-2}(x), \quad k = 2,3,\ldots. \tag{2.32}$$

Remark. The idea of incorporating the Lobatto polynomials into the shape functions is motivated mainly by the fact that the set of Legendre polynomials forms an orthonormal basis of the space $L^2(-1,1)$ and thus the Lobatto polynomials have excellent properties for the discretization of the Laplace operator in 1D (see [62]).

Remark. Although not necessary, it is very convenient for algorithmic purposes to incorporate functions that satisfy certain conditions. The most important of them is that the functions incorporated are either symmetric or antisymmetric with respect to the interval midpoint where the function is defined.

Looking briefly at the definition of l_k, $k = 2,3,4,\ldots$, we can see that l_k is symmetric (i.e. $l_k(-x) = l_k(x)$, $x = [-1,1]$) with respect to zero, the definition interval midpoint, if k is even, and it is antisymmetric (i.e. $l_k(-x) = -l_k(x)$, $x = [-1,1]$) with respect to zero, if k is odd. We will exploit these properties in the following sections.

For the tetrahedral reference geometry we consider one local internal order of approximation p_b, local polynomial orders $p_{\hat{s}_j}$, $j = 1,\ldots,4$, associated with each face $\hat{s}_j$, and local orders $p_{\hat{e}_j}$, $j = 1,\ldots,4$, associated with each edge $\hat{e}_j$. These nonuniform orders of approximation must be compatible with the minimum rule. The local polynomial orders of approximation then suggest that the reference element will be equipped with the local finite element space

$$\hat{W} = \{\hat{v} \in P^{p_b}(\hat{K}_t), \hat{v}|_{\hat{s}_i} \in P^{p_{\hat{s}_i}}(\hat{s}_i), \hat{v}|_{\hat{e}_j} \in P^{p_{\hat{e}_j}}(\hat{e}_j), \\ i = 1,\ldots,4,\ j = 1,\ldots,6\}, \quad (2.33)$$

where

$$P^p(\hat{K}_t) = \operatorname{span}\{\xi_1^i \xi_2^j \xi_3^k,\ (\xi_1,\xi_2,\xi_3) \in \hat{K}_t, \\ i,j,k = 0,\ldots,p,\ i+j+k \le p\}. \quad (2.34)$$

As we have already mentioned, the choice of basis is completely arbitrary with respect to the quality of the finite element solution. In what follows we will define the standard choice of such a basis: the Lobatto-based shape functions. Recall that a hierarchic basis of $\hat{W}_i$ consists of *vertex*, *edge*, *face*, and *bubble* functions.

Vertex functions $\hat{\varphi}_t^{\hat{v}_1},\ldots,\hat{\varphi}_t^{\hat{v}_4}$ are associated with the vertices $\hat{v}_1,\ldots,\hat{v}_4$. The function $\hat{\varphi}_t^{\hat{v}_j}$ is equal to one at $\hat{v}_j$ and vanishes at $\hat{v}_i$, $i \neq j$. These functions are chosen affine in the form

$$\hat{\varphi}_t^{\hat{v}_j} = \lambda_{k_j,t}, \quad j = 1,\ldots,4. \quad (2.35)$$

The index k_j is chosen in such a way that $\hat{s}_{k_j}$ is the only face not containing vertex v_j. An example of a vertex function is depicted in Figure 2.5.

Edge functions $\hat{\varphi}_t^{\hat{e}_j,k}$, $k = 2,\ldots,p_{\hat{e}_j}$, $j = 1,\ldots,6$ appear in the basis of $\hat{W}_i$ if $2 \le p_{\hat{e}_j}$. They are constructed so that their traces on edges match the Lobatto

shape functions $l_2, \ldots, l_{p_{\hat{e}_j}}$ on edge $\hat{e}_j$ and vanish on the five remaining edges. For reasons to be explained below, we need to incorporate orientation into the definition of edge functions. Let us consider an oriented edge $\hat{e}_j = \hat{v}_{i_1}\hat{v}_{i_2}$. By s_{j_1} and s_{j_2} denote the faces that share with the edge $\hat{e}_j$ a single vertex $\hat{v}_{i_1}$ and $\hat{v}_{i_2}$ respectively. We define

$$\hat{\varphi}_t^{\hat{e}_j,k} = \lambda_{j_1,t}\lambda_{j_2,t}\phi_{k-2}(\lambda_{j_1,t} - \lambda_{j_1,t}), \quad 2 \leq k \leq p_{\hat{e}_j}. \tag{2.36}$$

An example of a higher-order Lobatto-based edge function is shown in Figure 2.6.

Face functions $\hat{\varphi}_t^{\hat{s}_j,n_1,n_2}$ are part of the basis if $3 \leq p_{\hat{s}_j}$. They are constructed so that they have nonzero traces of polynomial orders $3 \leq p_{\hat{s}_j}$ on $\hat{s}_j$ and vanish on the remaining faces. Again, we need to define a unique orientation on each face. Let us consider an oriented face $\hat{s}_j = \hat{v}_{i_1}\hat{v}_{i_2}\hat{v}_{i_3}$ in such a way that $\hat{v}_{i_1}$ and $\hat{v}_{i_3}$ have the lowest and highest local index, respectively. This also means that for each face $\hat{s}_j$ we have three affine coordinates $\lambda_{k_1,t}, \lambda_{k_2,t}, \lambda_{k_3,t}$ such that $\lambda_{k_1,t}(\hat{v}_{i_1}) = \lambda_{k_2,t}(\hat{v}_{i_2}) = \lambda_{k_3,t}(\hat{v}_{i_3}) = 1$. There are $(p_{\hat{s}_j} - 1)(p_{\hat{s}_j} - 2)/2$ oriented face functions and we can define them as

$$\hat{\varphi}_t^{\hat{s}_j,n_1,n_2} = \lambda_{k_1,t}\lambda_{k_2,t}\lambda_{k_3,t}\phi_{n_1-1}(\lambda_{k_2,t} - \lambda_{k_1,t})\phi_{n_2-1}(\lambda_{k_1,t} - \lambda_{k_3,t}),$$
$$1 \leq n_1, n_2;\ n_1 + n_2 \leq p_{\hat{s}_j}. \tag{2.37}$$

See Figure 2.7 for an illustration of a Lobatto-based higher-order face function.

Bubble functions $\hat{\varphi}_t^{b,n_1,n_2,n_3}$, vanishing on the whole boundary of $\hat{K}$, appear in the basis of $\hat{W}_i$ if $4 \leq p_b$. Let us define $(p_b - 1)(p_b - 2)(p_b - 3)/6$ bubble functions as

$$\hat{\varphi}_t^{b,n_1,n_2,n_3} = \phi_{n_1-1}(\lambda_{1,t} - \lambda_{2,t})\phi_{n_2-1}(\lambda_{3,t} - \lambda_{2,t})\phi_{n_3-1}(\lambda_{4,t} - \lambda_{2,t})\prod_{i=1}^{4}\lambda_{i,t}, \tag{2.38}$$

where $1 \leq n_1, n_2, n_3;\ n_1 + n_2 + n_3 \leq p_b - 1$. An example of a tetrahedral Lobatto-based bubble function is shown in Figure 2.8.

Proposition 2.1. *Shape functions* (2.35), (2.36), (2.37), *and* (2.38) *constitute a hierarchic basis of* $\hat{W}$.

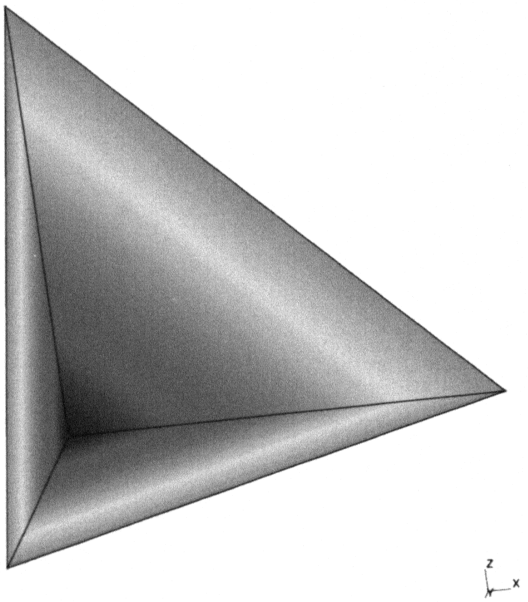


Figure 2.5: Vertex shape function on the reference tetrahedron.

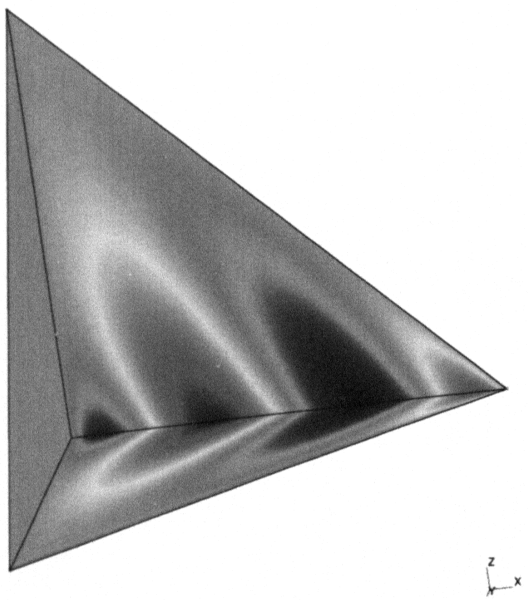


Figure 2.6: Edge shape function on the reference tetrahedron.

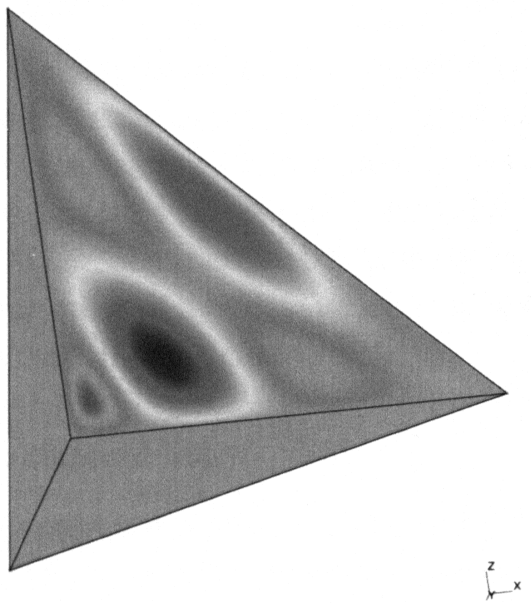


Figure 2.7: Face shape function on the reference tetrahedron.

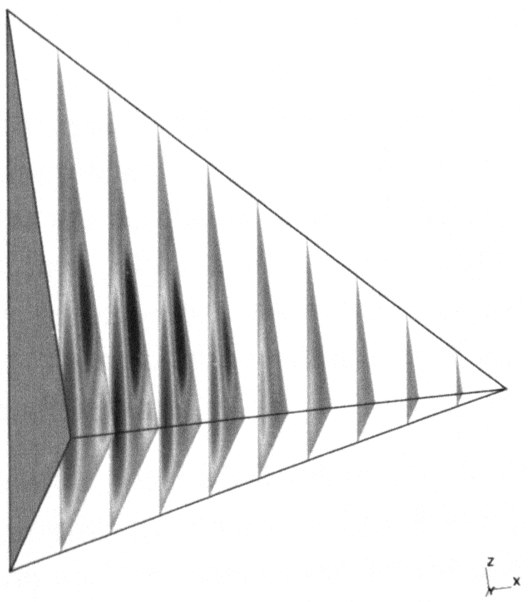


Figure 2.8: Bubble shape function on the reference tetrahedron.

Proof can be found, e.g., in [62].

Remark. Note that as the polynomial degree on elements increases, the number of vertex functions stays the same, the number of edge functions grows linearly, the number of face functions grows quadratically, and the number of bubble functions grows cubically. Thus, when the polynomial degree on elements grows, the bubble functions outnumber other types of functions quickly. Using the fact that the support of a bubble function is one single element, we will exploit this fact in the following chapters.

2.2.5 Hexahedral reference element

The geometry of the hexahedral reference element $\hat{K}_h$ defined in (2.24) imposes a different approach to the definition of the reference element basis than in the case of tetrahedra. The geometry is formed by a Cartesian product of 1D intervals which is convenient for us, since it respects the interval definition of Legendre and Lobatto polynomials defined in the previous section.

To allow for anisotropic p-refinement of hexahedral elements, we consider local directional orders of approximation $p_{b,1}$, $p_{b,2}$ and $p_{b,3}$ in the element interior, where each $p_{b,i}$ specifies the internal polynomial order of approximation in the direction of ξ_i, $i = 1,2,3$. Moreover, there is a possibility of anisotropic p-refinement of quadrilateral faces, for which we need to assign two local directional orders of approximation $p_{\hat{s}_j,1}, p_{\hat{s}_j,2}$ for each face $\hat{s}_j$, $j = 1,\ldots,6$. These directional orders are associated with a local two-dimensional system of coordinates on each face, which matches an appropriate pair of global coordinate axes. Edges $\hat{e}_j$ are equipped with local orders of approximation $p_{\hat{e}_j}$, $j = 1,\ldots,12$, as usual. These nonuniform orders of approximation must be compatible with the minimum rule.

Remark (Minimum rule for hexahedral elements)**.** In 3D the minimum rule limits the order of approximation on both edges and faces. Local directional orders of approximation on faces cannot exceed the minimum of the appropriate directional orders of approximation associated with the interior of the adjacent elements. Local order of approximation on an edge cannot exceed the minimum of appropriate directional orders corresponding to faces sharing the edge.

The local polynomial orders dictate the finite element space associated to $\hat{K}_h$ to have the form

$$\hat{W} = \{\hat{v} \in P^{p_{b,1},p_{b,2},p_{b,3}}(\hat{K}_h), \hat{v}|_{\hat{s}_i} \in P^{p_{\hat{s}_i,1},p_{\hat{s}_i,2}}(\hat{s}_i), \hat{v}|_{\hat{e}_j} \in P^{p_{\hat{e}_j}}(\hat{e}_j), \\ i = 1,\dots,6,\ j = 1,\dots,12\}, \quad (2.39)$$

where

$$P^{p_1,p_2,p_3}(\hat{K}_h) = \text{span}\{\xi_1^i \xi_2^j \xi_3^k,\ (\xi_1,\xi_2,\xi_3) \in \hat{K}_h, \\ i = 0,\dots,p_1, j = 0,\dots,p_2, k = 0,\dots,p_3\}. \quad (2.40)$$

The hierarchic basis of $\hat{W}$ again consists of *vertex*, *edge*, *face*, and *bubble* functions.

Vertex functions $\hat{\varphi}_h^{\hat{v}_1},\dots,\hat{\varphi}_h^{\hat{v}_8}$ are associated with the vertices $\hat{v}_1,\dots,\hat{v}_8$. The function $\hat{\varphi}_h^{\hat{v}_j}$ is equal to one at $\hat{v}_j$ and vanishes at $\hat{v}_i$, $i \neq j$. These functions are chosen trilinear in the form

$$\hat{\varphi}_h^{\hat{v}_j} = l_{d_1}(\xi_1)\, l_{d_2}(\xi_2)\, l_{d_3}(\xi_3), \quad (2.41)$$

where the functions $l_i(x)$ were defined in (2.31). The components of the vector index

$$\boldsymbol{d} = (d_1, d_2, d_3) \quad (2.42)$$

are related to the axial directions ξ_1, ξ_2 and ξ_3 in the following way: Let $\hat{e}_{j_1}$, $\hat{e}_{j_2}$ and $\hat{e}_{j_3}$ be the edges of $\hat{K}_h$ that share the vertex $\hat{v}_j$. We put $d_k = 0$ if $\hat{v}_j$ lies on the left of edge $\hat{e}_{j_k}$ (with respect to the axial direction ξ_k), and $d_k = 1$ otherwise. Such a vertex function is equal to one at vertex $\hat{v}_j$ and vanishes at other vertices. The trace of this function along the edges of the hexahedron $\hat{K}_h$ is a linear function. See Figure 2.9 for an example of a hexahedral vertex shape function.

Edge functions $\hat{\varphi}_h^{\hat{e}_j,k}$, $j = 1,\dots,12$, $k = 2,\dots,p_{\hat{e}_j}$, are constructed in such a way that the traces of $\hat{\varphi}_h^{\hat{e}_j,k}$ along the edge $\hat{e}_j$ match the Lobatto functions $l_2,\dots,l_{p_{\hat{e}_j}}$ and vanish on all remaining edges. Again, let $\boldsymbol{d} = (d_1, d_2, d_3)$ be a multiindex defined as follows: Put $d_m = k$, where ξ_m is the axis parallel to $\hat{e}_j$. The remaining two indices are set either to zero or one, depending on whether the edge lies on the left or right side of the reference hexahedron, with respect

to the remaining axial directions. An edge function of order k is then defined as

$$\hat{\varphi}_h^{\hat{e}_j,k} = l_{d_1}(\xi_1)\, l_{d_2}(\xi_2)\, l_{d_3}(\xi_3). \tag{2.43}$$

A higher-order hexahedral edge function is illustrated in Figure 2.10.

Face functions $\hat{\varphi}_h^{\hat{s}_j,n_1,n_2}$, $j = 1,\ldots,6$, $n_1 = 2,\ldots,p_{\hat{s}_j,1}$, $n_2 = 2,\ldots,p_{\hat{s}_j,2}$, corresponding to a face $\hat{s}_j$, $j = 1,\ldots,6$, are constructed to have a trace of directional orders $p_{\hat{s}_j,1}$ and $p_{\hat{s}_j,2}$ on the face $\hat{s}_j$, with respect to its local coordinate system specified above. They vanish on all the remaining faces. The components of the multiindex $\boldsymbol{d} = (d_1, d_2, d_3)$ corresponding to the local coordinate system are now set to n_1 and n_2, respectively. The remaining index is again set either to zero or one, depending on whether the face $\hat{s}_j$ lies on the left or right side of the reference hexahedron, with respect to the remaining axial direction. We then define

$$\hat{\varphi}_h^{\hat{s}_j,n_1,n_2} = l_{d_1}(\xi_1)\, l_{d_2}(\xi_2)\, l_{d_3}(\xi_3). \tag{2.44}$$

An example of such a hexahedral Lobatto-based face shape function is shown in Figure 2.11.

Recall the local directional orders of approximation $p_{b,1}$, $p_{b,2}$ and $p_{b,3}$ in the element interior. Bubble functions $\hat{\varphi}^{b,n_1,n_2,n_3}$, $n_1 = 2,\ldots,p_{b,1}$, $n_2 = 2,\ldots,p_{b,2}$, $n_3 = 2,\ldots,p_{b,3}$ vanish on the whole boundary of $\hat{K}_h$ and form the rest of the hierarchic basis of $\hat{W}$. We define

$$\hat{\varphi}_h^{b,n_1,n_2,n_3} = l_{n_1}(\xi_1)\, l_{n_2}(\xi_2)\, l_{n_3}(\xi_3). \tag{2.45}$$

Hexahedral Lobatto-based bubble function is depicted in Figure 2.12.

Remark. Again, it is easy to see that as the polynomial degree on elements increases, the number of vertex functions stays the same, the number of edge functions grows linearly, the number of face functions grows quadratically, and the number of bubble functions grows cubically. Hence, bubble functions quickly outnumber other types of shape functions as the polynomial degree associated to the element increases.

2.2.6 Construction of reference mappings

When dealing with noncurvilinear elements the construction of the reference mapping $\boldsymbol{x}_{K_i} : \hat{K} \to K_i$ is straightforward. The mathematical form of the map-

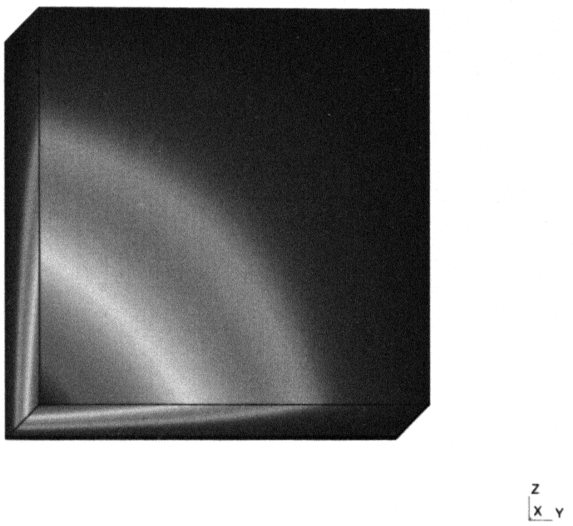


Figure 2.9: Vertex shape function on the reference hexahedron.

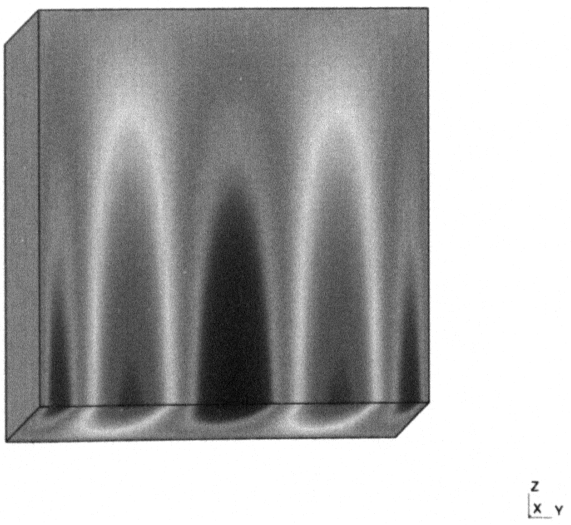


Figure 2.10: Edge shape function on the reference hexahedron.

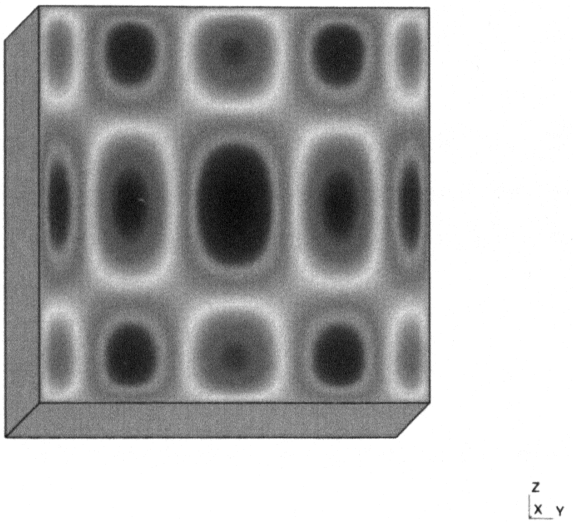


Figure 2.11: Face shape function on the reference hexahedron.

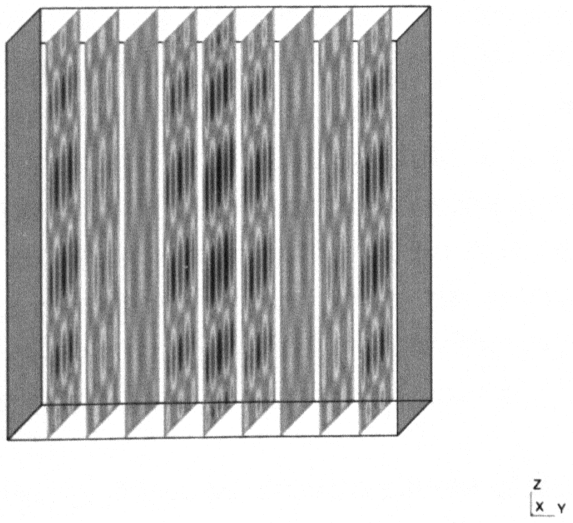


Figure 2.12: Bubble shape function on the reference hexahedron.

ping is usually sought in the form

$$\boldsymbol{x}_{K_i}(\xi) = \sum_{j=1}^{R_i} \boldsymbol{\alpha}_{K_i}^{\hat{v}_j} \, \hat{\varphi}^{\hat{v}_j}(\xi), \tag{2.46}$$

where $\hat{\varphi}^{\hat{v}_j}$ is a vertex function belonging to the vertex $\hat{v}_j$. What remains to be found are the coefficients $\boldsymbol{\alpha}_{K_i}^{\hat{v}_j}$. If we realize that the vertex functions $\hat{\varphi}^{\hat{v}_j}$, $j = 1, 2, \ldots, R_i$, attain the value of one at v_j and zero at all other vertices, it is easy to verify that the choice of

$$\boldsymbol{\alpha}_{K_i}^{\hat{v}_j} := \boldsymbol{x}_{i,j}, \quad j = 1, 2, \ldots, R_i, \tag{2.47}$$

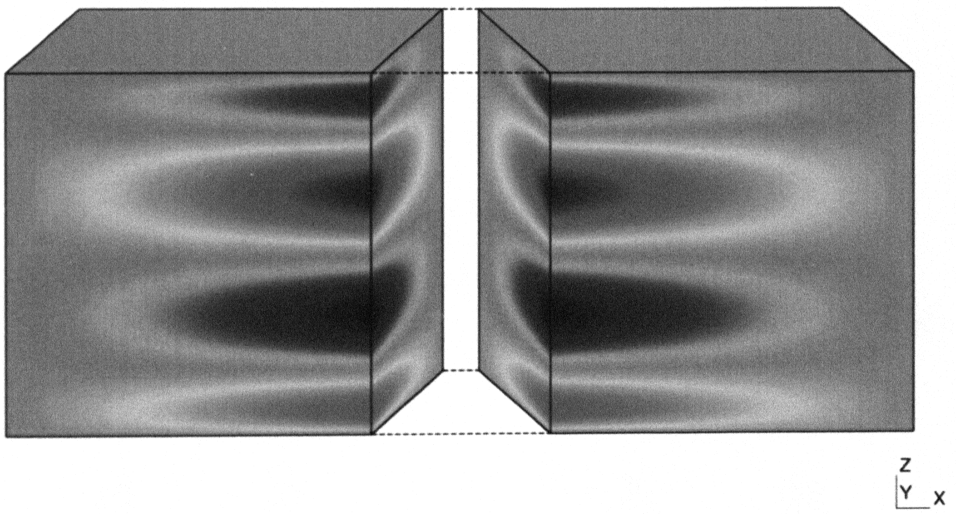
where $\boldsymbol{x}_{i,1}, \boldsymbol{x}_{i,2}, \ldots, \boldsymbol{x}_{i,R_i}$ are coordinates of vertices of K_i, leads to the desired mapping.

