Simultanous h-p Adaptation in Multilevel Finite Elements

Gerhard Zumbusch

Berlin, April 1995

Contents

Introduction

An important tool in engineering is the finite element method. It's story of success can be traced back to the early times of computers, dividing a problem into smaller computable sub-problems, known as elements. Standard FEM codes employ a certain number of unknowns and elements to attain the desired precision of the solution. A further reduction of the global error requests an additional number of unknowns.

A first way to do this is the successive reduction of element sizes h which gives rise to the so-called h–version of finite elements. For the reduction of the energy norm of the global error by a factor of two, one has to reduce the element size h uniformly by about this factor two for the regular case and linear elements. This means eight times the number of unknowns in three dimensions, a refinement procedure which is clearly limited by computer resources.

Adaptive finite elements apply this refinement by h reduction only locally. They are able to inherit the convergence rates in terms of the number of unknowns from H^2–regular problems with uniform refinement to less regular problems by adapting grids [BKP79, Osw90]. However, they are not able to overcome the algebraic (one over polynomial) convergence of the h–version in general. Hence the reduction of the error by two still asymptotically means eight times more work (in 3D), assuming an optimal order algorithm. Nevertheless it has been claimed that this adaptive process in conjunction with an optimal order algorithm for the solution of the linear system of equations is in a certain sense optimal [BD81b].

A second way is to increase the local approximation properties by successively raising the polynomial degree of approximation thus exploiting the behavior of higher order derivatives of the solution. This leads to the so-called p–version. By this means the convergence rates can be improved to a superior pre-asymptotic (sub)-exponential decay (in the number of unknowns) under some assumptions. The assumptions concern higher order derivatives of the right hand side, e.g. analytic functions, and adaptation of the grids to singularities of the solution. In the (unavoidable) presence of singularities the p–version convergence asymptotically falls back to algebraic convergence like the h–version. Hence it becomes necessary to use grids locally adapted to singularities considering a maximal degree p.

The combination of both methods, the h–p version, supplies the pre-asymptotic exponentially convergent p–version continuously with properly adapted grids. Hence it achieves the superior exponential convergence asymptotically, too, instead of algebraic convergence of its ingredients the h–version and the p–version. Although the first theoretical results claiming these convergence rates are quite classic (e.g. [BD81a]), the number of codes using the h–p–version of finite elements is still rather limited. Reasons for that are the pure implementational complexity and the details, in conjunction with the rumor of engineers' low precision requirements. But the major reason is the lack of a robust (self-) adaptive control delivering the desired exponential convergence. For a brief history of finite elements cf. [Bab93] and [Ode91], and for a review of h–p methods we refer to [BS94].

In the this thesis we present some steps towards an efficient implementation of the theoretically known exponential convergence. As it turns out, an efficient implementation requires additional theoretical considerations, which play a major role there as well. This includes both the fully automatic h–p–version and as a subset the p–version on suitable grids. We present some details concerning our approach implementing an adaptive h–p–version based on an adaptive multilevel h–version code named KASKADE. This software package uses unstructured grids of triangles in two dimensions and tetrahedra in three dimensions.

We will discuss the local polynomial extensions of linear finite elements on simplices (chapter 1 on page 12). This leads to some considerations on properties of shape functions and the derivation of new families of shape functions. We will consider the fast multilevel iterative solution of the linear systems of equations and its impact on shape functions (chapter 1.4 on page 41). There will be considerations on a posteriori error estimation (chapter 2.1.1 on page 53), grid refinement procedures and some heuristics controlling the h–p–adaptive refinement process (chapter 2.1.2 on page 57). There will be a discussion on parallelization of the adaptive code (chapter 2.2 on page 60) and on post-processing a solution given on an h–p–adapted grid (chapter 2.4 on page 64). There will be extensive numerical investigations of convergence histories for different FEM versions and strategies, actual condition numbers (chapter 3.1 on page 74), quality of error estimators and performance of iterative solvers. The experiments cover the different types of singularities of the Laplacian in 3D (chapter 3.2 on page 88) and some examples from linear elasto mechanics (chapter 3.3 on page 100).

Linear Elliptic Problems

We consider a linear second order elliptic symmetrical boundary value problem of the type

$$\sum_{i,k=1}^{d} \partial_i\Big(a_{ik}(x)\partial_k u(x)\Big) + a_0(x)u(x) = f(x),\ \forall x \in \Omega$$

on the Lipschitz domain $\Omega \subset \mathbb{R}^d$ with suitable boundary conditions on $\partial\Omega$, a_0 being non-negative. We want to calculate the solution $u(x) \in \mathbb{R}$ on Ω with the finite element method. Using the bilinear form

$$a(u,v) = \int_\Omega \Big(\sum_{i,k=1}^{d} a_{ik}(x)\partial_i u(x)\partial_k v(x) \ + \ a_0(x)u(x)v(x)\Big)dx$$