Remark. The form of (2.46) is advantageous for several reasons, the most important of them being easy generalization of such a mapping for isoparametric (curvilinear) elements.

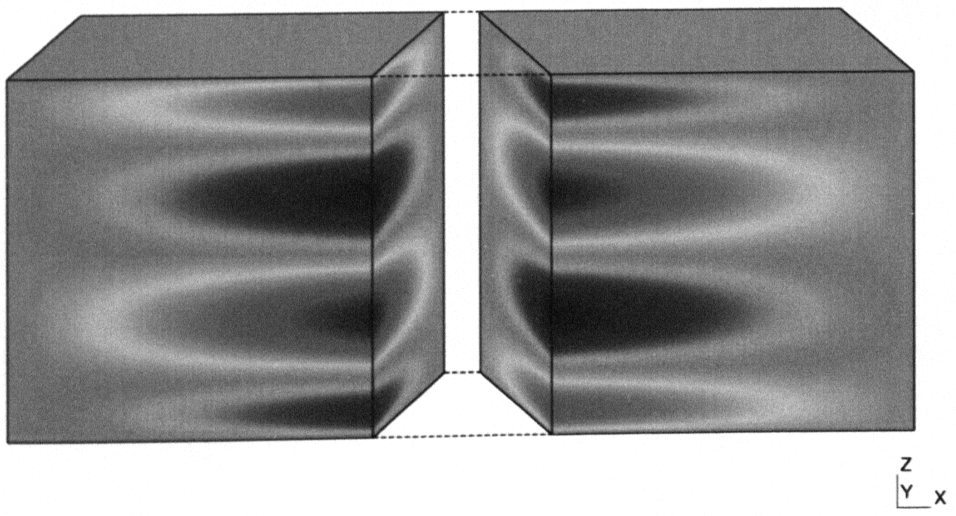
2.2.7 Orientation of reference element edge and face functions

When defining the shape functions on reference domains we equipped their edges and faces with unique orientations in order to make the definitions of reference element edge and face functions unique. In the same way one has to assign unique global orientations to edges and faces in the mesh $\mathcal{T}_{h,p}$ in order to ensure the uniqueness of basis functions of the space $V_{h,p}$. Let $\boldsymbol{x}_{K_i}$ be a bijective reference mapping corresponding to an element $K_i \in \mathcal{T}_{h,p}$, and let $\hat{K}$ be the appropriate reference domain. Since the reference and global orientations have been chosen independently, the orientations of the edges and faces of the geometrically identical domains K_i and $\boldsymbol{x}_{K_i}(\hat{K})$ are generally mismatched.

We need to resolve this problem in order to ensure global conformity of edge and face functions. For each element $K_i \in \mathcal{T}_{h,p}$ one has to adjust the basis of the reference element polynomial space in an algorithmically simple way, such that the space itself stays unchanged. We illustrate such a problem in Figures 2.13 and 2.14.



(a)



(b)

Figure 2.13: Two adjacent mesh hexahedral elements and possible configurations resulting from transformation of reference element edge functions using reference mappings to obtain a global edge function defined on the two elements.

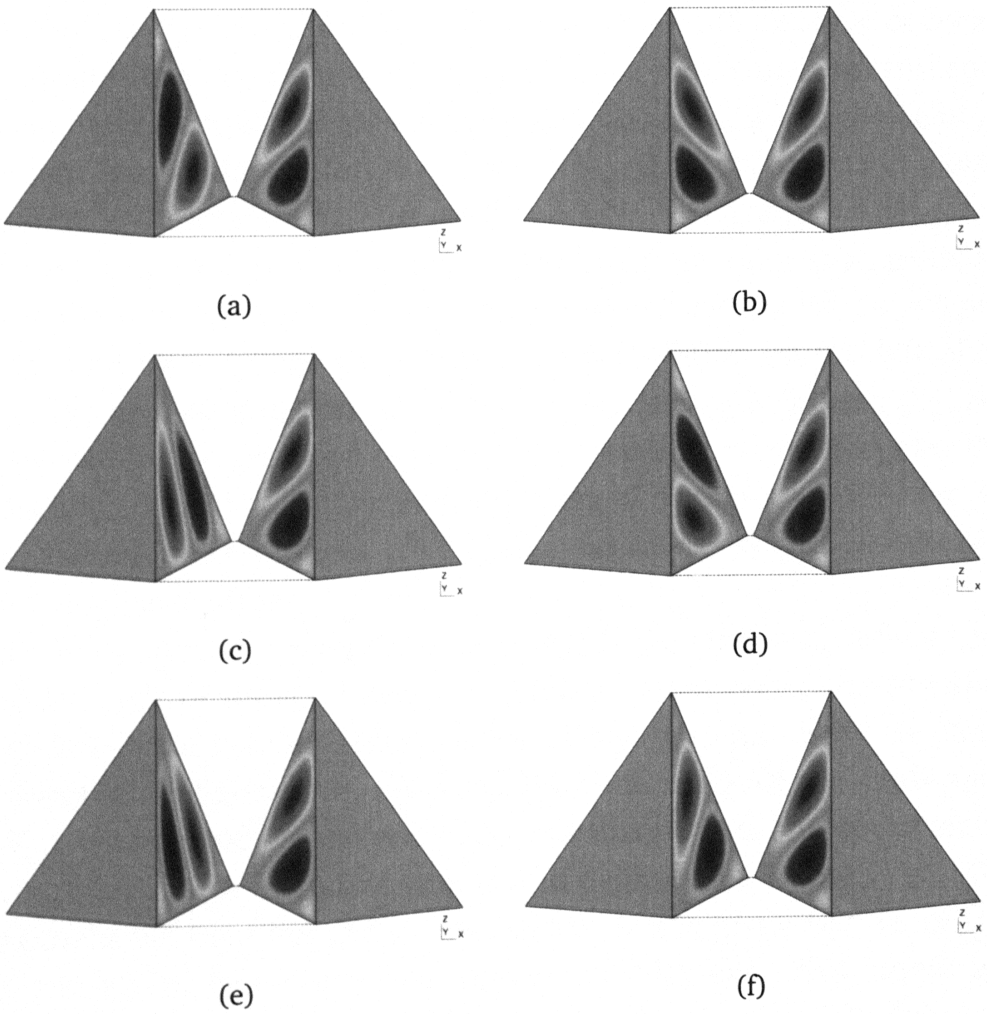


Figure 2.14: Two adjacent mesh tetrahedral elements and possible configurations resulting from transformation of reference element face functions using reference mappings to obtain a global face function defined on the two elements.

First, we present two neighboring elements with an edge function on each of them. By "gluing" these two parts together we get a global edge function defined on these two elements. Note that to define the edge function in its whole support we would need to present the edge function on all mesh elements sharing the edge. As we said before, the configuration of local and global orientations is generally arbitrary, thus we may obtain several configurations on these elements. If the configuration obtained by our choice of global orientations breaks the conformity of space $V_{h,p}$ (i.e., the resulting global approximation is discontinuous), we need to adjust the corresponding reference edge function on one of the elements. We illustrate the situation in Figure 2.13. On the right element we keep the global orientation on the given edge constant, while we change the global edge orientation on the left element. The two arising configurations are shown in Figures 2.13 (a) and (b), respectively. In case (a) we get a conforming approximation, whereas in case (b) the resulting approximation is nonconforming, and thus there is a need for adjustment in the reference element basis. These adjustments will be described in the following paragraphs.

Next, we show two neighboring tetrahedral elements with a face function defined on each of them. Again, by "gluing" these two parts together we get a global face function defined in its whole support. Now we get even more possible configurations. To keep the conformity of the finite element space $V_{h,p}$, we must adjust the reference face function on one of the elements. Again, an illustration of such a problem is depicted in Figure 2.14. On the right element we keep the global orientation on the given face constant and change the global face orientation on the left element. Now there are six possible configurations shown in Figures 2.14 (a), (b), (c), (d), (e), and (f). Now the only conforming case is (b), meaning that in the other cases we get a nonconforming approximation and an adjustment in the reference element basis must be made.

Now let us have a look at the problem from a more theoretical point of view. Recall that $K_i \in \mathcal{T}_{h,p}$ is a mesh element, $\hat{K}$ an appropriate reference domain, and $\boldsymbol{x}_{K_i} : \hat{K} \to K_i$ a bijective reference mapping. At the beginning of this section we explained that the orientation of the reference element edge and face functions has to be adjusted in order to compensate for the difference

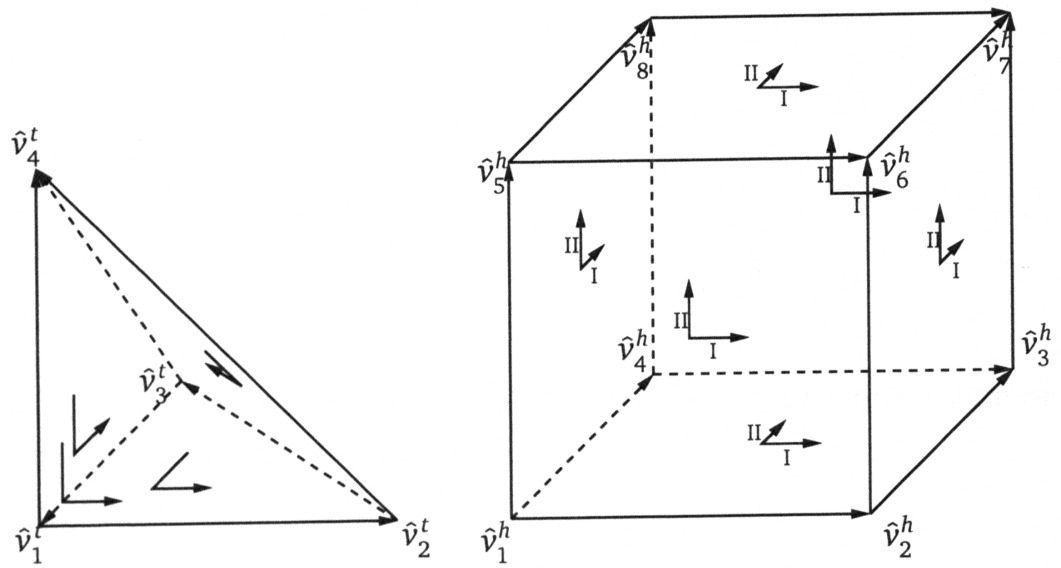


Figure 2.15: Local orientation of faces of the reference domains $\hat{K}_t$ (left) and $\hat{K}_h$ (right).

between the unique orientation of edges and faces of the element K_i, and the unique orientation of edges and faces of the the transformed reference domain $\boldsymbol{x}_{K_i}(\hat{K})$.

Local orientation of edges for the reference domains $\hat{K}_t$ and $\hat{K}_h$ can be found in Figures 2.1 and 2.2, respectively. Local orientations of faces of $\hat{K}_t$ and $\hat{K}_h$, which were defined in Sections 2.2.4 and 2.2.5, are depicted in Figure 2.15.

Each edge and face in the mesh is assigned a unique orientation as well. These global orientations can be retrieved from the unique global enumeration of vertices. For example, edges can be oriented according to the enumeration of their vertices in increasing order, as shown in Figure 2.16.

For each quadrilateral face we select its vertex A with the lowest index and two edges AB and AC such that $\text{index}(A) < \text{index}(B) < \text{index}(C)$. Triangular faces are oriented the same way, except that they do not have the product form of the quadrilateral ones, as illustrated in Figure 2.17

In the previous paragraphs we were describing the need for an adjustment in the reference element basis in order to keep the global conformity of the resulting finite element space. In the following text we describe the concrete measures that need to be taken to carry out these adjustments.

Figure 2.16: Global orientation of mesh edges based on a unique enumeration of vertices (here index(A) < index(B)).

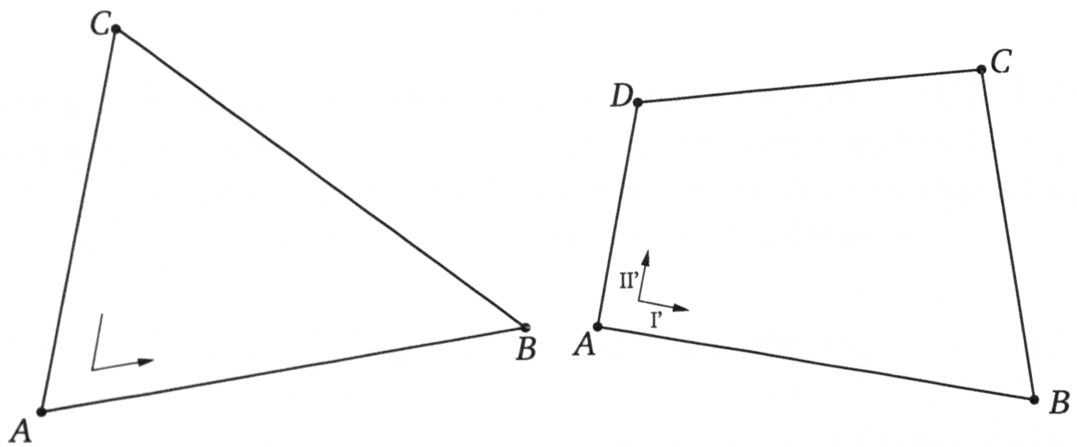


Figure 2.17: Global orientation of triangular and quadrilateral faces, index(A) < index(B) < index(C), and A has the lowest index among all vertices of the face.

Transformation of reference element edge functions

Consider an oriented edge $e = v_k v_l \subset \partial K_i$, $K_i \in \mathcal{T}_{h,p}$, reference domain $\hat{K}$ such that $\boldsymbol{x}_{K_i}(\hat{K}) = K_i$, and oriented edge $\hat{e} = \hat{v}_k \hat{v}_l \subset \partial \hat{K}$ such that $\boldsymbol{x}_{K_i}(\hat{e}) = e$, up to the orientation. The edge $\hat{e}$ needs to be equipped with an orientation flag $o(\hat{e}) = \pm 1$ that indicates whether its image $\boldsymbol{x}_{K_i}(\hat{e})$ has the same or opposite orientation with respect to the oriented edge e. In other words,

$$o(\hat{e}) = \begin{cases} 1 & \text{if } \boldsymbol{x}_{K_i}(\hat{v}_k) = v_k,\ \boldsymbol{x}_{K_i}(\hat{v}_l) = v_l, \\ -1 & \text{if } \boldsymbol{x}_{K_i}(\hat{v}_k) = v_l,\ \boldsymbol{x}_{K_i}(\hat{v}_l) = v_k. \end{cases} \tag{2.48}$$

If $o(\hat{e}) = 1$ then all reference element edge functions associated with e unchanged. Otherwise we have to transform them using a suitable bijective mapping $\boldsymbol{x}^o_{\hat{e}} : \hat{K} \to \hat{K}$, which inverts the parametrization of the edge $\hat{e}$. This can be carried out easily using the reference element $\hat{K}$ and its identical mapping $\boldsymbol{I}_{\hat{K}} : \hat{K} \to \hat{K}$, which can be expressed as a linear combination of vertex shape functions $\hat{\varphi}^{\hat{v}_j}$ and vertex coordinates $\hat{v}_j$,

$$\boldsymbol{I}_{\hat{K}}(\xi) = \sum_{j=1}^{\hat{R}} \hat{\varphi}^{\hat{v}_j}(\xi)\, \hat{v}_j, \tag{2.49}$$

where $\hat{R}$ is the number of vertices of the reference element $\hat{K}$. Note that the mapping $\boldsymbol{I}_{\hat{K}}$ is defined in the same manner as reference mappings $\boldsymbol{x}_{K_i}$ in Section 2.2.6. We arrive at $x^o_{\hat{e}}$ by switching the vertex functions corresponding to the vertices $\hat{v}_k$, $\hat{v}_l$ in (2.49), hence the mapping $\boldsymbol{x}^o_{\hat{e}}$ has the form

$$\boldsymbol{x}^o_{\hat{e}} = \sum_{j=1, j\neq k,l}^{\hat{R}} \hat{\varphi}^{\hat{v}_j}(\xi)\, \hat{v}_j + \hat{\varphi}^{\hat{v}_k}(\xi)\, \hat{v}_l + \hat{\varphi}^{\hat{v}_l}(\xi)\, \hat{v}_k. \tag{2.50}$$

When we further investigate the properties of edge functions defined in (2.36) and (2.43) (recall that they were defined so that their trace on the corresponding edge matches the Lobatto functions and these are symmetric or antisymmetric with respect to the edge midpoint) and the mapping (2.50) we come to the conclusion that the switch causes only a change of sign for odd degree edge functions. In other words, we get

$$\hat{\varphi}^{\hat{e},m} \circ \boldsymbol{x}^o_{\hat{e}} = o^m(\hat{e}) \hat{\varphi}^{\hat{e},m}, \quad m = 2, 3, \ldots. \tag{2.51}$$

Transformation of reference element triangular face functions

The situation is more delicate when dealing with triangular faces, since the face functions are not invariant with respect to the enumeration of vertices. Recall that for both the reference domain and physical mesh faces we selected the vertex with lowest index that plays a special role in the definition of their orientations.

Consider an oriented triangular mesh face $s = v_j v_k v_l \subset \partial K_i$, $K_i \in \mathcal{T}_{h,p}$ such that index(j) < index(k) < index(l), a reference domain $\hat{K}$ such that $K_i = \boldsymbol{x}_{K_i}(\hat{K})$, and an oriented face $\hat{s} = \hat{v}_A \hat{v}_B \hat{v}_C \subset \partial \hat{K}$ such that $\boldsymbol{x}_{K_i}(\hat{s}) = s$ (up to the orientation) and $A < B < C$. The six different situations that may occur are illustrated in Figure 2.18.

We may now equip the face $\hat{s}$ with two orientation flags $o_1(\hat{s}) = \{0, 1, 2\}$ and $o_2(\hat{s}) = \pm 1$ indicating the position of $\boldsymbol{x}_{K_i}(\hat{v}_A)$ with respect to v_j and compatibility of orientations of the faces $\hat{s}$ and s, respectively. The easiest case occurs when $o_1(\hat{s}) = 0$ and $o_2(\hat{s}) = 1$: the global orientation of the mesh face s and the transformed local orientation of the face $\boldsymbol{x}_{K_i}(\hat{s})$ coincide and hence all reference element face functions associated with the face $\hat{s}$ stay unchanged. In all other cases, analogously as before, we have to transform them by means of a suitable bijective mapping $\boldsymbol{x}^o_{\hat{s}} : \hat{K} \to \hat{K}$. The identical mapping $I_{\hat{K}_t}(\xi) : \hat{K}_t \to \hat{K}_t$ can be written in a similar way as in (2.49). We select three barycentric coordinates λ_A, λ_B, and λ_C such that $\lambda_A(\hat{v}_A) = \lambda_B(\hat{v}_B) = \lambda_C(\hat{v}_C) = 1$. If $o_2(\hat{s}) = 1$ and $o_1(\hat{s}) = 1$ or 2, the mapping $\boldsymbol{x}^o_{\hat{s}}$ cyclically rotates the vertices $\hat{v}_A$, $\hat{v}_B$, and $\hat{v}_C$ by one or two, respectively, in the backward direction. If $o_2(\hat{s}) = -1$, in addition to the rotation of the vertices the mapping also switches the vertices $\hat{v}_B$ and $\hat{v}_C$. For example, if $o_1(\hat{s}) = 1$, the vertices $\hat{v}_B$ and $\hat{v}_C$ are switched and moreover the vertices $\hat{v}_A$, $\hat{v}_C$, and $\hat{v}_B$ are rotated by one backward. If $o_1(\hat{s}) = 2$ we need to rotate the vertices by two backward. Finally, if $o_1(\hat{s}) = 0$, just the switch is performed and no rotation needs to be done. For further details see [62]. Recall that the face functions on a triangular face are defined by means of barycentric coordinates (see (2.37)). When composing these shape functions with the mapping $\boldsymbol{x}^o_{\hat{s}}$ we only have to rotate and switch the barycentric coordinates λ_A, λ_B, and λ_C in the way described above. An example of a face function composed with the mapping $\boldsymbol{x}^o_{\hat{s}}$ for all six combinations of $o_1(\hat{s})$ and $o_2(\hat{s})$ is depicted in Figure 2.19.

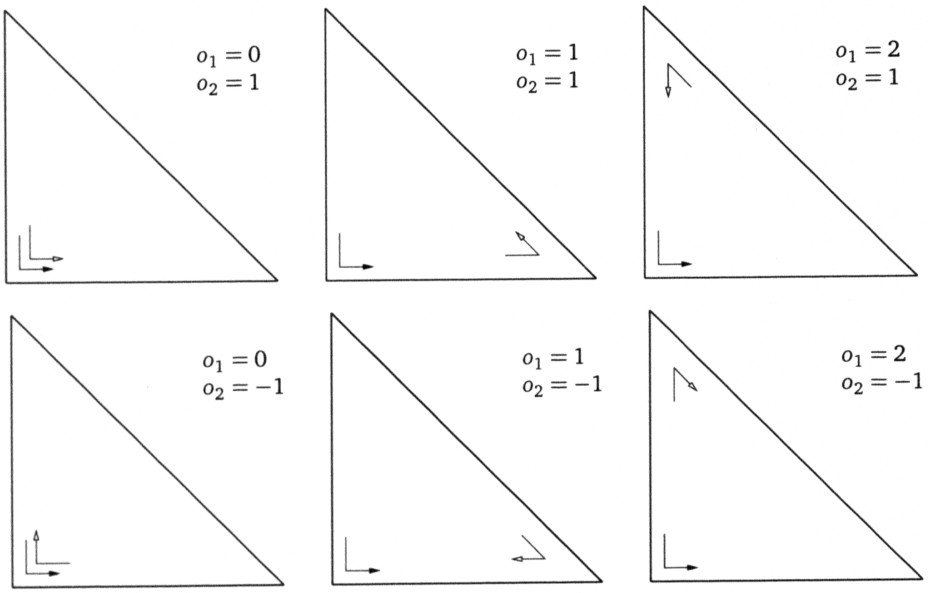


Figure 2.18: Six possible combinations of the global orientation of the triangular mesh face s (black arrow) and the transformed local orientation of the face $\boldsymbol{x}_{K_i}(\hat{s})$.

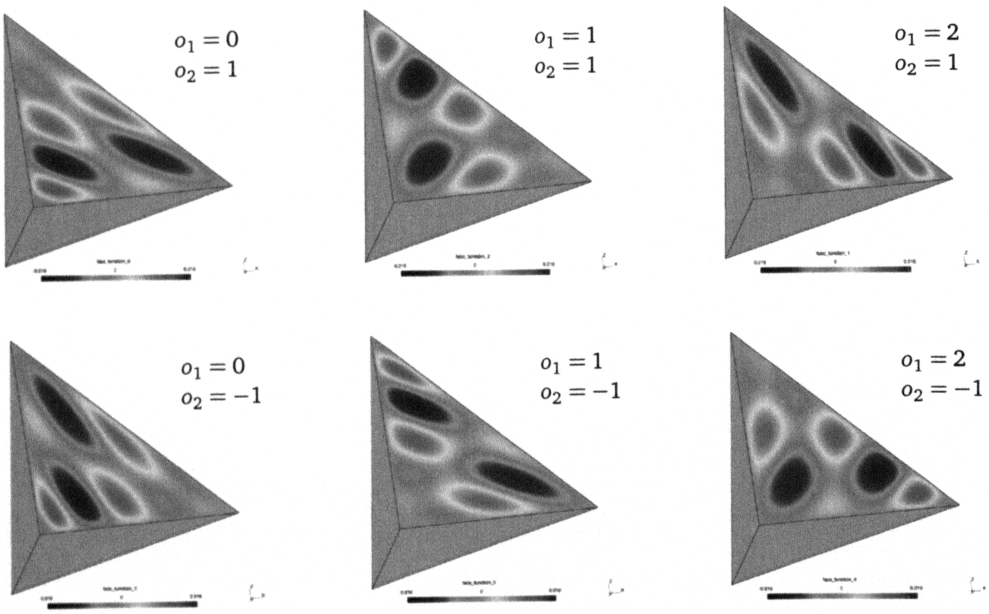


Figure 2.19: A shape function on a triangular face composed with six variants of the mapping $\boldsymbol{x}_{\hat{s}}^{o}$ corresponding to all six combinations of $o_1(\hat{s})$ and $o_2(\hat{s})$.

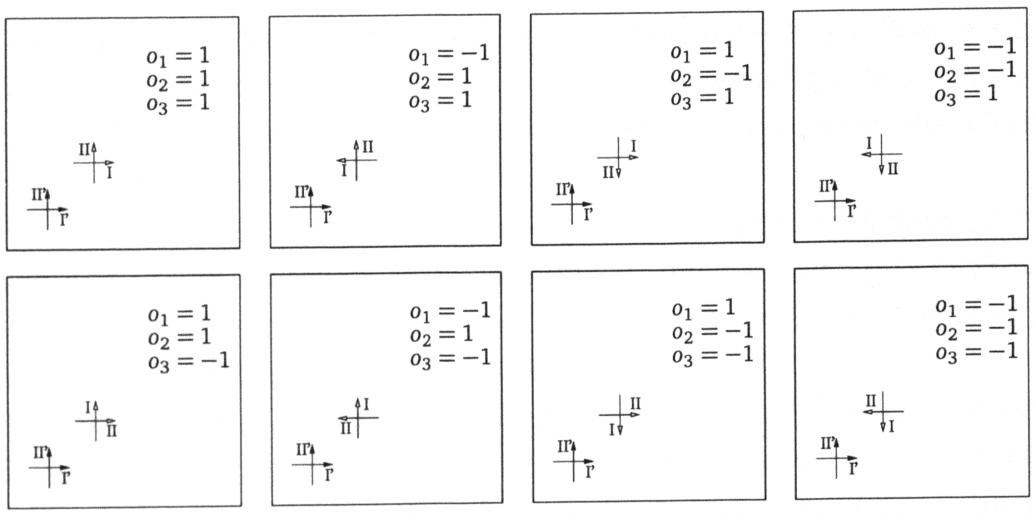


Figure 2.20: Eight possible combinations of the global directions I', II' of the quadrilateral mesh face s (black arrows) and the transformed local directions I, II of the face $\boldsymbol{x}_{K_i}(\hat{s})$.

Transformation of reference element quadrilateral face functions

It turns out that handling transformation of quadrilateral face functions is generally much easier than their triangular counterparts. The difference is that we can exploit the product geometry of quadrilateral faces. Consider an oriented quadrilateral mesh face $s = v_j v_k v_l v_m \subset \partial K_i$, $K_i \in \mathcal{T}_{h,p}$, reference domain $\hat{K}$ such that $K_i = \boldsymbol{x}_{K_i}(\hat{K})$ and oriented face $\hat{s} = \hat{v}_j \hat{v}_k \hat{v}_l \hat{v}_m \subset \partial \hat{K}$ such that $\boldsymbol{x}_{K_i}(\hat{s}) = s$ (up to the orientation). The eight different situations that may occur are depicted in Figure 2.20.

We may describe the eight situations by three orientation flags $o_n = \pm 1$, $n = 1, 2, 3$, in the following way: We put $o_3 = 1$, if the transformed local axes I, II coincide with the global axes I', II'. Otherwise, we set $o_3 = -1$. Next, if $o_3 = 1$, the flag o_1 indicates whether the transformed local direction I matches the global direction I', and the flag o_2 indicates whether the transformed local direction II matches the global direction II'. Finally, if $o_3 = -1$, the flag o_1 indicates whether the the transformed local direction I matches the global direction II', and the flag o_2 indicates if the transformed local direction II matches the global direction I'.

Clearly, if $o_1 = o_2 = o_3 = 1$ then all reference element face functions

associated with the face $\hat{s}$ stay unchanged. Otherwise, we follow a similar approach as in previous cases. We have to transform the face functions using a suitable bijective mapping $\boldsymbol{x}_{\hat{s}}^{o} : \hat{K} \to \hat{K}$. Again, we start from the identical mapping $I_{\hat{K}} : \hat{K} \to \hat{K}$ defined by (2.49).

First assume that $o_3 = 1$. If $o_1 = -1$, we obtain the transformation $\boldsymbol{x}_{\hat{s}}^{o}$ by swapping vertex functions corresponding to the pairs of vertices situated on the edges lying in the first local direction. The same situation occurs with $o_2 = -1$ and the second local direction II.

Let us return to the definition (2.43) of a Lobatto based face shape function on a hexahedral reference element $\hat{K}_h$. Due to the properties of the functions $l_k(x)$, $k = 2, 3, \ldots$, we realize that the transformation simplifies to the same ± 1 sign factors as for the edge functions above,

$$\hat{\varphi}_h^{\hat{s},n_1,n_2} \circ \boldsymbol{x}_{\hat{s}}^{o} = o_1^{n_1} o_2^{n_2} \, \hat{\varphi}_h^{\hat{s},n_1,n_2}, \quad n_1 = 2, 3, \ldots; \quad n_2 = 2, 3, \ldots, \tag{2.52}$$

if $o_3 = 1$. In the case that $o_3 = -1$ the situation is similar except that indices n_1 and n_2 in the definition of the face functions must be swapped,

$$\hat{\varphi}_h^{\hat{s},n_1,n_2} \circ \boldsymbol{x}_{\hat{s}}^{o} = o_1^{n_1} o_2^{n_2} \, \hat{\varphi}_h^{\hat{s},n_2,n_1}, \quad n_1 = 2, 3, \ldots; \quad n_2 = 2, 3, \ldots. \tag{2.53}$$

2.2.8 Transformation of reference element polynomial spaces

Before one can design basis functions of the space $V_{h,p}$, reference element polynomial spaces have to be transformed elementwise to the physical mesh $\mathcal{T}_{h,p}$. For H^1-conforming elements the situation is conventional: the mapping $\boldsymbol{\Phi}_{K_i}^{1}$ from the reference element space $\hat{W}(\hat{K})$ to the corresponding space $W(K)$ on the element K,

$$\hat{W}(\hat{K}) \xrightarrow{\boldsymbol{\Phi}_{K_i}^{1}} W(K), \tag{2.54}$$

requires that the function value of the reference element shape function $\hat{\varphi}$ at each reference point $\boldsymbol{\xi} \in \hat{K}$ coincides with the value of the transformed shape function φ at its image $\boldsymbol{x} = \boldsymbol{x}_{K_i}(\boldsymbol{\xi}) \in K_i$. Hence, it is easy to see that the transformation rule reads

$$\varphi = \boldsymbol{\Phi}_{K_i}^{1}(\hat{\varphi}) = \hat{\varphi} \circ \boldsymbol{x}_{K_i}^{-1}, \tag{2.55}$$

or, in other words,

$$\varphi(\boldsymbol{x}) = \hat{\varphi}(\xi), \text{ where } \boldsymbol{x} = \boldsymbol{x}_{K_i}(\xi). \tag{2.56}$$

For transformations of variational formulations in the space $H^1(\Omega)$ to the reference domain we also need to transform the gradient operator ∇_ξ on the reference domain to the gradient ∇_x in the physical mesh. We use the chain rule to differentiate

$$\begin{aligned} \nabla_x(\hat{\varphi} \circ \boldsymbol{x}_{K_i}^{-1})(\boldsymbol{x}) &= \left(\frac{D\boldsymbol{x}_{K_i}}{D\xi} \left(\xi|_{\xi=\boldsymbol{x}_{K_i}^{-1}(\boldsymbol{x})} \right) \right)^{-T} \nabla_\xi \hat{\varphi} \left(\xi|_{\xi=\boldsymbol{x}_{K_i}^{-1}(\boldsymbol{x})} \right) \\ &= \left[\left(\frac{D\boldsymbol{x}_{K_i}}{D\xi} \right)^{-T} \nabla_\xi \hat{\varphi} \right] \circ \boldsymbol{x}_{K_i}^{-1}(\boldsymbol{x}). \end{aligned} \tag{2.57}$$

The rule for the gradient transformation thus reads

$$\nabla_x = \left(\frac{D\boldsymbol{x}_{K_i}}{D\xi} \right)^{-T} \nabla_\xi. \tag{2.58}$$

2.2.9 Design of global basis functions

To be able to define the basis functions of the space $V_{h,p}$ we first had to solve two problems:

- First, in Section 2.2.7, for every element we adjusted the reference element edge and face functions in order to make them compatible with the global orientation of edges and faces in the physical mesh $\mathcal{T}_{h,p}$.
- Second, in Section 2.2.8, we described the transformation of reference element polynomial spaces to physical mesh elements by means of reference mappings.

Having these problems solved we can continue with the definition of the globally conforming basis functions on element patches. The conformity requirements of the of the space H^1 require to split the global basis functions in the same manner as we did for reference element shape functions. The basis functions are defined by means of the reference element shape functions and orientation adjustments $\boldsymbol{x}_{\hat{e}}^{o}$ and $\boldsymbol{x}_{\hat{s}}^{o}$.

Definition 2.6 (H^1 conforming basis functions of the space $V_{h,p}$). Let the space V be a subspace of $H^1(\Omega)$ and let $V_{h,p}$ be a finite-dimensional approximation of V on Ω.

Vertex function $\varphi^v \in V_{h,p}$, associated with a mesh vertex v, is a continuous function defined in Ω which equals to one at v and vanishes outside of the patch $S(v)$ formed by all elements that share the vertex v_k. Restricted to each element $K_i \in S(v)$,

$$\varphi^v|_{K_i} = \boldsymbol{\Phi}^1_{K_i}(\hat{\varphi}^{\hat{v}}), \tag{2.59}$$

where $\hat{v} = \boldsymbol{x}^{-1}_{K_i}(v)$ is the corresponding vertex of the reference domain, $\hat{\varphi}^{\hat{v}}$ is a reference element vertex function associated with the vertex $\hat{v}$ and the transformation $\boldsymbol{\Phi}^1_{K_i}$ was defined in (2.55).

Edge function $\varphi^e \in V_{h,p}$, associated with a mesh edge e, is a continuous function defined in Ω which vanishes outside of the patch $S(e)$ formed by all elements that share the edge e. Restricted to each element $K_i \in S(e)$,

$$\varphi^e|_{K_i} = \boldsymbol{\Phi}^1_{K_i}(\hat{\varphi}^{\hat{e}} \circ \boldsymbol{x}^o_{\hat{e}}), \tag{2.60}$$

where $\hat{e} = \boldsymbol{x}^{-1}_{K_i}(e)$ is the corresponding edge of the reference domain, $\hat{\varphi}^{\hat{e}}$ is a reference element edge function associated with the edge $\hat{e}$ and $\boldsymbol{x}^o_{\hat{e}}$ is the orientation adjustment defined in Section 2.2.7.

Face function $\varphi^s \in V_{h,p}$, associated with a mesh face s, is a continuous function defined in Ω which vanishes outside of the patch $S(s)$ formed by all elements that share the face s. Restricted to each element $K_i \in S(s)$,

$$\varphi^s|_{K_i} = \boldsymbol{\Phi}^1_{K_i}(\hat{\varphi}^{\hat{s}} \circ \boldsymbol{x}^o_{\hat{s}}), \tag{2.61}$$

where $\hat{s} = \boldsymbol{x}^{-1}_{K_i}(s)$ is the corresponding face of the reference domain, $\hat{\varphi}^{\hat{s}}$ is a reference element face function associated with the face $\hat{s}$ and $\boldsymbol{x}^o_{\hat{s}}$ is the orientation adjustment defined in Section 2.2.7.

Bubble function $\varphi^b \in V_{h,p}$, associated with an element K_i, is a continuous function defined in Ω which vanishes outside of the element K_i such that

$$\varphi^b|_{K_i} = \boldsymbol{\Phi}^1_{K_i}(\hat{\varphi}^b), \tag{2.62}$$

where $\hat{\varphi}^b$ is a reference element bubble function.

CHAPTER 3

Conditioning properties of higher-order shape functions

In the previous chapter we gave a brief introduction to the *hp*-FEM discretization technique. One of the the most important properties of general FEM is that the finite element solution does not depend on the concrete choice of the basis functions of the finite element space $V_{h,p}$. More precisely, the quality of the finite element solution only depends on the quality of discretization, but not on the set of shape functions chosen as the basis of reference elements.

As presented in Section 2.1.3, discretized finite element problem reduces to the solution of a linear problem (2.21). The main goal of *hp*-FEM is to make this system as small as possible, while retaining good approximation properties of the finite element solution. Using higher-order elements appropriately can significantly reduce the number of degrees of freedom. However, the fill in the stiffness matrix is usually much higher than in the case of low-order elements. Unfortunately, this also affects the conditioning properties. Despite the resulting linear systems are very small, the convergence of iterative solvers may be very slow and sometimes the solvers even fail to converge.

By choosing different bases of $V_{h,p}$ we do not change the error of the approximate solution, but we can highly affect the conditioning properties of the stiffness matrix. Hence, we can speed up the solution process by carefully designing the basis of $V_{h,p}$. In this chapter we address this problem and suggest several methods of conditioning optimization.

This chapter presents original results obtained by the author. This topic was thoroughly analyzed in several articles and technical reports published by our research group. Let us cite, e.g., [80, 74, 63] as the most important of them.

3.1 Motivation

As mentioned before, the finite element discretization of a linear problem results in a system of linear algebraic equations $A\mathbf{y} = \mathbf{l}$. Although the size of the matrix can be essentially reduced by a suitable application of the *hp*-FEM, the solution of the discrete problem may still pose a big computational problem. The main reasons are:

- The size of the linear problem grows as $O(h^{-3})$, where h is the mesh diameter,
- the size of the linear problem grows with used polynomial degree as $O(p^3)$.

Due to the nature of basis functions, a proper enumeration of degrees of freedom results in a block structure of the stiffness matrix (see Figure 3.1), where

- VV block corresponds to all the entries $a(v_i, v_j)$, where both v_i and v_j are vertex basis functions,
- VE, resp. EV blocks correspond to all the entries $a(v_i, v_j)$, where one of v_i and v_j is a vertex function and the latter is an edge function,
- VF, resp. FV blocks correspond to all the entries $a(v_i, v_j)$, where one of v_i and v_j is a vertex function and the latter is a face function,

- VB, resp. BV blocks correspond to all the entries $a(v_i, v_j)$, where one of v_i and v_j is a vertex function and the latter is a bubble function,
- similarly for the blocks EE, EF, FE, FF, FB, BF, and BB.

Let us focus on the block BB. Due to the fact that the support of a bubble function is one element only, we can see that

- either the support of two bubble basis functions v_i and v_j is the same element, then $a(v_i, v_j)$ is generally nonzero,
- or the basis functions v_i and v_j correspond to different elements, and then $a(v_i, v_j)$ is zero.

This means that the block BB itself has a block structure — it is a block diagonal matrix, where each block corresponds to the bilinear form applied to pairs of bubble functions corresponding to the same element of the finite element mesh (see Figure 3.2).

It is generally hard to optimize vertex, edge and shape functions because their design must comply with the conformity requirements of the underlying finite element space. However, the only requirement for bubble functions is that they vanish on the boundary of reference elements.

3.2 Optimization of reference domain shape functions

The historically first hierarchic shape functions were introduced by the group of B. Szabó [49, 50, 51] in the mid-1970s in the context of the p-FEM. The relation between the representation of the higher-degree polynomial spaces on the finite elements and the condition number of the stiffness and mass matrices was discovered later during more advanced p- and hp-FEM computations [11, 69].

New families of hierarchic finite elements for spaces with different conformity requirements with significantly better conditioning properties were proposed by I. Babuška, B. Szabó [7, 68], M. Ainsworth and J. Coyle [2]. These shape functions are based on integrated Legendre polynomials (known as Lobatto functions) and they were presented in Chapter 2. In some sense,

VV	VE	VF	VB
EV	EE	EF	EB
FV	FE	FF	FB
BV	BE	BF	BB

Figure 3.1: Block structure of the stiffness matrix.

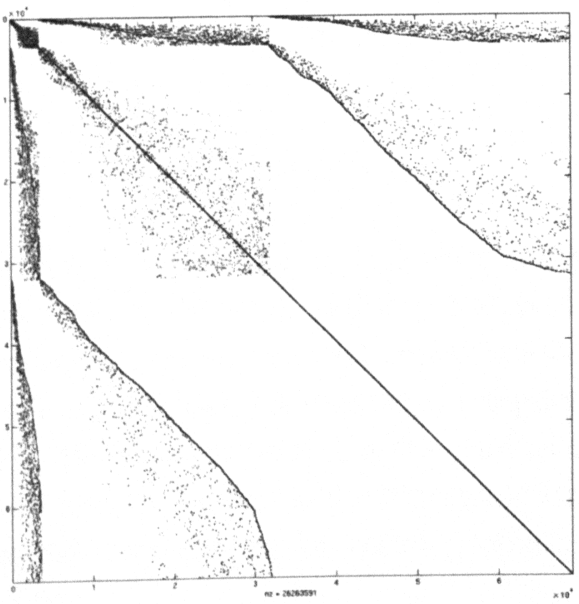


Figure 3.2: Sparsity pattern of a stiffness matrix.

the application of Lobatto polynomials can be viewed as a heuristic attempt to impose the orthogonality under the energetic inner product induced by the Laplace operator. In this chapter we try to go even further: we analyze the possibility of defining a shape function basis that would be "optimal" in some sense. This problem was thoroughly studied by our research group and the results were published in [80, 79, 72, 74, 64].

In this section we impose the following requirement on the bilinear form present in the weak formulation of the problem: let $a(u,v)$ be a V-elliptic symmetric bilinear form. Under these assumptions the bilinear form induces an *energetic inner product*

$$(u,v)_e = a(u,v), \quad u,v \in V. \tag{3.1}$$

It can be shown easily that the bilinear form (2.9) used in our model problem satisfies this assumption. For the sake of simplicity, we put $a_1(\boldsymbol{x}) \equiv 1$ in Ω, thus reducing model equation (2.1), (2.2), and (2.3) to the Poisson equation

$$-\Delta u = f \quad \text{in } \Omega, \tag{3.2}$$

$$u = 0 \quad \text{on } \partial\Omega, \tag{3.3}$$

and (2.9) to

$$a(u,v) := \int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}\boldsymbol{x}. \tag{3.4}$$

If the basis of the finite element space $V_{h,p}$ were orthonormal under the energetic inner product (3.1), then the stiffness matrix A would become an identity matrix. However, it was shown that construction of an orthonormal basis that would have local and hierarchic structure required by the *hp*-FEM is not possible (see [62]). Here we are bound by the conformity requirements of the finite element space and its impact on the hierarchic structure of the basis.

Therefore, we restrict ourselves to an easier problem and introduce the following simplifications.

- First, we leave out vertex, edge, and face shape functions from the orthogonalization process. This way we can construct an orthonormal basis in the subspace

$$\hat{W}_0(\hat{K}) = P_0^p(\hat{K}) = \{\hat{\varphi} \in \hat{W}(\hat{K}); \hat{\varphi}|_{\partial\hat{K}} = 0\} \tag{3.5}$$

of the reference element polynomial space $\hat{W}(\hat{K})$.

- Second, to be able to perform the orthogonalization process, we need to define a suitable bilinear form on the reference element that induces an inner product. Note that the bilinear form $a(\cdot,\cdot)$ is usually defined via a sum of integrals over the mesh elements $K_i \in \mathcal{T}_{h,p}$, $i = 1,\dots,M$. In case of the bilinear form (3.4) resulting from our model problem this translates to

$$a(u,v) = \sum_{i=1}^{M} a_{K_i}(u,v), \tag{3.6}$$

where

$$a_{K_i}(u,v) = \int_{K_i} \nabla u \cdot \nabla v \, \mathrm{d}\boldsymbol{x}, \quad u,v \in V. \tag{3.7}$$

Let us define a new bilinear form

$$a_{\hat{K}}(\cdot,\cdot): \quad \hat{W}(\hat{K}) \times \hat{W}(\hat{K}) \to \mathbb{R}, \tag{3.8}$$

on the reference element $\hat{K}$ by replacing the integration over K_i with the integration over $\hat{K}$. In case of our model problem we get

$$a_{\hat{K}}(\hat{\varphi},\hat{\psi}) = \int_{\hat{K}} \nabla\hat{\varphi} \cdot \nabla\hat{\psi} \, \mathrm{d}\boldsymbol{\xi}, \quad \hat{\varphi},\hat{\psi} \in \hat{W}(\hat{K}). \tag{3.9}$$

The space $\hat{W}_0(\hat{K})$ equipped with the inner product induced by the bilinear form $a_{\hat{K}}(\hat{\varphi},\hat{\psi})$, $\hat{\varphi},\hat{\psi} \in \hat{W}_0(\hat{K})$, is a Hilbert space. Let us denote its basis by $\hat{\mathcal{B}}_0^p$. An orthonormal basis can be constructed in various ways, the most straightforward one that comes to our mind being the modified Gram-Schmidt procedure. We denote the basis arisen from such an orthonormalization procedure by $\hat{\mathcal{B}}_{0,\mathrm{ortho}}^p$.

Remark. There is an infinite number of orthonormal bases of the space $\hat{W}_0(\hat{K})$. The concrete set of orthonormal functions depends, among other things, on what orthonormalization procedure we use and what set of functions we

choose as the underlying basis of $\hat{W}_0(\hat{K})$ for the orthonormalization. One would therefore ask if there is any basis that is special among them – one that has some additional properties that are desirable for the shape functions. In the following section we find out that there is indeed such a set.

By adding the vertex, edge and face shape functions to $\hat{\mathcal{B}}^p_{0,\text{ortho}}$ we get a set of shape functions that forms the basis of $\hat{W}(\hat{K})$ we call *partially orthonormal shape functions* and we denote it by $\hat{\mathcal{B}}^p_{\text{ortho}}$.

3.3 Generalized eigenproblem-based shape functions

A natural question is if there is a set of shape functions that would be optimal for a given problem. Consider model problem (2.1). The entries $\{a_{i,j}\}_{i,j=1}^N$ of the stiffness matrix A are given by

$$a(v_i, v_j) = \int_\Omega \nabla v_j \cdot \nabla v_i \, \mathrm{d}\boldsymbol{x}. \tag{3.10}$$

Again, let us take the polynomial space $\hat{W}_0(\hat{K}) = P^p_0(\hat{K})$, i.e., the space of polynomial functions up to degree p with zero trace on the boundary. It is obvious that the set $\hat{\mathcal{B}}^p_0 = \{\hat{\varphi}^b_i, i = 1, \ldots, m\}$ of all the bubble functions on the reference element up to degree p spans the space $\hat{W}_0(\hat{K})$. The identity (3.10) motivates us to analyze the bilinear form

$$a_{\hat{K}}(\hat{\varphi}^b_i, \hat{\varphi}^b_j) = \int_{\hat{K}} \nabla \hat{\varphi}^b_j \cdot \nabla \hat{\varphi}^b_i \, \mathrm{d}\boldsymbol{\xi}. \tag{3.11}$$

In the previous paragraph we showed that any available orthogonalization procedure can be used, for instance the Gram-Schmidt or modified Gram-Schmidt algorithm. However, these methods can only be applied if the bilinear form (3.11) is symmetric and $\hat{W}_0(\hat{K})$-elliptic. For this reason we prefer a more general approach based on the generalized eigenproblem.