and the L^2 scalar product

$$\langle v, f\rangle = \int_\Omega v(x)f(x)dx$$

in the variational formulation, one has to set up and solve the discrete linear system of equations

$$Au = b$$

defined by $A_{ik} = \big(a(\phi_i,\phi_k)\big)$ and $b_k = \langle\phi_k, f\rangle$. A suitable set of conforming shape functions $\phi_i \in \mathrm{H}^1(\Omega)$ has to be chosen. These shape functions are formed by local shape functions ψ_i on each finite element. If the solution is not accurate enough, one has to choose a better discretization ϕ_i and repeat the computation. In the case of vector valued solutions

$$u : \Omega \to \mathbb{R}^s$$

we generalize the whole setting by approximating each component $u_j, j \in 1,\ldots,s$ and using a variational form of an elliptic second order differential operator analogously to the bilinear form $a(.,.)$. The general properties of approximation and solution of the linear systems are preserved dealing with $s \cdot n$ number of unknowns. We shall consider this case in the examples of elasto mechanics in three dimensions with $s = d = 3$.

Depending on the kind of approximation, one has to distinguish some versions of finite element methods:

- The h–version is based on element subdivision, using identical (affine transformed) shape functions on all elements.

- The p–version keeps the elements, but creates enhanced shape functions on each element [BSK81].
- The h–p–version uses both procedures – subdivision and shape function enhancement [BS90].
- The r–version moves and distorts the given elements, preserving the number of unknowns, facing geometric difficulties in more than one space dimension.
- The more traditional interactive cycle of adaptation creates smaller elements which are not correlated with the older ones restarting the whole grid generation procedure with refined grid generation parameters.

Each version of finite elements methods can be applied adaptively (affecting only some parts of the domain) or globally (also called uniform refinement). The system of linear equations can be solved directly (e.g. by Cholesky decomposition) or iteratively. Iterative solvers need a solution to start with which can be supplied by the solution on the previous discretization level called nested iteration. Types of iterative solution techniques include one-grid methods such as conjugate gradients (cg) and Gauß-Seidel iterations and the family of multi-grid and multilevel solvers which exploit the history of all (coarser) levels. Here information on the solution process is transferred between different discretization levels.

A general finite element method consists of the following modules and operations which can merge into one another:

- *Constructing or managing a tessellation of the domain* Ω*.* We will only consider conforming tessellations of d–dimensional simplices. This means that two connected simplices have one and only one complete boundary face in common (a lower dimensional simplex). This prevents us from dealing with slave nodes which are local but not global degrees of freedom and have to be removed from the global system of equations by static condensation. Another point to mention is that we are working with only one element type, thus not mixing e.g. bricks, pyramids, wedges and tetrahedra.
- *Assembling the local element-wise matrices* A^{loc} *and right hand sides* b^{loc}. The numerical calculation of the integrals in $A^{\text{loc}}_{ik} = (a(\psi_i, \psi_k))$ and $b^{\text{loc}}_k = \langle \psi_k, f \rangle$ requires cubature formulas. The evaluation of the shape functions ψ_i is necessary only at the points of cubature and

can be done via tabulated function values. Using different sets of shape functions like in the p–version, one may have different cubature formulas. Hence for all formulas one needs such a table. The number is limited by the maximal polynomial degree which is usually lower than ten. In the case of constant or simple a_{ik}'s and a_0's the integrals may be calculated in advance using a transformation formula and, if necessary, a linear combination of such integrals. Thus it is not necessary to evaluate the shape functions in the assembling procedures if one can do some work in advance and does not use more sophisticated methods of integration. Hence we do not necessarily have to supply a numerical procedure for evaluation of shape functions which can be difficult or sometimes ill conditioned numerically in case of higher order polynomials.

- *Assembling the local terms into the global matrix A and right hand side b (coupling of the shape functions).* The simplest way of assembling is summing up the local matrices and vectors, having in mind the position of the local matrix elements in the global matrix. A more complicated situation arises in the assembly of slave nodes which leads to condensing them by solving local linear equations, a procedure which we excluded previously. General shape functions may cause the same trouble if they are not symmetrical in the sense of simple inter-element coupling. This means that for two connected elements each shape function ψ_i of one simplex E that is not vanishing on the common boundary $\overline{E} \cap \overline{E^*}$ there is a corresponding shape function ψ_k^* on the other simplex E^* which is identical on the border $\psi_k^*(x) = \psi_i(x)\ \forall x \in \overline{E} \cap \overline{E^*}$. In adaptive p–versions there is the additional problem of coupling different sets and degrees of polynomials. In the simplest case, one set $\Psi = \{\psi_i\}$ of non-vanishing polynomials is a subset of the other one $\Psi^* = \{\psi_i^*\}$ on the common boundary. There are two parts: coupling of the common functions $\Psi \cap \Psi^*$ as usual and coupling the hierarchic surplus $\Psi^* \setminus \Psi$ by restricting it to zero. In more generality one has to remove some degrees of freedom by static condensation.