3.3.1 Generalized eigenproblem

We consider the following problem on the reference element $\hat{K}$: Find $(\hat{\psi}, \lambda) \in \hat{W}_0(\hat{K}) \times \mathbb{C}$ such that $\hat{\psi}$ is nonzero on a subset of $\hat{K}$ with nonzero measure and

$$-\Delta\hat{\psi} = \lambda\hat{\psi}. \tag{3.12}$$

Multiplying (3.12) by an arbitrary function $\hat{\varphi} \in \hat{W}_0(\hat{K})$, integrating over $\hat{K}$, and applying Green's theorem to the left-hand side of the equation, we obtain the weak eigenproblem

$$\int_{\hat{K}} \nabla\hat{\psi} \cdot \nabla\hat{\varphi} \, \mathrm{d}\xi = \lambda \int_{\hat{K}} \hat{\psi}\hat{\varphi} \, \mathrm{d}\xi \quad \forall \hat{\varphi} \in \hat{W}_0(\hat{K}). \tag{3.13}$$

Assume that there exists a solution in the form of m pairwise distinct generalized eigenfunction-eigenvalue pairs $(\hat{\psi}_1, \lambda_1), (\hat{\psi}_2, \lambda_2), \ldots, (\hat{\psi}_m, \lambda_m)$. The m generalized bubble eigenfunctions $\hat{\psi}_i$, $i = 1, \ldots, m$, can be expanded in terms of the basis $\hat{\mathcal{B}}_0^p$,

$$\hat{\psi}_i(\xi) = \sum_{l=1}^{m} y_{il} \hat{\varphi}_l^b(\xi), \tag{3.14}$$

with unknown coefficients y_{il}, $i, l = 1, \ldots, m$. Using the expansion and substituting m basis functions for $\hat{\varphi}$ in (3.13), we obtain m equations

$$\sum_{k=1}^{m} y_{ik} \int_{\hat{K}} \nabla\hat{\varphi}_k^b \cdot \nabla\hat{\varphi}_j^b \, \mathrm{d}\xi = \lambda_i \sum_{k=1}^{m} y_{ik} \int_{\hat{K}} \hat{\varphi}_k^b \hat{\varphi}_j^b \, \mathrm{d}\xi, \quad j = 1, \ldots, m. \tag{3.15}$$

It is easy to see now that the weak eigenproblem attains the standard matrix form

$$S\boldsymbol{y}_i = \lambda_i M \boldsymbol{y}_i, \quad i = 1, \ldots, m. \tag{3.16}$$

Here S is the *generalized eigenproblem stiffness matrix*

$$S = \left\{ \int_{\hat{K}} \nabla\hat{\varphi}_k^b \cdot \nabla\hat{\varphi}_j^b \, \mathrm{d}\xi \right\}_{k,j=1}^{m}, \tag{3.17}$$

the *generalized eigenproblem mass matrix* has the form

$$M = \left\{ \int_{\hat{K}} \hat{\varphi}_k^b \hat{\varphi}_j^b \, \mathrm{d}\xi \right\}_{k,j=1}^{m}, \tag{3.18}$$

and $\boldsymbol{y}_i = (y_{i1}, y_{i2}, \ldots, y_{im})^T$, $i = 1, \ldots, m$, are the vectors of unknown expansion coefficients.

3.3.2 Properties of the generalized eigenfunctions

In this section we prove a theorem showing that there is indeed an orthogonal basis of $\hat{W}_0(\hat{K})$ that has some additional special properties that may be beneficial to us in the *hp*-FEM solution process. First, let us start with a well-known result from the linear algebra theory.

Theorem 3.1. *If $A \in \mathbb{R}^{n\times n}$ is symmetric, then there exists a matrix $Q \in \mathbb{R}^{n\times n}$ such that Q is orthogonal (i.e., $Q^TQ = QQ^T = I$) and*

$$Q^TAQ = \Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_n). \tag{3.19}$$

Moreover, for $k = 1, \ldots, n$,

$$A\boldsymbol{q}_k = \lambda_k \boldsymbol{q}_k, \tag{3.20}$$

where $\boldsymbol{q}_k$ is the k-th column of the matrix Q.

Proof. See Theorem 8.1.1 in [31]. □

Remark. Theorem 3.1 states one of the most important aspects of the linear algebra – if a matrix $A \in \mathbb{R}^{n\times n}$ is symmetric, there exists a solution of the eigenvalue problem

$$A\boldsymbol{v} = \lambda\boldsymbol{v}$$

in the form of n pairwise distinct eigenvalue-eigenvector pairs $(\lambda_i, \boldsymbol{v}_i)$, $i = 1, 2, \ldots, n$. Moreover, the eigenvectors $\boldsymbol{v}_i$, $i = 1, 2, \ldots, n$, can be chosen so that they are orthonormal, i.e.,

$$\boldsymbol{v}_i^T\boldsymbol{v}_j = \delta_{ij}, \quad \forall i, j = 1, 2, \ldots, n,$$

where δ_{ij} is the Kronecker delta.

Theorem 3.2. *Let $\hat{K}$ be a reference finite element equipped with a finite element space $\hat{W}_0(\hat{K})$ defined in* (3.5) *and $\hat{\mathcal{B}}_0^p$ be a basis of the space $\hat{W}_0(\hat{K})$. Let*

$$a_{\hat{K}}(\cdot,\cdot)\colon \hat{W}_0(\hat{K}) \times \hat{W}_0(\hat{K}) \to \mathbb{R}$$

be a bounded symmetric $\hat{W}_0(\hat{K})$-elliptic bilinear form inducing an energetic inner product $(\cdot,\cdot)_{e,\hat{K}}$. Then there exists a solution to the generalized eigenproblem (3.16) *in the form of m pairwise distinct generalized eigenfunction-eigenvalue*

pairs $(\lambda_1, \hat{\psi}_1), (\lambda_2, \hat{\psi}_2), \ldots, (\lambda_m, \hat{\psi}_m)$, *with real generalized eigenvalues* λ_i. *The set of generalized eigenfunctions*

$$\hat{\mathcal{B}}^p_{0,eigen} = \{\hat{\psi}_i, i = 1, \ldots, m\}$$

forms a basis of $\hat{W}_0(\hat{K})$ *and is orthogonal under both* $(\cdot,\cdot)_{e,\hat{K}}$ *and* $L^2(\hat{K})$ *inner products.*

Proof. First, let us start by converting the generalized eigenproblem into a standard one. The matrix M is symmetric positive definite and hence, we can use the Cholesky factorization

$$M = GG^T, \tag{3.21}$$

thus the generalized eigenvalue problem (3.16) attains the form

$$S\boldsymbol{y} = \lambda GG^T\boldsymbol{y}. \tag{3.22}$$

By defining $\boldsymbol{z} = G^T\boldsymbol{y}$ we may continue with

$$\begin{aligned} SG^{-T}\boldsymbol{z} &= \lambda G\boldsymbol{z}, \\ G^{-1}SG^{-T}\boldsymbol{z} &= \lambda\boldsymbol{z}, \end{aligned} \tag{3.23}$$

and finally by denoting $C = G^{-1}SG^{-T}$, we obtain a standard eigenvalue problem

$$C\boldsymbol{z} = \lambda\boldsymbol{z}. \tag{3.24}$$

Clearly, the matrix C is symmetric, since

$$\left(G^{-1}SG^{-T}\right)^T = G^{-1}S^TG^{-T} = G^{-1}SG^{-T}. \tag{3.25}$$

By Theorem 3.1 there exists a solution to (3.24) in the form of m eigenvalue-eigenvector pairs $(\lambda_1, \boldsymbol{z}_1)$, $(\lambda_2, \boldsymbol{z}_2)$, $\ldots$, $(\lambda_m, \boldsymbol{z}_m)$ and the eigenvectors

are orthogonal. Hence,

$$\begin{aligned}
\delta_{ij} &= \boldsymbol{z}_i^T \boldsymbol{z}_j = \left(G^T \boldsymbol{y}_i\right)^T G^T \boldsymbol{y}_j = \boldsymbol{y}_i^T G G^T \boldsymbol{y}_j \\
&= \boldsymbol{y}_i M \boldsymbol{y}_j \\
&= \sum_{k=0}^{m} y_{ik} \sum_{l=0}^{m} y_{jl} \int_{\hat{K}} \hat{\varphi}_k^b \hat{\varphi}_l^b \, \mathrm{d}\boldsymbol{\xi} && \text{(recalling (3.18))} \\
&= \int_{\hat{K}} \left(\sum_{k=0}^{m} y_{ik} \hat{\varphi}_k^b(\boldsymbol{\xi}) \right) \left(\sum_{l=0}^{m} y_{jl} \hat{\varphi}_l^b(\boldsymbol{\xi}) \right) \mathrm{d}\boldsymbol{\xi} \\
&= \int_{\hat{K}} \hat{\psi}_i \hat{\psi}_j \, \mathrm{d}\boldsymbol{\xi} && \text{(by (3.14))} \\
&= (\hat{\psi}_i, \hat{\psi}_j)_{L^2(\hat{K})}.
\end{aligned}$$

Indeed, the generalized eigenproblem-based basis $\hat{\mathcal{B}}^p_{0,\text{eigen}}$ is orthogonal under $L^2(\hat{K})$ inner product.

Finally, we need to prove the orthogonality of $\hat{\mathcal{B}}^p_{0,\text{eigen}}$ in the $(\cdot,\cdot)_{e,\hat{K}}$ inner product. We multiply the previous equation by λ_j so that

$$\begin{aligned}
\lambda_j \delta_{ij} &= \lambda_j (\hat{\psi}_i, \hat{\psi}_j)_{L^2(\hat{K})} = \\
&= \lambda_j \boldsymbol{y}_i^T M \boldsymbol{y}_j \\
&= \boldsymbol{y}_i^T \left(\lambda_j M \boldsymbol{y}_j^T \right) \\
&= \boldsymbol{y}_i^T S \boldsymbol{y}_j^T \\
&= \sum_{k=0}^{m} y_{ik} \sum_{l=0}^{m} y_{jl} \int_{\hat{K}} \nabla \hat{\varphi}_k^b \cdot \nabla \hat{\varphi}_l^b \, \mathrm{d}\boldsymbol{\xi} && \text{(recalling (3.17))} \\
&= \int_{\hat{K}} \nabla \hat{\psi}_i \cdot \nabla \hat{\psi}_j \, \mathrm{d}\boldsymbol{\xi} && \text{(by (3.14))} \\
&= (\nabla \hat{\psi}_i, \nabla \hat{\psi}_j)_{e,\hat{K}},
\end{aligned}$$

thus the $(\cdot,\cdot)_{e,\hat{K}}$-orthogonality is a direct consequence of the $(\cdot,\cdot)_{L^2(\hat{K})}$-orthogonality. The linear independence of $\hat{\mathcal{B}}^p_{0,\text{eigen}}$ follows trivially from the $(\cdot,\cdot)_{L^2(\hat{K})}$-orthogonality as well. □

Remark. We mentioned that there is an infinite number of bases of the space $\hat{W}_0(\hat{K})$ that are orthogonal under the $(\cdot,\cdot)_{e,\hat{K}}$ inner product. The set of the

generalized bubble eigenfunctions $\hat{\mathcal{B}}^p_{0,\text{eigen}}$ is optimal in the sense that if all the generalized eigenvalues λ_i, $i = 1, \ldots, m$, are distinct, there is no other basis (up to multiplicative constants) in $\hat{W}_0(\hat{K})$ orthogonal under both the $(\cdot,\cdot)_{e,\hat{K}}$ and $L^2(\hat{K})$ products.

Remark. Note that in the proof there was no need for the matrix S to be positive definite (i.e., the bilinear form $a_{\hat{K}}(\cdot,\cdot)$ does not have to be $\hat{W}_0(\hat{K})$-elliptic), and hence, a similar theorem may be proved in a more general setting. Of course, then the bilinear form $a_{\hat{K}}(\cdot,\cdot)$ does not induce an energetic inner product and we cannot speak about orthogonality. Nevertheless, it still holds that $a_{\hat{K}}(\hat{\psi}_i, \hat{\psi}_j) = 0$ if $i \neq j$, and $L^2(\hat{K})$ orthogonality is also preserved. This means that we can use this procedure in more general cases.

For example, we can construct generalized eigenfunctions of the curl-curl operator associated with the Maxwell's equations of electromagnetics. We studied this problem in more details in [72] and [74].

The orthogonality of the shape functions on the reference element gives us hope that the condition number of the stiffness matrix might drop significantly. However, we cannot expect the preservation of orthogonality in the global stiffness matrix. Due to the use of the reference maps (2.22), we have to consider the gradient transformation rule (2.58)

$$\nabla_{\boldsymbol{x}} = \left(\frac{D\boldsymbol{x}_{K_i}}{D\boldsymbol{\xi}}\right)^{-T} \nabla_{\boldsymbol{\xi}}.$$

Thus the contribution to the entries (3.10) of the stiffness matrix A on element K_i are given by

$$\int_{K_i} \nabla_{\boldsymbol{x}} \psi_i \cdot \nabla_{\boldsymbol{x}} \psi_j \, \mathrm{d}\boldsymbol{x} = \int_{\hat{K}} \nabla_{\boldsymbol{\xi}} \hat{\psi}_i \left(\frac{D\boldsymbol{x}_{K_i}}{D\boldsymbol{\xi}}\right)^{-1} \left(\frac{D\boldsymbol{x}_{K_i}}{D\boldsymbol{\xi}}\right)^{-T} \nabla_{\boldsymbol{\xi}} \hat{\psi}_j \, \mathrm{d}\boldsymbol{\xi}, \tag{3.26}$$

with the matrix

$$\left(\frac{D\boldsymbol{x}_{K_i}}{D\boldsymbol{\xi}}\right)^{-1} \left(\frac{D\boldsymbol{x}_{K_i}}{D\boldsymbol{\xi}}\right)^{-T} \tag{3.27}$$

being symmetric positive definite, but generally, unfortunately, nondiagonal. Looking at identities (3.26) and (3.27), we can see that the orthogonality of

the generalized bubble eigenfunctions in the $H_0^1(\hat{K})$ product is not preserved in the $H_0^1(K_i)$ product for their transformed counterparts.

However, this approach may still be very relevant in case we are performing computation on structured meshes when the mesh elements are similar to the reference element.

Theorem 3.3. *Let $\hat{K}$ be a reference domain and K_i its counterpart transformed by the reference mapping $\boldsymbol{x}_{K_i}$ such that $\hat{K}$ and K_i are geometrically similar. Consider bilinear forms $a_{K_i}(\cdot,\cdot)$ and $a_{\hat{K}}(\cdot,\cdot)$ as in (3.7) and (3.9), respectively. Next, let $\hat{\mathcal{B}}^p_{0,eigen}$ be the generalized eigenfunction-based basis of the finite element space $\hat{W}_0(\hat{K})$ defined above and $W_{0,i}$ its transformed counterpart,*

$$W_{0,i} = W_0(K_i) = \boldsymbol{\Phi}^1_{K_i}(\hat{W}_0(\hat{K})) = \{w \in W_i,\ w|_{\partial K_i} = 0\} = P_0^{p_i}(K_i). \tag{3.28}$$

Then the transformed basis

$$\mathcal{B}^p_{0,eigen} = \{\psi_i \in W_0(K_i),\ \psi_i = \hat{\psi}_i \circ \boldsymbol{x}^{-1}_{K_i}, \text{ where } \hat{\psi}_i \in \hat{\mathcal{B}}^p_{0,eigen},\ i = 1,\dots,m\} \tag{3.29}$$

is orthogonal in both the $(\cdot,\cdot)_{e,K_i}$ inner product induced by the bilinear form $a_{K_i}(\cdot,\cdot)$ and the $L^2(K_i)$ inner product.

Proof. First, let us recall that to prove the orthogonality in the $(\cdot,\cdot)_{e,K_i}$ inner product we have to show that under the above specified conditions the matrix (3.27) is a multiple of the identity matrix. Then we can write

$$\begin{aligned}\int_{K_i} \nabla_{\boldsymbol{x}}\psi_i \cdot \nabla_{\boldsymbol{x}}\psi_j \,\mathrm{d}\boldsymbol{x} &= \int_{\hat{K}} \nabla_\xi \hat{\psi}_i \left(\frac{D\boldsymbol{x}_{K_i}}{D\xi}\right)^{-1} \left(\frac{D\boldsymbol{x}_{K_i}}{D\xi}\right)^{-T} \nabla_\xi \hat{\psi}_j \,\mathrm{d}\xi \\ &= \int_{\hat{K}} \nabla_\xi \hat{\psi}_i (sI) \nabla_\xi \hat{\psi}_j \,\mathrm{d}\xi \\ &= s\int_{\hat{K}} \nabla_\xi \hat{\psi}_i \cdot \nabla_\xi \hat{\psi}_j \,\mathrm{d}\xi = s\delta_{ij},\end{aligned}$$

where I is the identity matrix, s a nonzero multiplicative constant and δ_{ij} the Kronecker delta.

Let us inspect the properties of the mapping $\boldsymbol{x}_{K_i}$ in detail. We assumed that $\hat{K}$ and K_i are geometrically similar, which imposes a restriction on $\boldsymbol{x}_{K_i}$. More

specifically, $\boldsymbol{x}_{K_i}$ can only be a composition of 3D translation $\boldsymbol{x}_{K_i,\text{trans}}$, scaling $\boldsymbol{x}_{K_i,\text{scale}}$, rotation $\boldsymbol{x}_{K_i,\text{rot}}$, and possible reflection $\boldsymbol{x}_{K_i,\text{ref}}$. Let us describe these mappings analytically:

$$\boldsymbol{x}_{K_i,\text{trans}}(\xi) = (\xi_1 + c_1, \xi_2 + c_2, \xi_3 + c_3)^T, \tag{3.30}$$

$$\boldsymbol{x}_{K_i,\text{scale}}(\xi) = r(\xi_1, \xi_2, \xi_3)^T, \tag{3.31}$$

$$\boldsymbol{x}_{K_i,\text{rot}}(\xi) = A(\xi_1, \xi_2, \xi_3)^T, \tag{3.32}$$

$$\boldsymbol{x}_{K_i,\text{ref}}(\xi) = (\xi_1, \xi_2, \pm\xi_3)^T, \tag{3.33}$$

where c_1, c_2, and c_3 are arbitrary constants of translation, $r \neq 0$ is an arbitrary scaling coefficient, and the $\pm$ sign just stands for the possibility of applying the reflection or not. Finally, A is the 3D rotation matrix

$$A = \begin{pmatrix} c_\alpha c_\gamma - s_\alpha c_\beta s_\gamma & s_\alpha c_\gamma + c_\alpha c_\beta s_\gamma & s_\beta s_\gamma \\ -c_\alpha s_\gamma - s_\alpha c_\beta c_\gamma & -s_\alpha s_\gamma + c_\alpha c_\beta c_\gamma & s_\beta c_\gamma \\ s_\alpha s_\beta & -c_\alpha s_\beta & c_\beta \end{pmatrix}, \tag{3.34}$$

where c_ω and s_ω stands for $\cos(\omega)$ and $\sin(\omega)$, respectively, and α, β and γ are the Euler angles of rotation.

Now any affine reference mapping $\boldsymbol{x}_{K_i}$ preserving similarity may be written as

$$\boldsymbol{x}_{K_i} = \boldsymbol{x}_{K_i,\text{trans}} \circ \boldsymbol{x}_{K_i,\text{scale}} \circ \boldsymbol{x}_{K_i,\text{rot}} \circ \boldsymbol{x}_{K_i,\text{ref}}, \tag{3.35}$$

or more specifically as

$$\boldsymbol{x}_{K_i}(\xi) = rA(\xi_1, \xi_2, \pm\xi_3)^T + (c_1, c_2, c_3)^T. \tag{3.36}$$

Hence the Jacobi matrix $\left(\frac{D\boldsymbol{x}_{K_i}}{D\xi}\right)$ has the form

$$\left(\frac{D\boldsymbol{x}_{K_i}}{D\xi}\right) = r \begin{pmatrix} c_\alpha c_\gamma - s_\alpha c_\beta s_\gamma & s_\alpha c_\gamma + c_\alpha c_\beta s_\gamma & \pm s_\beta s_\gamma \\ -c_\alpha s_\gamma - s_\alpha c_\beta c_\gamma & -s_\alpha s_\gamma + c_\alpha c_\beta c_\gamma & \pm s_\beta c_\gamma \\ s_\alpha s_\beta & -c_\alpha s_\beta & \pm c_\beta \end{pmatrix}. \tag{3.37}$$

It is straightforward to verify that

$$\left(\frac{D\boldsymbol{x}_{K_i}}{D\xi}\right)^{-1} = \frac{1}{r^2}\left(\frac{D\boldsymbol{x}_{K_i}}{D\xi}\right)^T, \tag{3.38}$$

and thus the matrix (3.27) attains the form

$$\left(\frac{D\boldsymbol{x}_{K_i}}{D\xi}\right)^{-1}\left(\frac{D\boldsymbol{x}_{K_i}}{D\xi}\right)^{-T}=\left(\frac{D\boldsymbol{x}_{K_i}}{D\xi}\right)^{-1}\frac{1}{r^2}\left(\frac{D\boldsymbol{x}_{K_i}}{D\xi}\right)=\frac{1}{r^2}I, \tag{3.39}$$

i.e., it is indeed a multiple of the identity matrix.

The orthogonality in the $L^2(K_i)$ inner product is a consequence of the substitution theorem. Assume that $k \neq l$. Then

$$\begin{aligned} 0 &= \int_{\hat{K}} \hat{\psi}_k(\xi)\,\hat{\psi}_l(\xi)\,\mathrm{d}\xi \\ &= \int_{K_i} \hat{\psi}_k(\boldsymbol{x}_{K_i}^{-1}(\boldsymbol{x}))\,\hat{\psi}_l(\boldsymbol{x}_{K_i}^{-1}(\boldsymbol{x}))\det\left(\frac{D\boldsymbol{x}_{K_i}^{-1}}{D\boldsymbol{x}}\right)\mathrm{d}\boldsymbol{x}. \end{aligned} \tag{3.40}$$

First, recall the way how we defined the transformed basis functions using the affine concept in (3.29). Next, from equations (3.37) and (3.38) it is easy to see that

$$\det\left(\frac{D\boldsymbol{x}_{K_i}^{-1}}{D\boldsymbol{x}}\right)=\frac{1}{r},$$

and hence,

$$0=\int_{\hat{K}}\hat{\psi}_k(\xi)\,\hat{\psi}_l(\xi)\,\mathrm{d}\xi=\frac{1}{r}\int_{K_i}\psi_k(\boldsymbol{x})\,\psi_l(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}, \tag{3.41}$$

proving the orthogonality of the transformed generalized eigenproblem-based bubble functions in the $L^2(K_i)$ inner product. □

Theorem 3.3 gives us a sufficient condition for the preservation of orthogonality of the generalized eigenproblem-based bubble shape functions after their transformation to the physical mesh. Unfortunately, this condition is very restrictive and is usually satisfied in the case when using a structured mesh such that all the mesh elements are similar to the reference element. Note that this mesh may also be isotropically adapted. Moreover, in 3D the practical benefit may only be gained when using hexahedral elements, since it is generally practically unfeasible to construct a geometrical division of a computational domain into tetrahedra similar to the reference tetrahedron.

Nevertheless, if the mesh is not structured, the approach described in this section may still serve as an excellent preconditioner for the problem. This topic was thoroughly analyzed in [71].

Remark. By using generalized bubble eigenfunctions, we are losing part of the hierarchy of the shape functions. This is due to the fact that the generalized eigenfunctions must be constructed separately for each polynomial degree p. Fortunately, this is just a minor implementation problem, since the support of bubble functions lies in one element only.

3.4 Numerical example

We present a simple example showing how choice of a set of shape functions can affect the conditioning properties of a stiffness matrix. To be able to compute condition numbers of stiffness matrices we construct rather a simple and academic problem. Using the computational domain

$$\Omega = (-1,1)\times(-1,1)\times(-1,0)\cup(-1,0)\times(0,1)\times[0,1)$$

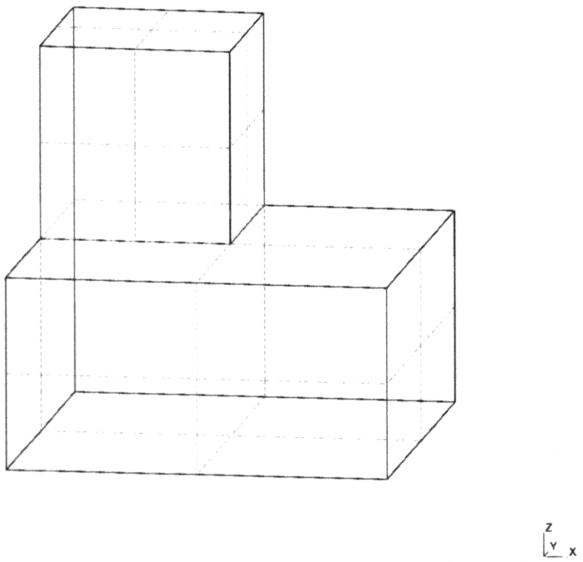
(shown in Figure 3.3a), find $u \in H_0^1(\Omega)$ such that

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega, \\ u|_{\partial\Omega} &= g_D. \end{aligned} \tag{3.42}$$

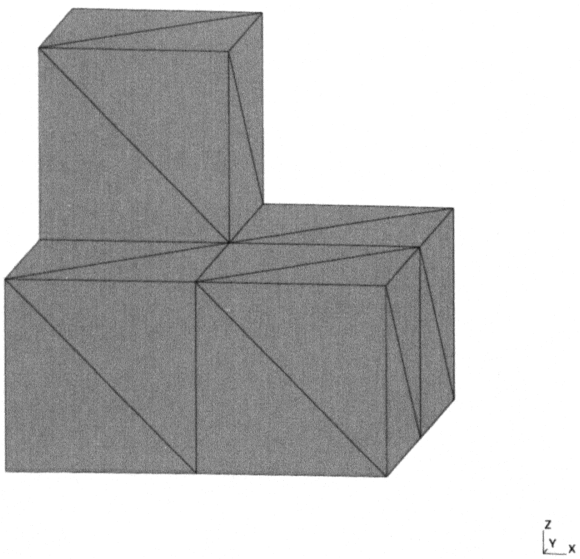
We discretize the computational domain using a mesh shown in Figure 3.3b comprising of 30 tetrahedral elements.

We are interested in the properties of the stiffness matrix only and hence, we do not need to specify f and g_D in detail, since both functions only affect the right-hand side of the resulting algebraic problem.

In this test we construct the stiffness matrix using uniform distribution of polynomial degrees $p = 4,5,\ldots,10$, across the whole mesh. We also use various sets of shape functions, namely the basic set of monomial-based shape functions (see [62]), Lobatto-based shape functions (see Section 2.2), and finally the generalized eigenproblem-based shape functions defined in this section.



(a)



(b)

Figure 3.3: Computational domain and mesh used to test conditioning properties of various types of shape functions.

Degree	Matrix size	Bubble DOFs	cond(A) (monomial)	cond(A) (Lobatto)	cond(A) (eigen)
4	183	30	$7.444 \cdot 10^4$	$3.942 \cdot 10^2$	$2.408 \cdot 10^2$
5	404	120	$7.314 \cdot 10^6$	$5.675 \cdot 10^3$	$2.573 \cdot 10^3$
6	755	300	$1.038 \cdot 10^9$	$9.225 \cdot 10^4$	$2.848 \cdot 10^4$
7	1266	600	$1.605 \cdot 10^{11}$	$1.908 \cdot 10^6$	$3.510 \cdot 10^5$
8	1967	1050	$2.586 \cdot 10^{13}$	$3.544 \cdot 10^7$	$4.656 \cdot 10^6$
9	2888	1680	$4.233 \cdot 10^{15}$	$8.137 \cdot 10^8$	$6.467 \cdot 10^7$
10	4059	2520	$6.994 \cdot 10^{17}$	$1.978 \cdot 10^{10}$	$9.286 \cdot 10^8$

Table 3.1: Comparison of stiffness matrix condition numbers obtained for different sets of shape functions and different polynomial degrees. For each polynomial degree we present the size of the corresponding stiffness matrix, number of degrees of freedom (DOFs) associated to bubble functions, and stiffness matrix condition numbers for all shape functions sets.

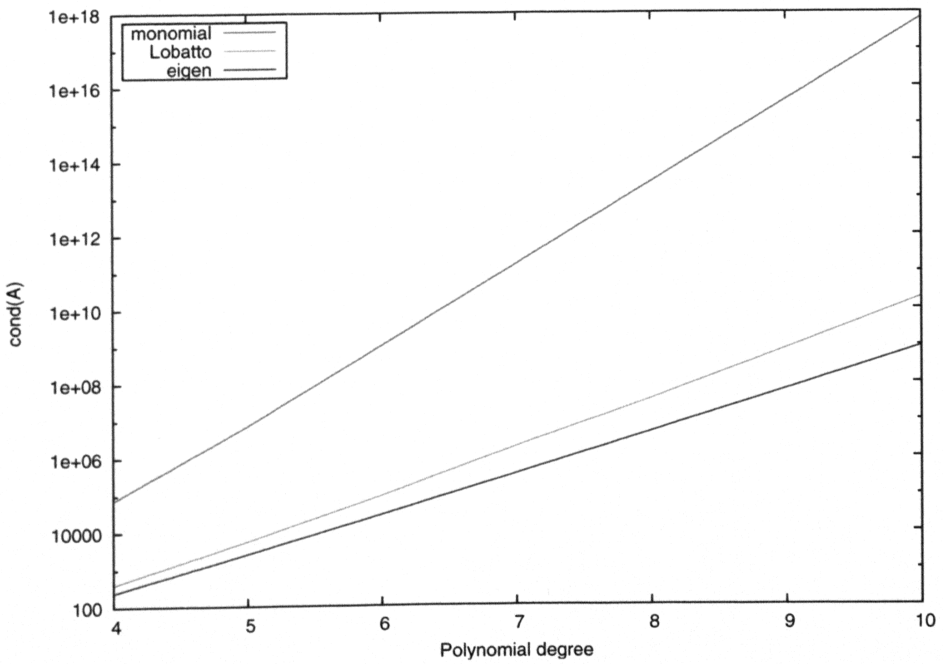


Figure 3.4: Dependence of the stiffness matrix condition number on the polynomial degree used for different types of shape functions.

The comparison is presented in Table 3.1 and in Figure 3.4. We can see that the set of Lobatto-based shape functions already performs much better than the monomial-based shape functions. Even more significant drop in the condition number was obtained using the generalized eigenproblem-based shape functions.

CHAPTER 4

Conditioning properties of basis functions in mesh elements

In the previous chapter we have presented a heuristic approach to shape function optimization on the reference element. However, this approach does not lead to ideal results – elimination of as many nonzero entries in the stiffness matrix as possible. This is mainly due to non-preservation of orthogonality during the transformation of bilinear forms from the reference domains to the mesh elements. Thus we want to study approaches that do not suffer from this problem.

In this chapter we present a method based on the construction of mesh element generalized eigenfunctions that not only help us to remove an essential number of nonzero entries from the stiffness matrix but also reduce the size of the stiffness matrix considerably.

In contrast to the method of generalized eigenproblem-based shape functions presented in the previous section this method diverts from the affine concept of the *hp*-FEM and thus we are talking about *non-affine concept* of the finite element method.

This chapter presents original contribution of the author. The analysis of the generalized eigenproblem-based non-affine finite element method together with numerical results were published in [64, 80]. The chapter is heavily based on the aforementioned publications.

4.1 Construction of partially orthogonal basis functions in mesh elements

The concept shown in the previous chapter has a serious drawback: the orthogonality achieved on the reference domain is spoiled by the incorporation of reference maps $\boldsymbol{x}_{K_i}$ when transforming gradients to the mesh elements. This leads us to the idea of computing the generalized eigenfunctions in the mesh elements. Of course, it is not desirable to construct the basis of $V_{h,p}$ consisting of generalized eigenfunctions only. This is because the support of vertex, edge and face functions spans multiple mesh elements and thus the problem of finding a basis consisting of generalized eigenfunctions is a global problem of the same size as the original elliptic problem. However, if we focus on bubble functions only, we can see that we can eliminate the BB block in the stiffness matrix quite easily.

4.1.1 Generalized bubble eigenfunctions

For each mesh element let us recall a polynomial space W_i defined in (2.26)

$$W_i = \{w \in P^{p_i}(K_i),\ w|_{s_{i_l}} \in P^{p_{s_{i_l}}}(s_{i_l}),\ l = 1, \dots, S_i,\\ w|_{e_{i_j}} \in P^{p_{e_{i_j}}}(e_{i_j}),\ j = 1, \dots, E_i\}.$$

Next we reintroduce $W_{0,i}$ a subspace of W_i spanned by the bubble functions on the element K_i only

$$W_{0,i} = W_0(K_i) = \{w \in W_i,\ w|_{\partial K_i} = 0\} = P_0^{p_i}(K_i). \tag{4.1}$$

In this section we consider the bilinear form (2.9) without any simplifications,

$$a(u, v) = \int_\Omega a_1(\boldsymbol{x})\ \nabla u \cdot \nabla v\ \mathrm{d}\boldsymbol{x}. \tag{4.2}$$

When assembling the stiffness matrix the evaluation of the bilinear form is usually decomposed into the sum

$$a(u,v) = \sum_{i=1}^{M} a_{K_i}(u,v), \tag{4.3}$$

where

$$a_{K_i}(u,v) = \int_{K_i} a_1(\boldsymbol{x})\, \nabla u \cdot \nabla v \, \mathrm{d}\boldsymbol{x}, \quad u,v \in V. \tag{4.4}$$

This leads us to a detailed analysis of an eigenproblem for the differential operator $-\nabla \cdot (a_1 \nabla \cdot)$: Find $(\psi_i, \lambda_i) \in W_{0,i} \times \mathbb{C}$ such that ψ_i is nonzero on a subset of K_i with nonzero measure and

$$-\nabla \cdot (a_1(\boldsymbol{x})\, \nabla \psi_i) = \lambda_i \psi_i. \tag{4.5}$$

Following the same path as in the previous chapter, we may reformulate this problem in its weak form in terms of the integrals

$$\int_{K_i} a_1(\boldsymbol{x})\, \nabla \psi_i \cdot \nabla \varphi \, \mathrm{d}\boldsymbol{x} = \lambda_i \int_{K_i} \psi_i \varphi \, \mathrm{d}\boldsymbol{x} \quad \forall \varphi \in W_{0,i}, \tag{4.6}$$

or, using relation (4.4),

$$a_{K_i}(\psi_i, \varphi) = \lambda_i (\psi_i, \varphi)_{L^2(K_i)} \quad \forall \varphi \in W_{0,i}. \tag{4.7}$$

If the bilinear form $a_{K_i}(\cdot,\cdot)$ induces an inner product (denoted by $(\cdot,\cdot)_{e,i}$) then we can rewrite (4.6) as

$$(\psi_i, \varphi)_{e,i} = \lambda_i (\psi_i, \varphi)_{L^2(K_i)} \quad \forall \varphi \in W_{0,i}. \tag{4.8}$$

Note that for some differential operators the bilinear form does not necessarily have to induce an inner product, but still we can use the framework present in this section. We will discuss this generalization later.

Similarly as in Section 3.3, for each element K_i we arrive to the generalized eigenproblem

$$S_i \mathbf{y}_{i,j} = \lambda_{i,j} M_i \mathbf{y}_{i,j}, \; 1 \le j \le m_i, \tag{4.9}$$

where the local stiffness matrix S_i is defined as

$$S_i = \left\{ \int_{K_i} a_1(\boldsymbol{x}) \, \nabla \varphi_{i,k} \cdot \nabla \varphi_{i,l} \, \mathrm{d}\boldsymbol{x} \right\}_{k,l=1}^{m_i} . \tag{4.10}$$

The local mass matrix M_i has the form

$$M_i = \left\{ \int_{K_i} \varphi_{i,k} \varphi_{i,l} \, \mathrm{d}\boldsymbol{x} \right\}_{k,l=1}^{m_i} , \tag{4.11}$$

where $\varphi_{i,k}$, $k = 1, \ldots, m_i$, are elements of a basis $\mathcal{B}_{0,i}$ of the space $W_{0,i}$ and m_i is the cardinality of such a basis. Recall that if p_b is the internal polynomial order of approximation for a tetrahedral element K_i, then $m_i = (p_b - 1)(p_b - 2)(p_b - 3)/6$. If K_i is a hexahedron and $p_{b,k}$, $k = 1, 2, 3$, specify the internal directional orders of approximation, then $m_i = (p_{b,1} - 1)(p_{b,2} - 1)(p_{b,3} - 1)$. Finally, the vectors $\boldsymbol{y}_{i,j} = (y_{i,j1}, y_{i,j2}, \ldots, y_{i,jm_i})^T$, $j = 1, \ldots, m_i$, contain the expansion coefficients of the generalized bubble eigenfunctions $\psi_{i,j} = \sum_{k=1}^{m_i} y_{i,jk} \varphi_{i,k}$.

In our setting both matrices S_i and M_i are symmetric and positive definite and thus we can formulate a theorem similar to the one in Section 3.3.

Theorem 4.1. *Let $\mathcal{T}_{h,p} = \{K_1, K_2, \ldots, K_M\}$ be the finite element mesh covering the computational domain Ω and $V_{h,p}$ be the associated finite element space. Let*

$$a(\cdot, \cdot) : V_{h,p} \times V_{h,p} \to \mathbb{R}$$

be a bounded symmetric $V_{h,p}$-elliptic bilinear form inducing an energetic inner product $(\cdot, \cdot)_e$. Then for each element K_i there exists a solution to the generalized eigenproblem (4.7) in the form of m_i pairwise distinct generalized eigenfunction-eigenvalue pairs $(\lambda_{i,1}, \psi_{i,1}), (\lambda_{i,2}, \psi_{i,2}), \ldots, (\lambda_{i,m_i}, \psi_{i,m_i})$, with real generalized eigenvalues $\lambda_{i,j}$. For all K_i, $i = 1, \ldots, M$, the set of generalized eigenfunctions

$$\mathcal{B}_{0,i,eigen} = \{\psi_{i,j}, j = 1, \ldots, m_i\}$$

forms a basis of $W_{0,i}$ and is orthogonal under both $(\cdot, \cdot)_{e,K_i}$ and $L^2(K_i)$ inner products.

Proof. We use the fact that if $a(\cdot,\cdot)$ is a bounded $V_{h,p}$-elliptic bilinear form, then for each K_i the bilinear form $a_{K_i}(\cdot,\cdot)$ is also bounded and $W_{0,i}$-elliptic, since a_{K_i} is just a restriction of a on K_i. Next, we construct the proof in a similar way as in Theorem 3.2. □

Remark. Note that S_i does not have to be positive definite (i.e., $a_{K_i}(\cdot,\cdot)$ does not have to be $V_{h,p}$-elliptic) to achieve the same result. Then, again, the bilinear form does not induce an inner product and we only can speak about the orthogonality in the $L^2(K_i)$ inner product. Anyway, it still holds that

$$a_{K_i}(\psi_{i,k},\psi_{i,l}) = 0 \text{ if } k \neq l.$$

This is the case when seeking the generalized bubble eigenfunctions for the curl-curl operator appearing in the weak formulation of Maxwell's equations.

By solving the generalized eigenproblems on each mesh element, we define a new set of bubble functions $\mathcal{B}_{0,i,\text{eigen}} = \{\psi_{i1},\psi_{i2},\dots,\psi_{im_i}\}$ forming the basis of $W_{0,i}$. Using the conclusion of Theorem 4.1, we can see that

$$\int_{K_i} \psi_{i,k}\psi_{i,l}\,\mathrm{d}\boldsymbol{x} = 0, \quad \text{whenever } k \neq l, \tag{4.12}$$

and also

$$\int_{K_i} a_1(\boldsymbol{x})\,\nabla\psi_{i,k}\cdot\nabla\psi_{i,l}\,\mathrm{d}\boldsymbol{x} = 0, \quad k \neq l. \tag{4.13}$$

Now let us look at the functions $\psi_{i,j}$, $i = 1,\dots,M$, $j = 1,\dots,m_i$, as basis functions of $V_{h,p}$ by extending their domain of definition on the whole Ω. We define $\psi_{i,k}|_{K_j} = 0$ for any i and j such that $i \neq j$, and hence, we may conclude that

$$a(\psi_{i,k},\psi_{j,l}) = \int_{\Omega} a_1(\boldsymbol{x})\,\nabla\psi_{i,k}\cdot\nabla\psi_{j,l}\,\mathrm{d}\boldsymbol{x} = 0 \quad \text{if } i \neq j \text{ or } l \neq k. \tag{4.14}$$

Finally, without loss of generality let us assume that

$$a(\psi_{i,k},\psi_{i,k}) = 1 \tag{4.15}$$

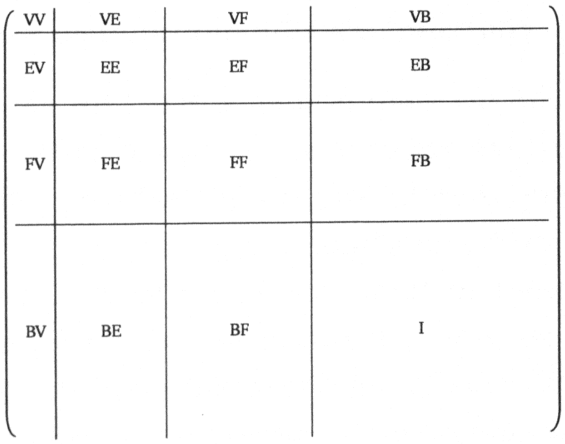


Figure 4.1: Block structure of the stiffness matrix.

for every element K_i and for $k = 1, \ldots, m_i$. This can be achieved by scaling the generalized bubble eigenfunctions using the generalized eigenvalues λ. Hence,

$$a(\psi_{i,k}, \psi_{j,l}) = \int_\Omega a_1(\boldsymbol{x})\, \nabla\psi_{i,k} \cdot \nabla\psi_{j,l}\, \mathrm{d}\boldsymbol{x} = \delta_{ij}\delta_{kl}. \tag{4.16}$$

We denote the set of all generalized bubble eigenfunctions on all elements by $\mathcal{B}_{0,\text{eigen}}$. By adding vertex, edge, and face basis functions to $\mathcal{B}_{0,\text{eigen}}$, we get a complete basis of the finite element space $V_{h,p}$ that we denote by $\mathcal{B}_{b,\text{eigen}}$.

When using the basis $\mathcal{B}_{b,\text{eigen}}$ (the subscript b stands for the fact that we substituted the generalized bubble eigenfunctions for the standard *bubble* functions) for the finite element discretization the structure of the stiffness matrix A gets simplified considerably. It follows from (4.16) that the BB block is now formed by the identity matrix as illustrated in Figure 4.1.

Remark. Note that we used the generalized eigenproblem to orthogonalize the bubble functions. If the bilinear form $a(\cdot,\cdot)$ induces an inner product then any orthogonalization procedure yields a similar result. However, orthogonalization by solving the generalized eigenproblem has several advantages that we present at the end of this section.

4.1.2 Vertex, edge, and face eigenproblem-based functions

Once we eliminated the BB block a natural question arises: Is it possible to eliminate any of the other blocks of the stiffness matrix? Further analysis shows that not only it is possible, but the procedure is relatively easy.

Let φ_j^v, $j = 1,\ldots,R_i$, be the vertex functions corresponding to the vertices of the element K_i. Note that the support of these functions extends also to the neighbors of K_i. Let us construct a new set of vertex functions $\{\psi_j^v,\ j = 1,\ldots,R_i\}$ by defining the restriction of each function on each element K_i using restrictions of the original vertex functions φ_j^v on K_i and the generalized bubble eigenfunctions $\psi_{i,k}$, $k = 1,\ldots,m_i$,

$$\psi_j^v|_{K_i} = \varphi_j^v|_{K_i} - \sum_{k=1}^{m_i} c_{jk}^v \psi_{i,k}, \quad j = 1,\ldots,R_i. \tag{4.17}$$

The coefficients c_{jk}^v are defined by

$$c_{jk}^v = a_{K_i}(\varphi_j^v|_{K_i}, \psi_{i,k}), \quad 1 \le j \le R_i,\ 1 \le k \le m_i. \tag{4.18}$$

Remark. Observe that each original vertex function is slightly modified in each element that is present in the support of that function. However, by this change we are not modifying the traces of vertex functions along the edges and faces of the mesh elements. Hence, the global conformity is not impaired in any way.

Lemma 4.1. *Let ψ_j^v be the modified vertex functions defined by the relations* (4.17) *and* (4.18). *For the sake of clarity let us recall that $\psi_{i,l}$, $l = 1,\ldots,m_i$, are the generalized eigenproblem-based bubble functions on K_i. Then for all the mesh elements K_i, $i = 1,\ldots,M$, it holds that*

$$a_{K_i}(\psi_j^v, \psi_{i,l}) = 0, \quad \text{for all } j = 1,\ldots,R_i, \text{ and } l = 1,\ldots,m_i, \tag{4.19}$$

where $a_{K_i}(\cdot,\cdot)$ is the bilinear form defined in (4.4).

Proof. Let us start with the expression $a_{K_i}(\psi_j^v, \psi_{i,l})$ expanding and simplifying

it using relations (4.15), (4.16), (4.17) and (4.18),

$$\begin{aligned} a_{K_i}(\psi_j^v, \psi_{i,l}) &= a_{K_i}\left(\varphi_j^v|_{K_i} - \sum_{k=1}^{m_i} c_{jk}^v \psi_{i,k}, \psi_{i,l}\right) \\ &= a_{K_i}\left(\varphi_j^v|_{K_i}, \psi_{i,l}\right) - \sum_{k=1}^{m_i} c_{jk}^v \underbrace{a_{K_i}\left(\psi_{i,k}, \psi_{i,l}\right)}_{=0 \text{ if } k\neq l, \text{by (4.16)}} \\ &= a_{K_i}\left(\varphi_j^v|_{K_i}, \psi_{i,l}\right) - c_{jl}^v \underbrace{a_{K_i}\left(\psi_{i,l}, \psi_{i,l}\right)}_{=1, \text{by (4.15)}} \\ &= a_{K_i}\left(\varphi_j^v|_{K_i}, \psi_{i,l}\right) - a_{K_i}(\varphi_j^v, \psi_{i,l}) = 0. \end{aligned} \tag{4.20}$$

□

This result has a direct consequence: by using the generalized eigenproblem-based vertex functions ψ_j^v we have virtually eliminated the blocks VB and BV of the stiffness matrix A.

The same procedure may be applied to edge and face functions. Let us define a new set of edge functions on each element K_i by

$$\psi_j^e|_{K_i} = \varphi_j^e|_{K_i} - \sum_{k=1}^{m_i} c_{jk}^e \psi_{i,k}, \quad j = 1, \ldots, E_i, \quad i = 1, \ldots, M, \tag{4.21}$$

where

$$c_{jk}^e = a_{K_i}(\varphi_j^e, \psi_{i,k}), \quad 1 \le j \le E_i,\ 1 \le k \le m_i. \tag{4.22}$$

Finally, let the new set of face functions be defined as

$$\psi_j^s|_{K_i} = \varphi_j^s|_{K_i} - \sum_{k=1}^{m_i} c_{jk}^s \psi_{i,k}, \quad j = 1, \ldots, S_i, \quad i = 1, \ldots, M, \tag{4.23}$$

where

$$c_{jk}^s = a_{K_i}(\varphi_j^s, \psi_{i,k}), \quad 1 \le j \le S_i,\ 1 \le k \le m_i. \tag{4.24}$$

Again, note that the traces of the new edge and face functions are the same as for the original ones and thus the global conformity of the finite element space is conserved.

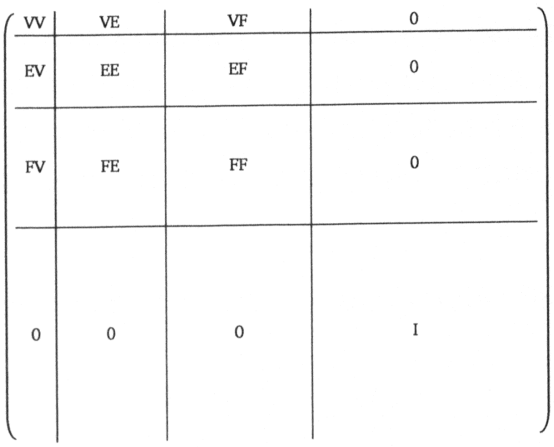


Figure 4.2: Block structure of the stiffness matrix.

Using the same reasoning as in (4.20) it is easy to show that

$$a_{K_i}(\psi_j^e, \psi_{i,l}) = 0 \text{ for all } j = 1, \ldots, E_i,\ l = 1, \ldots, m_i,\ i = 1, \ldots, M, \tag{4.25}$$

and also

$$a_{K_i}(\psi_j^s, \psi_{i,l}) = 0 \text{ for all } j = 1, \ldots, S_i,\ l = 1, \ldots, m_i,\ i = 1, \ldots, M. \tag{4.26}$$

Hence, the blocks EB, BE, FB, and BF were eliminated as well. The resulting stiffness matrix structure is shown in Figure 4.2. This means that we do not have to assemble any of the entries that arise from the bilinear form, where at least one of the arguments is a bubble eigenfunction. As we can see, the size of the problem drops considerably and thus leads to great savings in both memory and CPU time.

4.2 Schur complement, static condensation of degrees of freedom, and their relation to the generalized eigenproblem-based basis functions

As we showed in the previous section the utilization of the set of generalized eigenfunction-based basis functions leads to a significant simplification in the stiffness matrix structure, thus reducing the amount of work needed for the stiffness matrix assembly.

A similar result may be obtained by using a wider-known technique, the so called *static condensation of degrees of freedom*. First, let us start with the definition of the Schur complement of the block U in the stiffness matrix A and then we proceed with some technical aspects of the static condesation process and its consequences.

Definition 4.1. Let us consider a matrix A that attains the block form

$$A = \begin{pmatrix} P & R \\ S & U \end{pmatrix}, \tag{4.27}$$

where $A \in \mathbb{R}^{(p+q)\times(p+q)}$, $P \in \mathbb{R}^{p\times p}$, $R \in \mathbb{R}^{p\times q}$, $S \in \mathbb{R}^{q\times p}$, and $U \in \mathbb{R}^{q\times q}$. Schur complement of the block U in A is the matrix $C \in \mathbb{R}^{p\times p}$ such that

$$C = P - RU^{-1}S. \tag{4.28}$$

It is well known how the Schur complement can be used to simplify the *hp*-FEM solution process. Nevertheless, let us present the procedure for the sake of completeness.

After the discretization of a PDE problem by *hp*-FEM we arrive to the linear system (2.21), where the block structure of the stiffness matrix A is presented in Figure 3.1. We can see that it perfectly matches the form we used to define the Schur complement in (4.27): it is enough to put

$$P = \begin{pmatrix} VV & VE & VF \\ EV & EE & EF \\ FV & FE & FF \end{pmatrix}, \quad R = \begin{pmatrix} VB \\ EB \\ FB \end{pmatrix},$$

$$S = \begin{pmatrix} BV & BE & BF \end{pmatrix}, \quad \text{and } U = \begin{pmatrix} BB \end{pmatrix}.$$

Recall that in our setting, i.e., when the bilinear form is symmetric, the stiffness matrix A is symmetric and $P = P^T$, $R = S^T$, and $U = U^T$. Moreover, the block U is block diagonal and hence easily invertible.

Let us rewrite linear system (2.21) using the block form of A as

$$\begin{pmatrix} P & R \\ R^T & U \end{pmatrix} \begin{pmatrix} \boldsymbol{y}_p \\ \boldsymbol{y}_q \end{pmatrix} = \begin{pmatrix} \boldsymbol{l}_p \\ \boldsymbol{l}_q \end{pmatrix}, \tag{4.29}$$

where $(\boldsymbol{y}_p, \boldsymbol{y}_q)^T = \boldsymbol{y}$ and $(\boldsymbol{l}_p, \boldsymbol{l}_q)^T = \boldsymbol{l}$. We multiply system (4.29) by the matrix

$$L = \begin{pmatrix} I & -RU^{-1} \\ 0 & U^{-1} \end{pmatrix} \tag{4.30}$$

from the left and we obtain a new system

$$\begin{pmatrix} P - RU^{-1}R^T & 0 \\ U^{-1}R^T & I \end{pmatrix} \begin{pmatrix} \boldsymbol{y}_p \\ \boldsymbol{y}_q \end{pmatrix} = \begin{pmatrix} \boldsymbol{l}_p - RU^{-1}\boldsymbol{l}_q \\ U^{-1}\boldsymbol{l}_q \end{pmatrix}. \tag{4.31}$$

Note how the Schur complement of U in A appeared in the top left block of the matrix in linear system (4.31). Now system (4.31) can be naturally decomposed into two systems

$$\begin{aligned} \boldsymbol{y}_p &= C^{-1}\left(\boldsymbol{l}_p - RU^{-1}\boldsymbol{l}_q\right), \\ \boldsymbol{y}_q &= U^{-1}\left(\boldsymbol{l}_q - R^T\boldsymbol{y}_p\right), \end{aligned} \tag{4.32}$$

where C denotes the Schur complement of U in A.

Let us have a brief look at the systems (4.32). Instead of solving the linear system (2.21) of the size $N = p + q$, we need to invert the $q \times q$ matrix U matrix and solve another linear problem of the size p. Moreover, since the only nonzero entries of U come from the bilinear form applied to the bubble functions associated to the same element (and hence, attains the above mentioned block diagonal form), it can be inverted during the actual stiffness matrix assembly elementwise. This process is called the *static condensation of degrees of freedom* (see [71] for more information on this topic).

It is easy to show that the Schur complement of a stiffness matrix A resulting from an *hp*-FEM discretization of a PDE problem is independent of the concrete choice of bubble functions (see [74], Lemma 6.1.1). Situation gets a little bit more complicated if we use the complete eigenproblem-based basis (i.e., eigenproblem-based vertex, edge, face and bubble functions). However, it is still possible to show that the static condensation of degrees of freedom leads to the same stiffness matrix as the utilization of generalized eigenproblem-based basis functions. We prove this result in the following lemma.

Lemma 4.2. *Let $V_{h,p}$ be a finite element space and N be its dimension. Consider a basis of $V_{h,p}$ in the form*

$$\mathcal{B} = \{\varphi_1, \varphi_2, \ldots, \varphi_p, \varphi_1^b, \varphi_2^b, \ldots, \varphi_q^b\}, \tag{4.33}$$

such that $p+q=N$. The basis functions φ_i, $i=1,\dots,p$, stand for the vertex, edge and face basis functions, whereas φ_i^b, $i=1,\dots,q$, represent the bubble functions. Next, let $a(u,v)$ be a symmetric bilinear form. Recall that the stiffness matrix $A\in\mathbb{R}^{N\times N}$ corresponding to the bilinear form $a(\cdot,\cdot)$ and the basis $\mathcal{B}$ is defined as

$$A=\begin{pmatrix} P & R \\ R^T & U \end{pmatrix}, \quad \text{where} \begin{cases} P\in\mathbb{R}^{p\times p}, & P=\{p_{ij}\}_{i,j=1}^{p}, \\ R\in\mathbb{R}^{p\times q}, & R=\{r_{ij}\}_{i=1,\dots,p,\ j=1,\dots,q}, \\ U\in\mathbb{R}^{q\times q}, & U=\{u_{ij}\}_{i,j=1}^{q}, \end{cases} \tag{4.34}$$

and the matrices P, R, and U are defined by

$$\begin{aligned} p_{ij}&=a(\varphi_j,\varphi_i), && i,j=1,\dots,p, \\ r_{ij}&=a(\varphi_j^b,\varphi_i), && i=1,\dots,p,\ j=1,\dots,q, \\ u_{ij}&=a(\varphi_j^b,\varphi_i^b), && i,j=1,\dots,q. \end{aligned} \tag{4.35}$$

Recall that the submatrix U is block diagonal.

Finally, let

$$\mathcal{B}_{\text{eigen}}=\{\psi_1,\psi_2,\dots,\psi_p,\psi_1^b,\psi_2^b,\dots,\psi_q^b\}, \tag{4.36}$$

be the generalized eigenproblem-based basis of $V_{h,p}$. We have already shown that the corresponding stiffness matrix A_{eigen} then attains the form

$$A_{\text{eigen}}=\begin{pmatrix} P_{\text{eigen}} & 0 \\ 0 & I \end{pmatrix}. \tag{4.37}$$

Then the Schur complement of U in A is identical to P_{eigen}, or

$$P-RU^{-1}R^T=P_{\text{eigen}}. \tag{4.38}$$

Proof. Let us start the proof by recalling the way how ψ_i and ψ_j^b were defined,

$$\psi_j^b=\sum_{k=1}^{q} z_{jk}\varphi_k^b, \quad j=1,\dots,q, \tag{4.39}$$

and

$$\psi_i=\varphi_i-\sum_{k=1}^{q} h_{ik}\psi_k^b, \quad i=1,\dots,p. \tag{4.40}$$

Here the coefficients z_{jk} were obtained by the solving the generalized eigenproblems (4.9) in all mesh elements and we may denote the matrix of coefficients $\{z_{ij}\}_{i,j=1}^{q}$ by $Z \in \mathbb{R}^{q\times q}$. The fundamental property of Z is that

$$I = ZUZ^T, \tag{4.41}$$

since

$$\begin{aligned}\delta_{ij} = a(\psi_j^b, \psi_i^b) &= a\left(\sum_{k=1}^{q} z_{jk}\varphi_k^b, \sum_{l=1}^{q} z_{il}\varphi_l^b\right) \\ &= \sum_{k=1}^{q}\sum_{l=1}^{q} z_{jk}z_{il}\underbrace{a(\varphi_k^b, \varphi_l^b)}_{u_{lk}} = \left(ZUZ^T\right)_{ij}.\end{aligned} \tag{4.42}$$

Remember that the coefficients h_{ik} are the result of modifying the vertex, edge, and face functions described in (4.18), (4.22) and (4.24), generally

$$h_{ik} = a(\varphi_i, \psi_k^b), \quad i = 1,\ldots,p,\ k = 1,\ldots,q. \tag{4.43}$$

At this point it is useful to show that the upper right and lower left blocks of the matrix A_{eigen} are indeed zero. The entries of the blocks are in the form

$$\begin{aligned}a(\psi_i, \psi_j^b) &= a\left(\varphi_i - \sum_{k=1}^{q} h_{ik}\psi_k^b, \psi_j^b\right) \\ &= a(\varphi_i, \psi_j^b) - \sum_{k=1}^{q} h_{ik}\underbrace{a(\psi_k^b, \psi_j^b)}_{=\delta_{kj}} \\ &= \underbrace{a(\varphi_i, \psi_j^b)}_{=h_{ij}} - h_{ij} = 0.\end{aligned} \tag{4.44}$$

We denote the p by q matrix $\{h_{ik}\}_{i=1,\ldots,p,\ k=1,\ldots,q}$ by H. It follows easily from (4.43) that

$$H = RZ^T, \tag{4.45}$$

since

$$\begin{aligned}h_{ik} = a(\varphi_i, \psi_k^b) &= a\left(\varphi_i, \sum_{l=1}^{q} z_{kl}\varphi_l^b\right) = \sum_{l=1}^{q} z_{kl}\underbrace{a(\varphi_i, \varphi_l^b)}_{r_{il}} \\ &= \left(RZ^T\right)_{ik}.\end{aligned} \tag{4.46}$$

Let us take a closer look at the entries of the submatrix $P_{\text{eigen}} = \{p_{ij}^{\text{eigen}}\}_{i,j=1}^{p}$,

$$\begin{aligned} p_{ij}^{\text{eigen}} &= a(\psi_j, \psi_i) = a\left(\psi_j, \varphi_i - \sum_{k=1}^{q} h_{ik}\psi_k^b\right) \\ &= a(\psi_j, \varphi_i) - \sum_{k=1}^{q} h_{ik} \underbrace{a(\psi_j, \psi_k^b)}_{=0 \text{ by } (4.44)} \\ &= a\left(\varphi_j - \sum_{k=1}^{q} h_{jk}\psi_k^b, \varphi_i\right) = \underbrace{a(\varphi_j, \varphi_i)}_{=p_{ij}} - \sum_{k=1}^{q} h_{jk} a(\psi_k^b, \varphi_i) \\ &= p_{ij} - \sum_{k=1}^{q} h_{jk} a\left(\sum_{l=1}^{q} z_{kl}\varphi_l^b, \varphi_i\right) = p_{ij} - \sum_{k=1}^{q}\sum_{l=1}^{q} h_{jk} z_{kl} \underbrace{a(\varphi_l^b, \varphi_i)}_{=r_{il}}. \end{aligned}$$

From the previous expression it follows directly that

$$P_{\text{eigen}} = P - RZ^T H^T, \tag{4.47}$$

since

$$\left(RZ^T H^T\right)_{ij} = \sum_{l=1}^{q}\sum_{k=1}^{q} r_{il} z_{kl} h_{jk}. \tag{4.48}$$

From (4.41) we get an equivalence

$$I = ZUZ^T \Leftrightarrow Z^T = U^{-1}Z^{-1}. \tag{4.49}$$

Next, by (4.45), we get

$$H = RZ^T \Leftrightarrow R^T = Z^{-1}H^T. \tag{4.50}$$

We may conclude the proof by using (4.47), (4.49), and (4.50),

$$P_{\text{eigen}} = P - RZ^T H^T = P - RU^{-1}Z^{-1}H^T = P - RU^{-1}R^T. \tag{4.51}$$

□

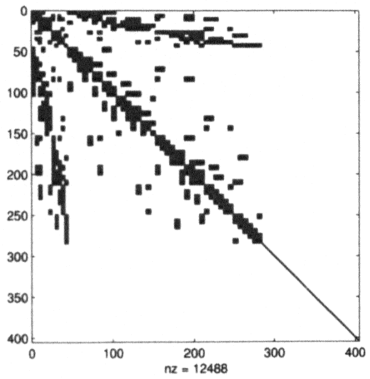
	Matrix size		Nonzero entries		cond(A)
Degree	(standard)	(mesh-eigen)	(standard)	(mesh-eigen)	(mesh-eigen)
4	183	153	4003	3761	$2.265 \cdot 10^2$
5	404	284	16024	12488	$2.409 \cdot 10^3$
6	755	455	49547	30383	$2.659 \cdot 10^4$
7	1266	666	127056	63086	$3.274 \cdot 10^5$
8	1967	917	283453	116461	$4.341 \cdot 10^6$
9	2888	1208	571666	198244	$6.029 \cdot 10^7$
10	4059	1539	1066687	316685	$8.657 \cdot 10^8$

Table 4.1: Comparison of stiffness matrix sizes, nonzero entries, and condition numbers for standard *hp*-FEM using Lobatto-based shape functions and non-affine *hp*-FEM using vertex, edge, face, and bubble generalized eigenproblem-based basis functions for the problem presented in Section 3.4.

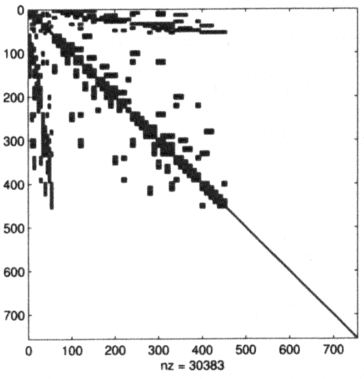
4.3 Numerical example

We show the numerical properties of the generalized eigenproblem-based basis functions on the same example as in Section 3.4. To emphasize the effect of the utilization of eigenproblem-based basis functions we present several properties of the stiffness matrices for problem (3.42). First, the sparsity patterns of the stiffness matrices for polynomial degrees $p = 5, 6, \ldots, 10$ are presented in Figure 4.3.

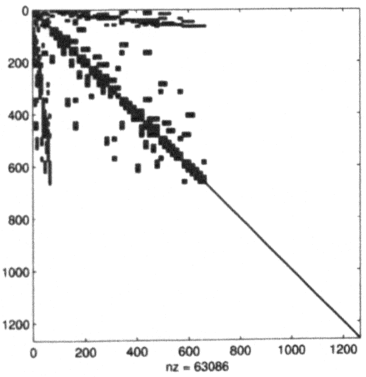
We should mention here that Figure 4.3 is illustrative only, since it shows the stiffness matrices after the complete assembly using all basis functions was performed. However, as we mentioned before, the entries of the stiffness matrices corresponding to evaluation of the bilinear form such that one of the arguments is a bubble shape function do not have to be assembled at all. We present a comparison of the resulting linear problem sizes as well as the number of corresponding nonzero entries and the condition numbers in Table 4.1.



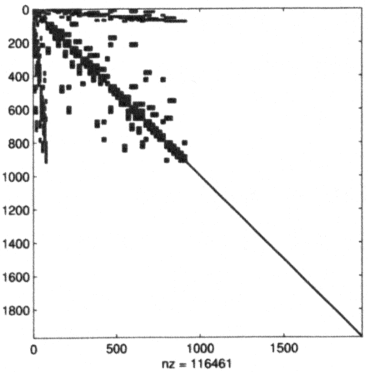
(a) $p = 5$



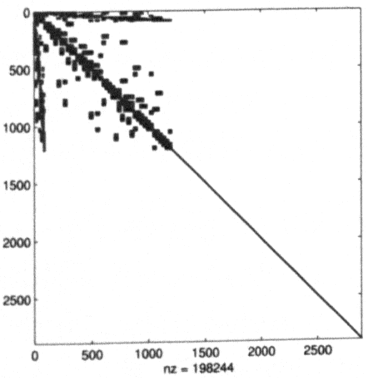
(b) $p = 6$



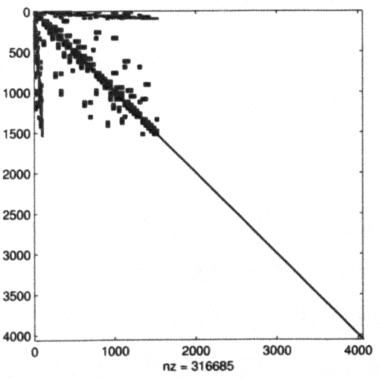
(c) $p = 7$



(d) $p = 8$



(e) $p = 9$



(f) $p = 10$

Figure 4.3: Sparsity patterns of stiffness matrices for the problem (3.42) using uniform polynomial degree $p = 5, 6, \ldots, 10$.

4.4 Conclusion

The utilization of generalized eigenproblem basis functions brings several positive aspects to the *hp*-FEM solution process. Let us now summarize the benefits and compare this approach to the aforementioned static condensation of degrees of freedom.

- The size of the problem gets fundamentally reduced in both cases. Both methods lead to mathematically equivalent linear systems.
- One additional benefit of the generalized eigenproblem-based basis functions is that the BB block of the mass matrix gets diagonalized too. This may become very useful when solving parabolic problems. If desirable, we can use the same approach to reduce the size of the mass matrix instead of the stiffness matrix.
- From the theory presented in this chapter it follows that the generalized eigenproblem-based basis functions can only be used in the case when the bilinear form $a(\cdot,\cdot)$ is symmetric. The static condensation of degrees of freedom does not impose this requirement.
- The generalized eigenproblem-based basis functions can be stored for subsequent computations using the same discretization of the original problem, e.g., in the case when only the right-hand side changes. On the other hand, for the static condensation of degrees of freedom we may store the inverse of the matrix U as well.
- The price we pay for the benefits is the need to solve a generalized eigenproblem (or to perform a matrix inversion of the same size in the case of static condensation of degrees of freedom) on each mesh element K_i, $i = 1, \ldots, M$. However, since both problems are local, the whole process can be easily parallelized to achieve better performance.

CHAPTER 5

3D hp-FEM system HERMES

One of the main goals of this work was to construct an *hp*-FEM software useful for solving problems in three spatial dimensions that exhibit multiscale or singular behavior. Another reason for the construction of such a software was the demand for a program that could be used to verify some aspects of mathematical theories we developed.

Most of the work was done when contributing to the project HERMES[1] developed by the *hp*-FEM group at the University of Texas at El Paso. The main efforts of the project are focused on the construction of a modular hierarchic *hp*-FEM platform for the solution of nonlinear multi-physics coupled problems.

The chapter contains original results obtained by the author of this thesis. We also have to acknowledge the work of Martin Lazar, the author of the mesh module, and Pavel Kús, who added a significant amount of functionality in all modules, added automatic *hp*-adaptivity, and who maintains the whole package since September 2006.

[1] See `http://hpfem.math.utep.edu/wwwhermes`.

5.1 Modules of HERMES

From the very beginning we tried hard to build the software from several blocks – modules – that would be easily extensible and maintainable. This has several advantages, most notably that different modules may be reused for different projects.

5.1.1 Mesh

Mesh generation

Before we start any FEM computation, we have to define our computational domain and cover it with a finite element mesh. The process of mesh generation is not included in HERMES and so we have to solve this task separately. We use several mesh generators to obtain three dimensional finite element meshes, GMSH [30] and TetGen [56] being the most frequently used.

To facilitate fast mesh loading and preprocessing we developed a mesh format based on a scientific standard HDF5[2] format. The main advantages of this format are its versatility, portability, excellent support on majority of platforms and focus on performance. Of course, to allow for the utilization of different mesh generators we included simple scripts converting mesh data from formats used by various mesh generators to HDF5.

Mesh storage and preprocessing

So far we can use tetrahedral and hexahedral meshes in HERMES. Although it would be relatively easy to add support for other types of elements (e.g. prisms), there are several important reasons why we will probably stay with the current functionality and focus on hexahedra in future. To start off, probably the most important goal of HERMES is the development of *hp*-adaptive schemes. Once these algorithms are ready to use the necessity of hybrid meshes decreases. Next, while it is relatively easy to perform tessellations of 3D domains by hexahedra, whereas the same task for tetrahedra (and consequently for prisms, if we need to use hybrid meshes) is much more demanding

[2] See `http://hdf.ncsa.uiuc.edu/`.

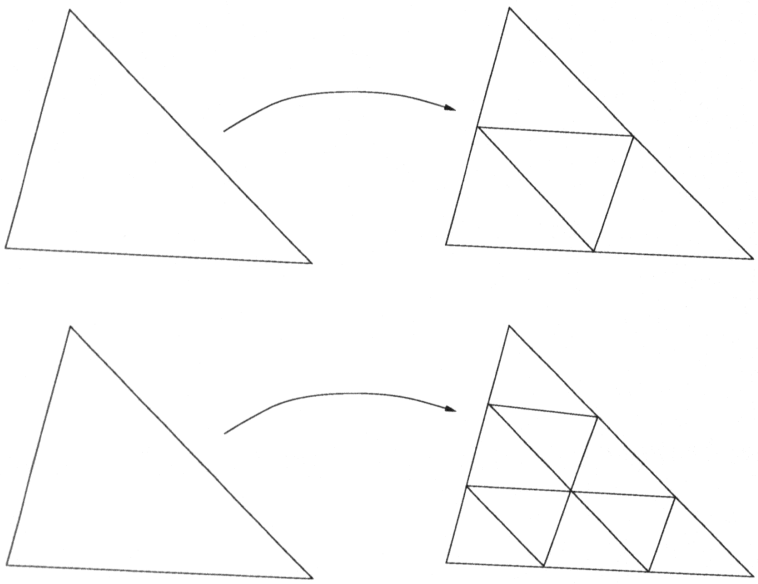


Figure 5.1: Refinement of a triangle into n^2 similar subtriangles.

and less understood. Finally, the adaptive algorithms for tetrahedral meshes usually result in degeneration of refined tetrahedra, which is known to be undesirable for the FEM solution process. Let us expose this problem on an easy example.

Example. Consider a triangle in a two-dimensional finite element mesh. It is relatively easy to refine it, since the division of a triangle into n^2 triangles is an easy task (see Figure 5.1). Moreover, the resulting triangles are similar to the original ones, and thus a refinement of a nonobtuse triangle produces nonobtuse subtriangles.

On the other hand, even the easiest division of a tetrahedron into eight subtetrahedra by connecting midpoints of edges is a non-unique operation and may produce tetrahedra with obtuse dihedral angles (see Figure 5.2). By cutting away the four subtetrahedra containing the vertices of the original tetrahedron we are left with a union of two pyramids, that can be split into tetrahedra into four tetrahedra in three different ways, depending on which spatial diagonal we use for the division. In any case we are likely to get a tetrahedron with obtuse dihedral angles, since the spatial diagonal we used for the refinement becomes an edge shared by four subtetrahedra. This means

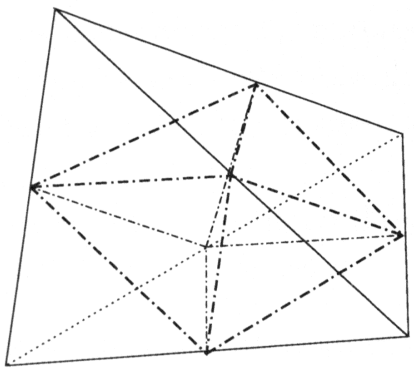


Figure 5.2: Division of a tetrahedron into eight tetrahedra using edge midpoints.

that at least one of the dihedral angles corresponding to this edge is greater than $\pi/2$, unless all of them are equal to $\pi/2$.

The mathematical community shares a great interest in the topic of tetrahedral refinement. However, the techniques work only under some special conditions or they also produce obtuse partitions. For the most notable works in this field let us mention [40, 39, 37, 38, 14, 77] and references therein. Recently there has been a method presented that introduces a stable refinement of simplices. See [15] for details.

The last reason why we favor hexahedra over tetrahedra and prisms is that the adaptive process on hexahedral meshes is well understood and allows for anisotropic refinement as well.

The mesh structures were designed to represent a general 2D or 3D mesh and all the geometrical information needed for the FEM solution process. The main data structure is a collection of elements. To be able to retrieve information about vertices, edges and faces effectively we store them in hashtable-like structures that provide $O(1)$ retrieval complexity.

5.1.2 Reference elements and shape functions

We use tetrahedral and hexahedral reference domains exactly as defined in (2.23) and (2.24). To equip these domains with polynomial spaces we need to define a suitable basis consisting of shape functions. We have implemented

the monomial-, Lobatto- and generalized eigenproblem-based shape functions up to tenth polynomial order for both reference domains. The set of shape functions used to generate the reference element polynomial space can be replaced or enriched easily.

Another feature we would like to mention is the precalculated set of shape functions. This is useful mainly for two reasons:

- First, no matter what set of shape functions you use you are always interested in the values (and derivatives) of the shape functions in quadrature points only. This means we can precalculate the values and derivatives of the shape functions and then reuse these values according to our needs.
- Second, the generalized eigenproblem-based set of shape functions is handicapped in the sense that these functions are usually expressed as expansions with respect to the initial basis of shape functions. This slows down the evaluation of these shape functions and again it is desirable to precalculate the values and derivatives of the shape functions to speed up the solution process.

5.1.3 Elliptic problem solver

The finite element mesh needs to be equipped with a suitable finite element space. The modular structure of HERMES allows to build finite element spaces conforming to various functional spaces, though the only space currently implemented is the H^1-conforming finite element space, suitable for discretization of elliptic PDEs.

The elliptic problem solver allows to discretize equations in the form of our model problem

$$\begin{aligned} -\nabla \cdot (a_1(\boldsymbol{x})\, \nabla u) &= f \quad \text{in } \Omega, \\ u|_{\Gamma_D} &= g_D \quad \text{on } \Gamma_D, \\ \frac{\partial u}{\partial \nu} &= g_N \quad \text{on } \Gamma_N. \end{aligned}$$

The equation is then discretized using the given mesh and polynomial degree on each mesh element.

5.1.4 Quadrature

Unlike for standard low-order FEM, a significant amount of CPU time during the solution process is spent on the stiffness matrix assembly. Recall that the entries of the stiffness matrix are defined as integrals. These integrals are obtained employing a set of numerical quadrature rules, hence the necessity for fast and and accurate quadrature rules is obvious.

To satisfy these needs we implemented Gauss quadrature tables up to twentieth polynomial order present in [62]. Note that to be able to evaluate boundary and volume integrals on reference domains we need 1D, 2D and 3D quadrature rules for the most common reference domains. The Gauss quadrature points and weights are known analytically in 1D. Unfortunately, this is not true for 2D and 3D on noncartesian product elements (i.e., triangles, tetrahedra, prisms, etc.). In these cases the points and weights have to be computed numerically by solving systems of nonlinear algebraic equations. However, these methods often tend to diverge, or the weights converge to negative numbers or the quadrature points converge to points outside of the reference domains.

The task to calculate Gauss quadrature points and weights in 3D for high degrees of accuracy is still an open problem.

5.1.5 Algebraic problem solvers

In HERMES we implemented support for today's state-of-the-art iterative and direct solvers. The iterative solvers are represented by the PETSc library (see [12]), allowing for easy matrix and vector representation in parallel. We use the routines included in UMFPACK (see [21]) for the direct solution of algebraic systems.

5.1.6 Detailed description of algorithms

The solution process is divided into several steps and is described using pseudocode in Algorithm 1. We will describe the most important steps in more detail in the following paragraphs.

Algorithm 1 *hp*-FEM solver workflow.

```
1: program Elliptic hp-FEM solver
2:     Mesh initialization and preprocessing.
3:     DOF assignment and numbering.
4:     Computation of the boundary condition projection.
5:     Stiffness matrix and load vector assembly.
6:     Algebraic system solution.
7:     Error estimation.
8:     Output.
9: end
```

Mesh file description

We mentioned earlier that the mesh used for the computation is stored in the HDF5 format. First let us describe briefly the internal structure of the HDF5 file describing the mesh. Any HDF5 file consists of hierarchically organized entities. Our file consits of two kinds of entities, called *groups* ("folders") and *datasets* (data tables). The mesh file may contain several *groups*, each containing logically corresponding data. Allowed groups are characterized by the keywords *vertices*, *triangles*, *quads*, *tetrahedra*, and *hexahedra*.

The *vertices* group contains one dataset only, named *coordinates*. The dataset contains the coordinates of all the vertices present in the mesh. The rest of the groups shares a similar structure since each of them describes a set of geometric objects. All of them must contain two datasets, one of them named *connections* and the latter named *markers*. These two datasets must contain the same number of rows (i.e., each row in one dataset corresponds to the same row in the latter dataset). The *connections* dataset specifies the numbers of vertices forming the geometric object (either an element or a boundary face). The *markers* dataset contains a marker for each geometric object. For example for the *triangles* and *quads* groups the markers are used to identify the boundary condition corresponding to the boundary face.

To make things clear, let us present a short example. The *hexahedra* group must contain a mandatory *connections* dataset. The number of rows in this dataset is equal to the number of hexahedral elements present in the mesh.

There are eight columns present in the dataset (recall that each hexahedron has eight vertices). The eight numbers in each row specify the vertices that form the hexahedron (the numbers are "pointers" to the *coordinates* dataset in the *vertices* group).

For a valid 3D mesh one has to specify the *vertices* group, at least one of the *tetrahedra* or *hexahedra* groups, and corresponding groups describing boundary faces.

Dirichlet lift and the boundary condition projection

One of the tasks we need to do before the actual stiffness matrix and load vector assembly is the projection of the boundary condition into a suitable polynomial space and creating a suitable *Dirichlet lift* (see Section 2.1.2). We construct the Dirichlet lift using the *projection-based interpolation*. This procedure imposes several advantageous properties:

- It is *local*, i.e., the projection-based interpolant is constructed element-wise and we avoid time consuming global computation.
- The procedure preserves *global conformity* of the projection-based interpolant.
- The projection-based interpolation is *optimal* in the sense that the distance of the interpolant from the original function is minimal in an appropriate norm.

Projection-based interpolation is explained in detail in [58], Chapter 3. The monography also addresses other problems connected to the projection-based interpolation, for example the choice of the optimal norm for the measurement of function distances, the meaning of global conformity for different finite element spaces, and others.

We need to create a function $G \in H^1(\Omega)$ such that ideally $G|_{\Gamma_D} = g_D$ in the sense of traces. Usually this condition cannot be met and we impose a weaker condition $\left|\left|G|_{\Gamma_D} - g_D\right|\right| \rightarrow \min$ in a suitable norm.

Let us describe the process briefly. The first approximation of the Dirichlet lift is obtained by interpolating the Dirichlet boundary condition by a set of

vertex functions belonging to the vertices on the Dirichlet boundary. Let us denote the function resulting from the interpolation by $G^v_{h,p}$. Next we project the difference $g_D - G^v_{h,p}\Big|_{\Gamma_D}$ onto a space spanned by the traces of edge functions belonging to all edges corresponding to the Dirichlet boundary. The function is then denoted by $G^e_{h,p}$. Finally, we project the difference $g_D - \left(G^v_{h,p} - G^e_{h,p}\right)\Big|_{\Gamma_D}$ onto a space spanned by the traces of face functions belonging to all faces corresponding to the Dirichlet boundary to obtain the function $G^s_{h,p}$. The Dirichlet lift is then constructed as the sum $G^v_{h,p} + G^e_{h,p} + G^s_{h,p}$.

Stiffness matrix and load vector assembly

The stiffness matrix and load vector assembly is carried out in the standard way and is described in Algorithm 2.

5.2 Numerical examples

In the very last section of this chapter we present two numerical examples. These are more proof-of-concept problems rather than serious industrial calculations. However, they show why the *hp*-FEM is important for problems exhibiting singularities and how its utilization can save both computational resources and time.

Note that although the presented examples are not computed using an automatic *hp*-adaptive process, such an algorithm was developed during the work on this thesis by other members of the workgroup and some preliminary results were published in [42] and [43]. The topic will be completely covered in [41].

5.2.1 Distribution of electrostatic potential in the Fichera corner domain

In this problem we analyze the distribution of electric potential in the Fichera corner domain. This problem is well known for exhibiting a strong singularity at the reentrant corner.

Let

$$\Omega = ((-1,1)\times(-1,1)\times(-1,1)) \setminus ([0,1]\times[0,1]\times[0,1])$$

Algorithm 2 Stifness matrix and load vector assembly.

for all elements K_i in $\mathcal{T}_{h,p}$ **do**
  Calculate reference map $\boldsymbol{x}_{K_i}$, its inverse and Jacobian.
  Identify all the shape functions $\hat{\varphi}_j$, $j = 1, \ldots, n_i$,
    up to the elements polynomial order of approximation p_i.
  for all shape functions $\hat{\varphi}_j$, $j = 1, \ldots, n_i$ **do**
    Find o_j (shape function orientation).
      //I.e., determine the orientation adjustment map $\boldsymbol{x}^{o_j}$ if needed.
    Find m_j (the corresponding global degree of freedom index).
      //I.e., $\varphi_{m_j}|_{K_i} = \boldsymbol{\Phi}^1_{K_i}(\hat{\varphi}_j \circ \boldsymbol{x}^{o_j})$, where φ_{m_j} is a global basis function.
      //Assign $m_j = -1$ if φ_{m_j} does not represent a degree of freedom.
    Find c_j (the corresponding coefficient).
      //Assign $c_j = 1$ if φ_{m_j} represents a degree of freedom.
      //Assign appropriate precomputed c_j if φ_{m_j} is used
      //to reconstruct the Dirichlet lift.
  end for
  for $j = 1 \ldots n_i$ **do**
    for $k = 1 \ldots n_i$ **do**
      $s = c_k c_j a_{K_i}\left(\varphi_{m_k}, \varphi_{m_j}\right)$ //compute the bilinear form
      if $m_k \geq 0$ **then** //φ_{m_k} represents a degree of freedom
        $A_{m_k m_j} = A_{m_k m_j} + s$ //add s to the stiffness matrix entry
      else //φ_{m_k} does not represent a degree of freedom
        $l_{m_j} = l_{m_j} - s$ //subtract s from the load vector entry
      end if
    end for
    $v = l(\varphi_j)$ //compute the right hand side linear form contribution
    $l_{m_j} = l_{m_j} + v$ //add v to the load vector entry
  end for
  //Similarly for the surface part of the linear form.
end for

be the computational domain depicted in Figure 3.3a. We solve the problem

$$-\Delta u = f \quad \text{in } \Omega, \tag{5.1}$$

$$u = u_D \quad \text{on } \partial\Omega, \tag{5.2}$$

where f and u_D are chosen to comply with the exact solution

$$u(x_1, x_2, x_3) = (x_1^2 + x_2^2 + x_3^2)^{1/4}, \tag{5.3}$$

i. e.,

$$f(x_1, x_2, x_3) = -\frac{3}{4}(x_1^2 + x_2^2 + x_3^2)^{-3/4}. \tag{5.4}$$

The exact solution u is continuous in the entire computational domain. The gradient of u has a singularity at $(0,0,0)$, yet the H^1-norm of u is finite. See Figure 5.3 for the graph of the exact solution and the magnitude of its gradient.

We perform two computations: one with quadratic finite elements and one with higher-order elements. In each case we present the best performing combination of spatially adapted finite element mesh and associated polynomial degrees.

Quadratic elements

First we use the tetrahedral mesh shown in Figure 5.4. It contains 130630 tetrahedra and by using quadratic elements we obtain 157260 degrees of freedom. Using this mesh, we obtained the relative H^1 norm of the error

$$\frac{||e_{h,p}||_{H^1}}{||u||_{H^1}} = \frac{||u_{h,p} - u||_{H^1}}{||u||_{H^1}} = 0.682\%. \tag{5.5}$$

Higher-order elements

Next we use the tetrahedral mesh shown in Figure 5.5. The polynomial degrees used are represented by different colors. The mesh contains 369 elements and we obtain 3387 degrees of freedom total. Using this mesh we obtained the H^1 relative norm of the error

$$\frac{||e_{h,p}||_{H^1}}{||u||_{H^1}} = \frac{||u_{h,p} - u||_{H^1}}{||u||_{H^1}} = 0.588\%. \tag{5.6}$$

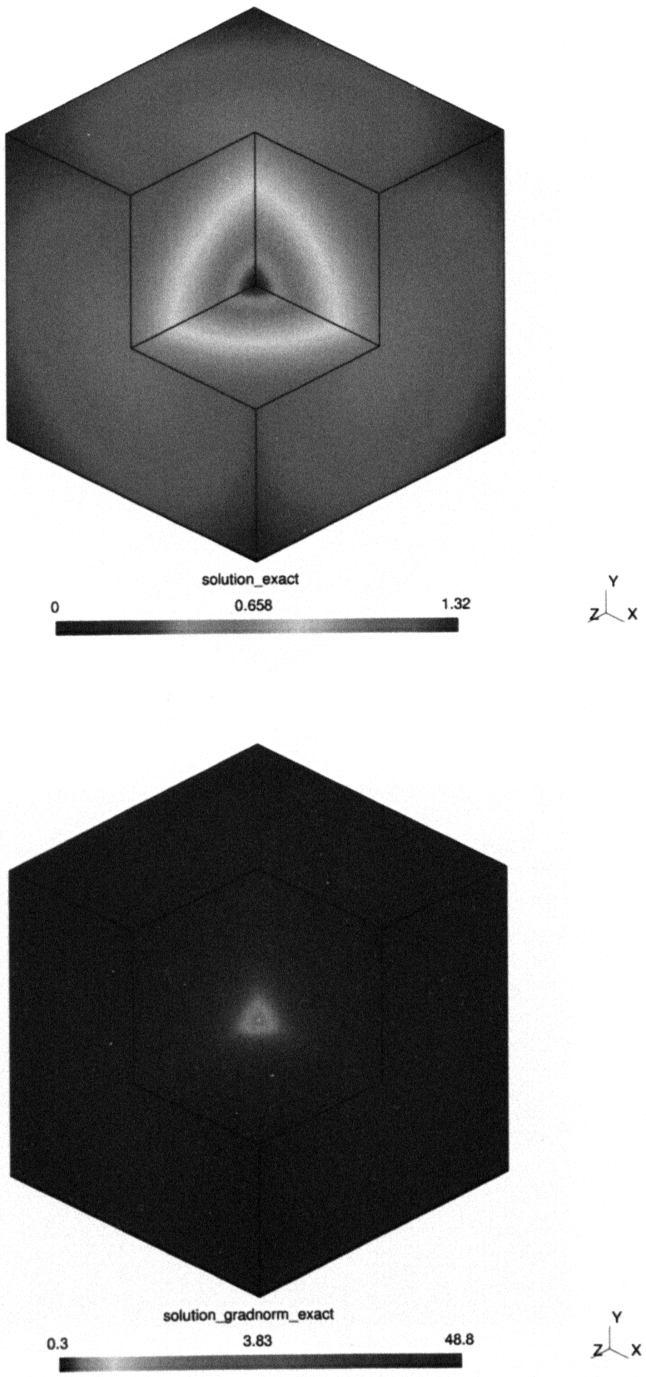


Figure 5.3: Exact solution of the problem (5.1) (top), solution gradient norm with a singularity at the re-entrant corner (bottom, note the logarithmic scale).

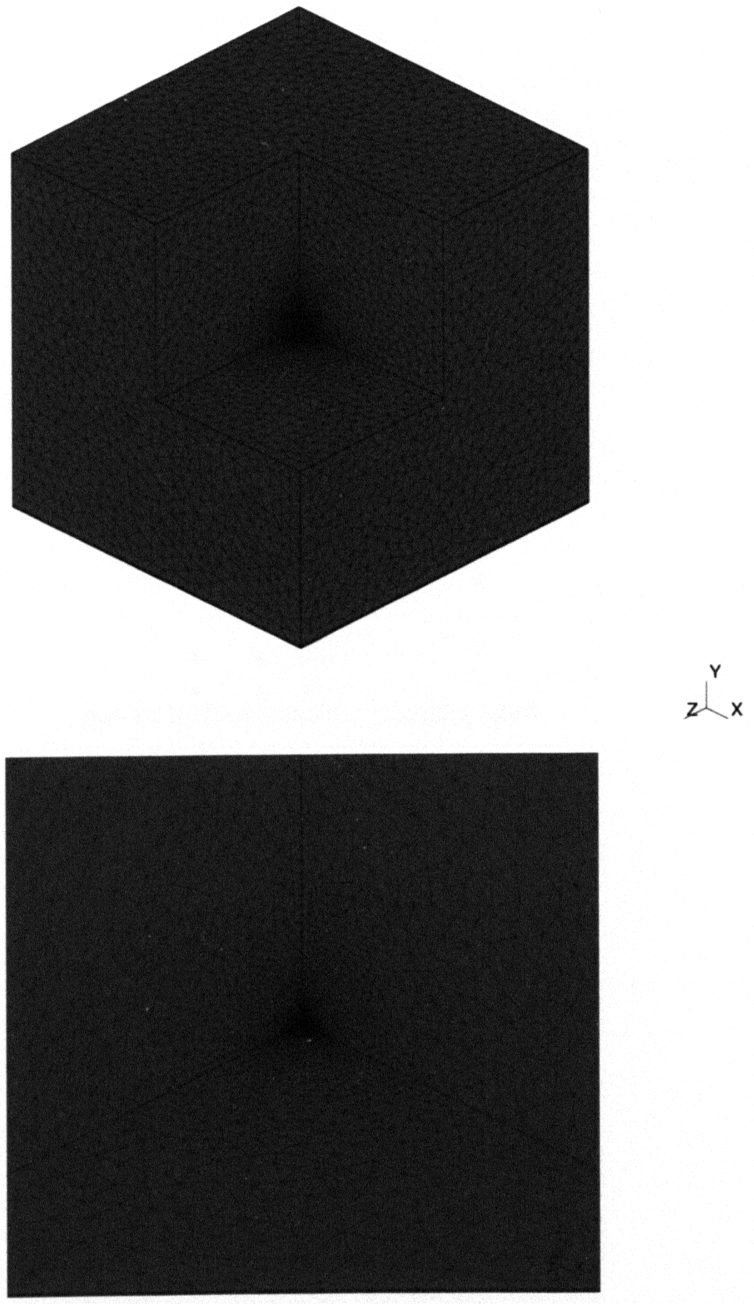


Figure 5.4: Mesh used to obtain finite element approximation to the solution of problem (5.1) using quadratic elements (top), zoom to the singularity corner (bottom).

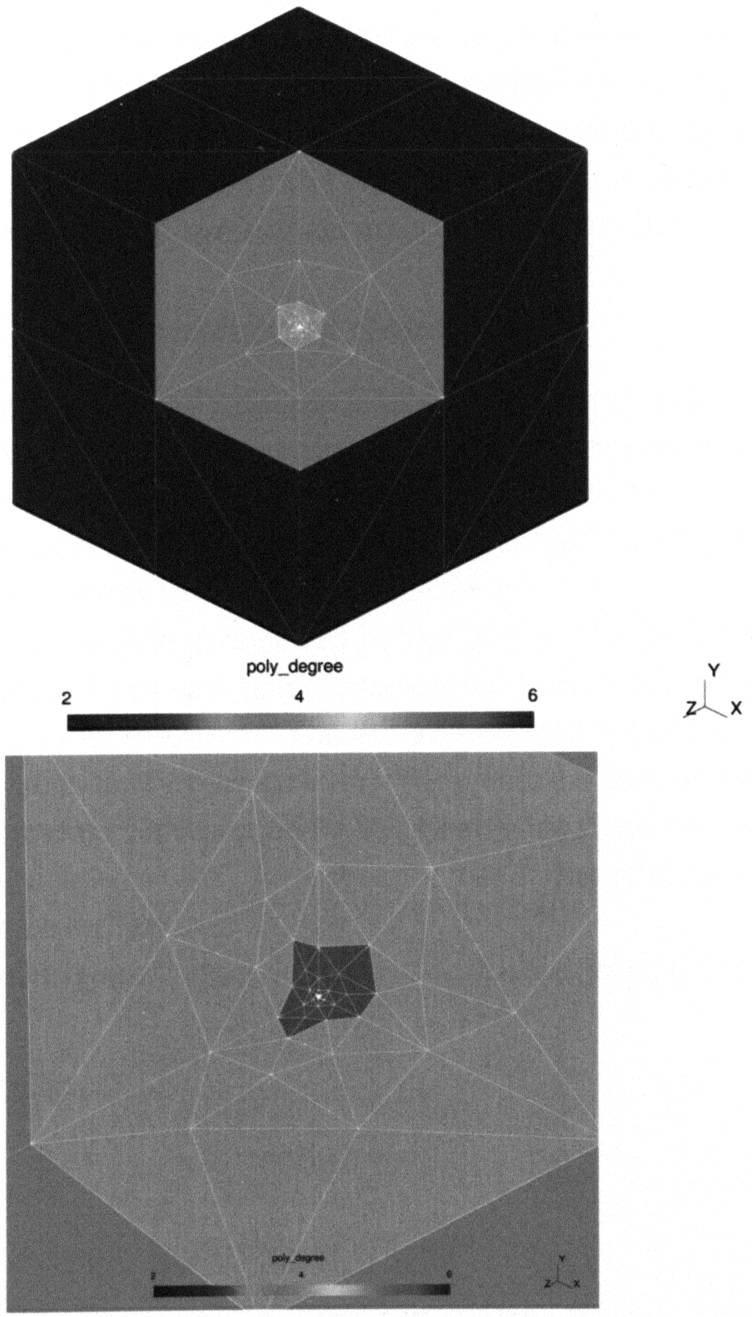


Figure 5.5: Mesh used to obtain finite element approximation to the solution of problem (5.1) using higher-order elements (top), zoom to the singularity corner (bottom).

Conclusion

While using almost fifty times less degrees of freedom compared to standard FEM with quadratic elements, by the *hp*-FEM we were able to obtain a significantly more accurate result. According to our experience with 2D problems, we can expect even better performance when utilizing automatic *hp* adaptivity. Preliminary results published in [42] and [43] show that it is almost impossible for standard FEM to reach levels of accuracy obtained by the *hp*-FEM.

5.2.2 Electric potential between two charged electrodes

Next we present a more practical example exhibiting strong singularities in the norm of the solution gradient. The setting is a well-known physical problem of two pyramidal electrodes with their tips pointing to each other. In practice a similar device is connected to a high voltage source and used to ionize the air which consequently leads to an artificial lightning.

The computational domain is depicted in Figure 5.6. The problem is symmetric and hence, we use just one half of the actual domain for the computation to save computational resources. The top and the bottom of the computational domain represent the two electrodes. We prescribe Dirichlet boundary condition on the boundary represented by the electrodes since the electric potential of the electrodes is known. We prescribe the homogeneous Neumann boundary condition elsewhere.

Again, we are solving the Poisson equation for the electrostatic potential u,

$$-\Delta u = 0 \qquad \text{in } \Omega, \tag{5.7}$$

$$u = 0 \qquad \text{on } \Gamma_{D_a}, \tag{5.8}$$

$$u = 10000 \qquad \text{on } \Gamma_{D_b}, \tag{5.9}$$

$$\frac{\partial u}{\partial \nu} = 0 \qquad \text{on } \Gamma_N, \tag{5.10}$$

where ν is the unit outer normal to Γ_N, $\Gamma_{D_a} = \Gamma_{D_{p_0}} \cup \Gamma_{D_{\text{bottom}}}$, $\Gamma_{D_b} = \Gamma_{D_{p_1}} \cup \Gamma_{D_{\text{top}}}$, and $\Gamma_N = \Gamma_{N_1} \cup \Gamma_{N_2} \cup \Gamma_{N_3} \cup \Gamma_{N_4}$.

The norm of the gradient of the electrostatic potential u (the electrostatic field) exhibits a large singularity at the tips of the pyramidal electrodes (see Figures 5.7, 5.8, and 5.9). Unfortunately we do not know the exact solution

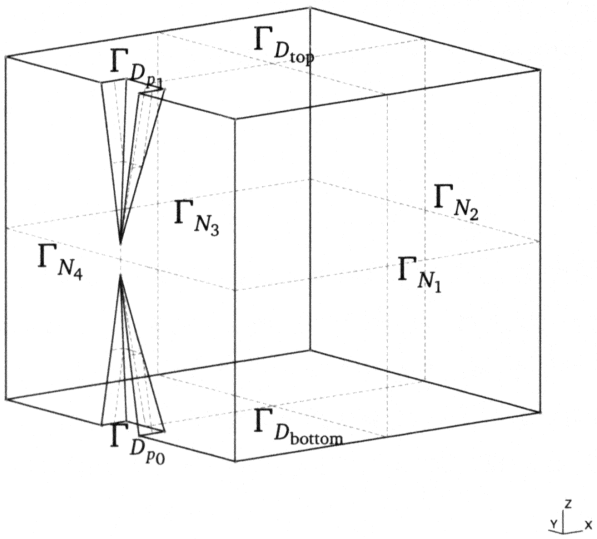


Figure 5.6: Computational domain for the problem of two charged pyramidal electrodes.

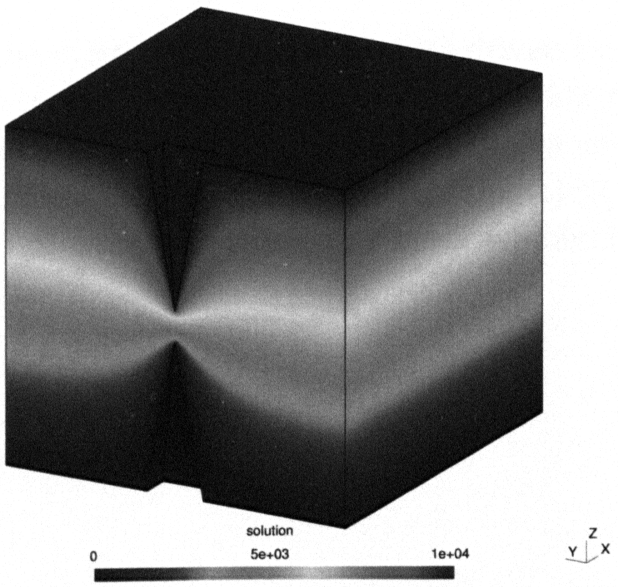


Figure 5.7: Approximate solution of the problem of two charged pyramidal electrodes.

for this problem and therefore we can only estimate the actual error. To come up with a fair error estimator we used the approximate solution computed on a once refined mesh with the polynomial degree incremented globally by one as our reference solution.

Quadratic elements

First let us perform the computation using a tetrahedral mesh of quadratic elements. Again, we performed several computations with different meshes and by comparing the distribution of the error norm we chose the best performing one. It is depicted in Figure 5.10 and contains 130774 elements. This gives us 175380 degrees of freedom total. Using this mesh we obtained relative H^1 norm of the error

$$\frac{||e_{h,p}||_{H^1}}{||u||_{H^1}} = \frac{||u_{h,p} - u||_{H^1}}{||u||_{H^1}} = 1.244\%. \tag{5.11}$$

Higher-order elements

Next we performed the mesh optimization process driven by the error of the approximate solution to choose the best performing *hp*-FEM mesh. We ended up with the mesh shown in Figure 5.11. Again, the polynomial degrees used are represented by different colors. The mesh contains 2182 elements which yields 18690 degrees of freedom. Using this mesh we obtained relative H^1 norm of the error estimate

$$\frac{||e_{h,p}||_{H^1}}{||u||_{H^1}} = \frac{||u_{h,p} - u||_{H^1}}{||u||_{H^1}} = 1.160\%. \tag{5.12}$$

Conclusion

Using the *hp*-FEM we were able to obtain a more accurate result while using almost ten times less degrees of freedom compared to standard FEM with quadratic elements. One might expect much bigger difference in the efficiency

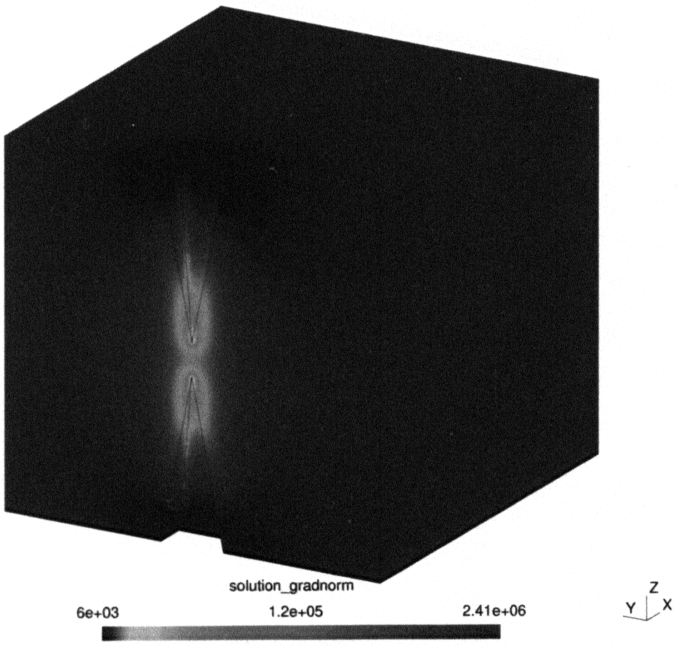


Figure 5.8: Approximate solution of the gradient norm for the problem of two charged pyramidal electrodes (note the logarithmic scale).

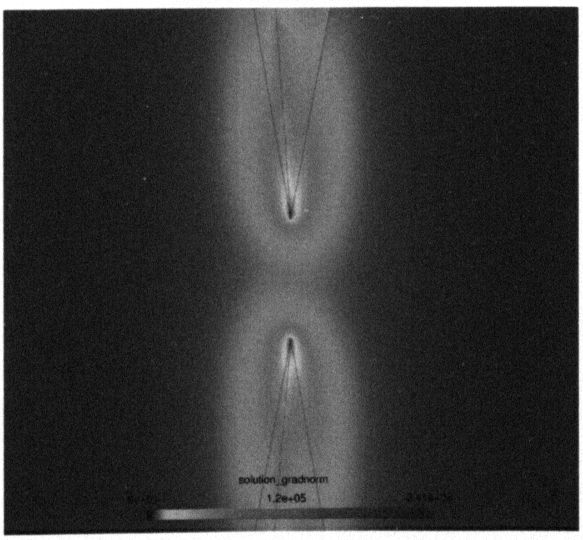


Figure 5.9: Norm of the gradient of the approximate solution zoomed to the tips of the pyramidal electrodes (note the logarithmic scale).

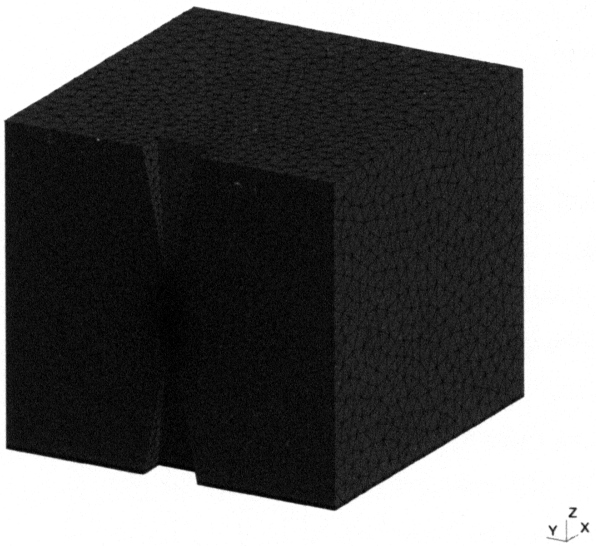


Figure 5.10: Mesh used to obtain finite element approximation of the solution of the problem (5.7) using quadratic elements.

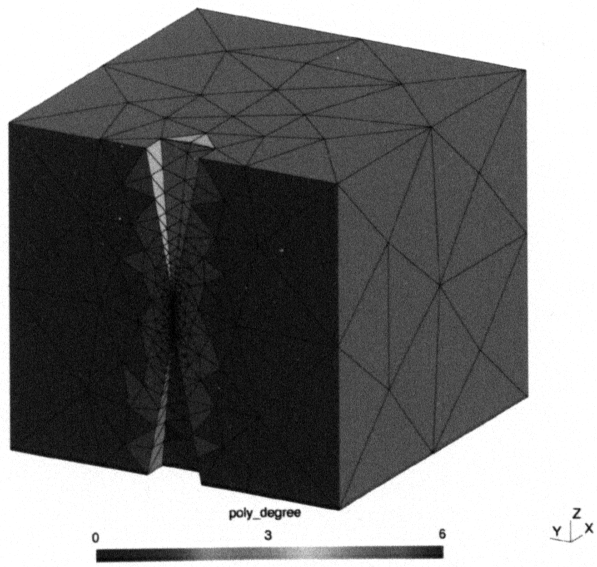


Figure 5.11: Mesh used to obtain finite element approximation of the solution of the problem (5.7) using higher-order elements.

of the two presented methods in this section. However, there is one important aspect that prevented us from delivering a better comparison. The problem is the method of error estimation we used for the computation with low-order elements.

If we use a mesh with hundreds of thousands of degrees of freedom, then after refining the mesh and raising the polynomial degree by one globally, the number of degrees of freedom grows by an order of magnitude. We get a linear system with a matrix whose number of columns and rows reaches millions causing the computation even on a modern PC workstation unfeasible. This drawback was already addressed in the work of Pavel Kús and his collaborators (see [41, 42, 43]) where he shows that if an appropriate adaptive method is used, it is practically impossible for the low-order elements to reach the level of accuracy obtained by *hp*-FEM, regardless of the number of degrees of freedom.

CHAPTER 6

Concluding remarks

In this rather short chapter we present a short conclusion of our work. We also refer to several reports and articles that follow ideas presented in this thesis. Some problems that arose during the work on this thesis have already been addressed by other people and some are still waiting for an analysis. We point out the open problems that we find interesting.

In the first two chapters we presented the main concept of *hp*-FEM for elliptic partial differential equations in three spatial dimensions. We showed in detail some problematic parts of the discretization process and the algorithmic procedures to solve them.

In the next Chapter we tried to head towards the optimal shape functions for the H^1 conforming finite elements. Generalized eigenfunction-based shape functions proved to have excellent properties. This lead to further research in the field of $H(\mathrm{curl})$ conforming elements. A similar generalized eigenfunction-based basis was constructed for such elements and shown that they are Moreover, it was shown that the sets of generalized eigenfunction-based shape functions for different function spaces fit naturally into the De Rham diagram. The analysis was performed by Tomáš Vejchodský and Lenka Dubcová (see [26])

and presented at the FEMTEC 2006 conference.

The non-affine concept of FEM presented in Chapter 4 provides a novel approach to reducing the size of an algebraic system resulting from the *hp*-FEM discretization procedure. We showed that the method is mathematically equivalent to the process of static condensation of degrees of freedom. Both methods achieve the same results, although by different means. Both methods add some complexity to the already complicated process of 3D *hp*-FEM discretization. We cannot say strictly which one of them is better. For different problems one must always have a look at the character of each single problem and carefully decide which one of the methods is more beneficial to use. The non-affine concept is still in focus of the *hp*-FEM group at the University of Texas at El Paso. Some new results were published recently in [71].

The software presented in Chapter 5 proved to be a solid base for the research of numerical methods and algorithms connected to the 3D *hp*-FEM. It was taken over by Pavel Kús (Institute of Thermomechanics of the Academy of Sciences) and David Andrš (University of Texas at El Paso). Both of them contributed to the rapid development of new features and improved performance of the solver. Pavel implemented *hp*-adaptive algorithms and was successful in completing the support for computation on meshes with arbitrary level hanging nodes. This novel technique seems to be the most promising *hp*-adaptive procedure. Thanks to David the code is now much more versatile and it is much easier for new people to start developing new features. The work will continue in the same direction as in the case of 2D code. Support for time-dependent and multi-physics coupled problems will be added, as well as implementation of computation on *hp*-adapted moving meshes and multi-meshes.

Bibliography

[1] A. Ahagon, K. Fujiwara, T. Nakata, *Comparison of Various Kinds of Edge Elements for Electromagnetic Field Analysis*, IEEE Trans. Magn., 32, 898–901, 1996.

[2] M. Ainsworth, J. Coyle, *Hierarchic hp-edge Element Families for Maxwell's Equations on Hybrid Quadrilateral/Triangular Meshes*, Comput. Methods Appl. Mech. Engrg., 190, 6709–6733, 2001.

[3] P. L. Arlett, A. K. Bahrani, and O. C. Zienkiewicz, *Application of Finite Elements to the Solution of Helmholtz's Equation*, Proceedings of the Institution of Electrical Engineers, 115, 1762–1766, 1968.

[4] J. Argyris, P. C. Dunne, *Structural Analysis, Structural Principles and Data, Part 2*, Handbook of Aeronautics No. 1, The New Eva Publ. Co. Ltd., London, 1952.

[5] J. Argyris, S. Kelsey, *Energy Theorems and Structural Analysis*, Butterworths, London, 1960.

[6] I. Babuška, A. K. Aziz, *Survey Lectures on the Mathematical Foundation of the Finite Element Method*, The Mathematical Foundations of the Finite Element Method, Academic Press, New York, 1972.

[7] I. Babuška, M. Griebel, J. Pitkäranta, *The Problem of Selecting the Shape Functions For a p-type Finite Element*, Internat. J. Numer. Methods Engrg., 28, 1891–1908, 1989.

[8] I. Babuška, W. C. Rheinboldt, *Error Estimates for Adaptive Finite Element Computations*, SIAM J. Numer. Anal., 15, 736–754, 1978.

[9] I. Babuška, W. C. Rheinboldt, *A-posteriori Error Estimates for the Finite Element Method*, Internat. J. Numer. Methods Engrg., 12, 1597–1615, 1979.

[10] I. Babuška, T. Strouboulis, *Finite Element Method and Its Reliability*, Clarendon Press, Oxford, 2001.

[11] I. Babuška, B. Szabo, I. N. Katz, *The p-version of the Finite Element Method*, SIAM J. Numer. Anal., 18, 515–545, 1981.

[12] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, H. Zhang, *Portable, Extensible Toolkit for Scientific Computation*, web page: `http://www.mcs.anl.gov/petsc`, 2001.

[13] W. Bangerth, R. Hartmann and G. Kanschat, *deal.II – a General-Purpose Object-Oriented Finite Element Library*, ACM Transactions on Mathematical Software, 33, no. 4, article 24, 2007.

[14] L. Beilina, S. Korotov, M. Křížek, *Nonobtuse Tetrahedral Partitions That Refine Locally Towards Fichera-Like Corners*, Appl. Math., 50, 569–581, 2005.

[15] J. Bey, *Finite-Volumen- und Mehrgitterverfahren für Elliptische Randwertprobleme*, Doctoral thesis, Eberhard-Karls-University Tübingen, 1997.

[16] J. Červený, *Adaptive Finite Element Methods for Nonlinear Coupled Problems*, Master Thesis, University of Texas at El Paso, 2007.

[17] P. G. Ciarlet, *The Finite Element Method for Elliptic Problems*, North-Holland, Amsterdam, 1979.

[18] R. W. Clough, *The Finite Element Method in Plane Stress Analysis*, Proceedings, 2nd Conference on Electronic Computation, A.S.C.E. Structural Division, Pittsburgh, Pennsylvania, 1960.

[19] R. W. Clough, J. Penzien, *Dynamics of Structures*, McGraw-Hill, 1975.

[20] R. L. Courant, *Variational Methods for the Solution of Problems of Equilibrium and Vibrations*, Bull. Amer. Math. Soc., 49, 1–23, 1943.

[21] T. I. Davis, I. S. Duff, *An Unsymmetric-Pattern Multifrontal Method for Sparse LU Factorization*, SIAM J. Matrix Anal. Appl., 18, 140–158, 1997.

[22] L. Demkowicz, D. Pardo, W. Rachowicz, *3D hp-Adaptive Finite Element Package (3Dhp90)*, TICAM Report 02-24, University of Texas at Austin, 2002.

[23] L. Demkowicz, W. Rachowicz, P. Devloo, *A Fully Automatic hp-Adaptivity*, TICAM Report 01-28, University of Texas at Austin, 2001.

[24] L. Demkowicz, L. Vardapetyan, *Modelling of Electromagnetic Absorption/Scattering Problems Using hp-adaptive Finite Elements*, Comput. Methods Appl. Mech. Engrg., 152, 103–124, 1998.

[25] V. Dolejší, *On the Discontinuous Galerkin Method for the Numerical Solution of the Navier-Stokes Equations*, Internat. J. Numer. Methods Fluids, 45, 1083–1106, 2004.

[26] L. Dubcová, P. Šolín, *On hp-FEM Based on Generalized Eigenfunctions*, FEMTEC 2006 conference talk, University of Texas at El Paso, 2006.

[27] M. Feistauer, J. Felcman, I. Straškraba, *Mathematical and Computational Methods for Compressible Flow*, Clarendon Press, Oxford, 2003.

[28] G. Fichera, *Numerical and Quantitative Analysis*, Surveys and Reference Works in Mathematics, Vol. 3, Pitman, London–San Francisco–Melbourne, 1978.

[29] B. G. Galerkin, *On Electrical Circuits for the Approximate Solution of the Laplace Equation* (in Russian), Vestnik Inzhenerov, 1, 897–908, 1915.

[30] C. Geuzaine, J.-F. Remacle, *Gmsh: a Three-Dimensional Finite Element Mesh Generator With Built-in Pre- and Post-processing Facilities*, Web page: `http://www.geuz.org/gmsh/`.

[31] G. H. Golub, C. F. Van Loan: *Matrix Computations*, Johns Hopkins, Baltimore, 1996.

[32] P. Hood, C. Taylor, *A Numerical Solution of the Navier-Stokes equations using the finite element technique*, Comp. and Fluids, 1, 73–100, 1973.

[33] P. Houston, C. Schwab, E. Süli, *Discontinuous hp-Finite Element Methods For Advection-Diffusion Problems*, SIAM J. Numer. Anal., 39, 2133–2163, 2001.

[34] P. Houston, E. Süli, *hp-Adaptive Discontinuous Galerkin Finite Element Methods For First-Order Hyperbolic Problems*, SIAM J. Sci. Comput., 23, 1226–1252, 2001.

[35] G. E. Karniadakis, S. J. Sherwin, *Spectral/hp Element Methods for CFD*, Oxford University Press, 1999.

[36] A. Kirsch, P. Monk, *A Finite Element/Spectral Element Method for the Approximation of the Time-Harmonic Maxwell System in* $\mathbb{R}^3$, SIAM J. Appl. Math., 55, 1324–1344, 1995.

[37] S. Korotov, M. Křížek, *Acute Type Refinements of Tetrahedral Partitions of Polyhedral Domains*, SIAM J. Numer. Anal., 39, 724–733, 2001.

[38] S. Korotov, M. Křížek, *Global and Local Refinement Techniques Yielding Nonobtuse Tetrahedral Partitions*, Comput. Math. with Appl., 50, 1105–1113, 2005.

[39] S. Korotov, M. Křížek, P. Neittaanmäki, *Weakened Acute Type Condition for Tetrahedral Triangulations and the Discrete Maximum Principle*, Math. Comp., 70, 107–119, 2000.

[40] M. Křížek, T. Strouboulis, *How to Generate Local Refinements of Unstructured tetrahedral Meshes Satisfying a Regularity Ball Condition*, Numer. Methods Partial Differential Equations, 13, 201–214, 1997.

[41] P. Kús, *hp-FEM for 3D Electromagnetics and Related Coupled Problems*, Doctoral thesis, Charles University, Prague, expected 2009.

[42] P. Kús, P. Šolín, I. Doležel, *Solution of 3D singular electrostatics problems using adaptive hp-FEM*, Proceedings of ISEF Conference, Prague, Czech Republic, September 2007.

[43] P. Kús, P. Šolín, I. Doležel, *hp-FEM with Arbitrary Level Hanging Nodes in 3D*, Proceedings of ENUMATH Conference, Graz, Austria, September 2007.

[44] P. Lesaint, P. A. Raviart, *On a Finite Element Method for Solving the Neutron Transport Equation* Mathematical Aspects of Finite Elements in Partial Differential Equations, C. A. deBoor (Ed.), Academic Press, New York, 89–145, 1974.

[45] J. K. MacDonald, *Successive Approximations by the Rayleigh-Ritz Variation Method*, Phys. Rev., 43, 830, 1933.

[46] Y. Maday, A. T. Patera, *Spectral Element Methods for the Incompressible Navier-Stokes equations*, State-of-the-art Surveys in Computational Mechanics, ASME, New York, 1988.

[47] J. C. Nédélec, *Mixed Finite Elements in* $\mathbb{R}^3$, Numer. Math., 93, 315–341, 1980.

[48] A. T. Patera, *A Spectral Element Method for Fluid Dynamics; Laminar Flow in a Channel Expansion*, J. Comput. Phys., 1984.

[49] A. G. Peano, *Hierarchies of Conforming Finite Elements*, Ph.D. Dissertation, Washington University, 1975, Advisor: B. Szabó.

[50] A. G. Peano, *Hierarchies of Conforming Finite Elements for Plane Elasticity and Plate Bending*, Comput. Math. Appl., 2, 211–224, 1976.

[51] A. G. Peano, *Conforming Approximation for Kirchhoff Plates and Shells*, Internat. J. Numer. Methods Engrg., 14, 1273–1291, 1979.

[52] J. W. Rayleigh, *In Finding the Correction for the Open End of an Organ-Pipe*, Phil. Trans., 161, 77, 1870.

[53] W. H. Reed, T. R. Hill, *Triangular Mesh Methods for the Neutron Transport Equation*, Technical Report LA-UR-73-479, Los Alamos Scientific Laboratory, 1973.

[54] W. Ritz, *Über Eine Neue Methode zur Lösung Gewisser Variationsprobleme der Mathematischen Physik*, J. Reine Angew. Math., 135, 1–61, 1908.

[55] C. Schwab, *p- and hp-Finite Element Methods*, Clarendon Press, Oxford, 1998.

[56] H. Si, *TetGen: A Quality Tetrahedral Mesh Generator and Three-Dimensional Delaunay Triangulator*,
Web page: `http://tetgen.berlios.de/`.

[57] P. Silvester, *Finite-Element Solution of Homogeneous Waveguide Problems*, Alta Frequenza, 38, 313–317, 1969.

[58] P. Šolín, *Partial Differential Equations and the Finite Element Method*, Wiley & Sons, 2005.

[59] P. Šolín, *Scalar and Vector-Valued Shape Functions of Variable Order*, TICAM Report 02-36, University of Texas at Austin, 2002.

[60] P. Šolín, J. Červený, I. Doležel, *Arbitrary-Level Hanging Nodes and Automatic Adaptivity in the hp-FEM*, Math. Comp. Simul., 77, 117–132, 2008.

[61] P. Šolín, L. Dubcová, I. Doležel, *Adaptive hp-FEM With Arbitrary-Level Hanging Nodes for Time-Harmonic Maxwell's Equations*, Internat. J. Numer. Meth. Engrg., submitted.

[62] P. Šolín, K. Segeth, I. Doležel, *Higher-Order Finite Element Methods*, Chapman & Hall/CRC Press, 2003.

[63] P. Šolín, T. Vejchodský, *Higher-Order Finite Elements Based on Generalized Eigenfunctions of the Laplacian*, Int. J. Numer. Methods Engrg., 73, 1374–1394, 2007.

[64] P. Šolín, T. Vejchodský, M. Zítka, *Orthogonal hp-FEM for Elliptic Problems Based on a Non-Affine Concept*, In: Numerical Mathematics and Advanced Applications (Proceedings of ENUMATH 2005; A. Bermudez, D. Gomez, P. Quintela, P. Salgado Eds.), Springer, 683–690, 2006.

[65] P. Šolín, M. Zítka, I. Doležel, *On a hp-Finite Element Method for Singular Electro- and Magnetostatic Problems*, Proceedings of ISEF, Baiona, Spain, September 2005.

[66] P. Šolín, M. Zítka, K. Segeth, *Singularities in Electro- and Magnetostatics and Their Efficient Resolution by hp-FEM*, Proceedings of the Seminar of Applied Mathematics, 155–172, Czech Technical University, Prague, 2005.

[67] G. Strang, G. J. Fix, *An Analysis of the Finite Element Method*, Prentice-Hall, Englewood Cliffs, NJ, 1973.

[68] B. Szabó, I. Babuška, *Finite Element Analysis*, John Wiley & Sons, New York, 1991.

[69] B. Szabó, *PROBE, Theoretical Manual*, NOETIC Technologies Corp., St. Louis, 1985.

[70] M. J. Turner, R. W. Clough, H. C. Martin, and L. J. Topp, *Stiffness and Deflection Analysis of Complex Structures*, J. Aeronautical Sci., 23, No. 9, 805–823, 1956.

[71] T. Vejchodský, P. Šolín, *Static Condensation, Partial Orthogonalization of Basis Functions, and ILU Preconditioning in hp-FEM*, accepted by J. Comput. Appl. Math., 2007.

[72] T. Vejchodský, P. Šolín, M. Zítka, *On Some aspects of the hp-FEM for Time-Harmonic Maxwell's equations*, In: Numerical Mathematics and Advanced Applications (Proceedings of ENUMATH 2005; A. Bermudez, D. Gomez, P. Quintela, P. Salgado Eds.), Springer, 691–699, 2006.

[73] T. Vejchodský, P. Šolín, M. Zítka, *Modular hp-FEM System HERMES and Its Application to the Maxwell's Equations*, Math. Comp. Simul., 76, 223–228, 2007.

[74] S. Vyvialová, *Analysis and Optimization of a Class of Hierarchic Finite Element Methods*, Master thesis, University of Texas at El Paso, 2006.

[75] J. Wang, N. Ida, *Curvilinear and Higher-Order Edge Elements in Electromagnetic Field Computation*, IEEE Trans. Magn., 29, 1491–1494, 1993.

[76] H. Whitney, *Geometric Intergation Theory*, Princeton University Press, Princeton, 1957.

[77] S. Zhang, *Successive Subdivisions of Tetrahedra and Multigrid Methods on Tetrahedral Meshes*, Houston J. Math., 21, 541–556, 1995.

[78] O. C. Zienkiewicz, Y. K. Cheung, *The Finite Element Method in Structural and Continuum Mechanics*, McGraw-Hill, 1967.

[79] M. Zítka, *hp-FEM for Large Scale Singular 3D Problems*, Master Thesis, University of Texas at El Paso, 2006.

[80] M. Zítka, P. Šolín, T. Vejchodský, F. Ávila, *Imposing Orthogonality to Hierarchic Higher-Order Finite Elements*, Math. Comp. Simul., 76, 211–217, 2007.

[81] M. Zlámal, *On the Finite Element Method*, Numer. Math., 12, 394–409, 1968.