- *Solving the linear system of equations.* Using fast iterative multi-grid or multilevel solvers, one has to transfer functions and updates of solutions between discretization levels. This requires a fast local transfer from one set of shape functions to another one (p–version) and from one element to the subdivided elements or vice versa (h–version). Hierarchic shape functions (p– or h–hierarchical) permit a relative convenient transfer by setting some coefficients to zero or by ignoring them. All solvers appreciate a well-conditioned and sparse global matrix A.

- *Estimating the error of the solution or determining whether the solution is acceptable.* Error estimation often leads to the comparison of different approximations which are sometimes only local. Hence it reduces work if one can expand or restrict (project) a calculated solution easily to another discretization. Following this line it is convenient to use p– or h–hierarchical shape functions or to allow a simple transfer in another way.

- *Constructing a new tessellation of the domain by considering the estimated local errors.* Apart from the algorithmical problem of constructing a new conforming tessellation which fulfills the necessary element angle conditions one may want to save the calculated solution. This can be achieved easily if the transfer operations between discretizations levels are simple.

- *Post-processing the solution.* After calculating the solution, we may want to determine some properties of the solution or simply visualize it. We may have to manipulate the solution using the shape functions.

H Adaptivity

The term of 'adaptivity' has been attractive for a long time in finite elements. It is used in several different connotations. We already vaguely have explained the meaning 'adaptive' in our context by the term of an iterative refinement procedure applied to the tessellation. The basic algorithm can be traced back as early as [BSW83]. The refinement procedure slightly differs from the one introduced by M. Rivara [Riv84a]. There is a vast number of publications and actual computer codes. To mention but a few in mathematical research:

- FEARS due to Mesztenyi and Szymczak [MMS82]
- PLTMG due to Bank [Ban82, Ban94]
- EXPDES due to Rivara [Riv84b]
- KASKADE due to Deuflhard, Leinen and Yserentant [DLY89, Lei90, Roi89a, Roi89b, ERFA91]
- MGGHAT due to W. Mitchell [Mit89]

- UG due to Bastian [BW93, Bas93]
- GOOFE due to Hiptmair [Hip95]
- and others, Craig, Zhu and Zienkiewicz [CZZ84], Johnson and Hansbo [JH92] or Nambiar, Valera and Lawrence [NVL93]

With generalization to three space dimensions by Bänsch [Bän91] and [BEK93a] publication continues. There are different directions of research concerning this kind of adaptivity, which are approximation with angle conditions [Zla68, Jam76, BA76, Kri92] and convergence [Dör94], error estimation and refinement control [BR78, BW85, BM87, DLY89, BDR92, BEK93b] and fast solution of the linear systems [OR70, BD81b, Yse86, BPX90, Bor91, Deu94, BD95].

Sometimes the refinement procedure is called 'self-adaptive' [McC89] to distinguish it from using the finite element method on grids which are adapted to the probable solution by other means than the code itself. This may be an intuition guided or interactive process delivering such adapted grids. The 'real' adaptivity should appear as a black box, doing the right decision itself. But of course in spite of adaptive control for real problems the initial coarse grid has to be chosen properly, cf. [BEK93b]. A general convergence prove for an a posteriori error estimator requires restrictions on input data, especially the initial grid [Dör94]. Another term used is 'feedback' [BM87, SB91] denoting the difference between a control dependent on some computational results and a provable optimal control, which [SB91] call adaptive. Hence any claims of optimality are today restricted to structured grids, to mention the key word of asymptotically exact error estimators. This means we may have to call all involved procedures feedback instead of adaptive. Hence we keep the term adaptive.

There is another kind of adaptive algorithms based on rectangular block structured grids with a refinement by overlay patches of rectangles: Brandt's MLAT [Bra76] and McCormick's FAC and AFAC. The procedure exploits the local regularity of grids for computer performance and ease of implementation, but relies heavily on the geometry of rectangles. Hence we do not consider this farther. There are a lot of finite element codes using rectangles and bricks than just simplices. In general there are some difficulties with geometry. For an arbitrary polyhedral shaped domain one has to cope with (only a few) heavily distorted elements. So simplices seem to be superior from a geometric point of view. Certainly there are degradations of convergence with linear functions on simplices compared with bilinear or tri-linear functions on rectangles and bricks, but we do propose higher order finite elements in this text, too.

Convergence

For standard convergence theory on finite elements we also refer to Ciarlet [Cia80, Cia91]. Convergence of finite elements depends on the regularity of the problem, the size of the finite elements h and the order of the functions on the elements (and the error norm involved). It can be majorized by the properties of interpolation with finite elements. Hence we end up with a term

$$||\epsilon||_a = \mathcal{O}(h^\alpha), \quad h = c \cdot n^{-d}$$

for the uniform case. α denotes on the regularity of the solution and the local order of approximation, h denotes the diameter of the finite elements, ϵ is the approximation error (in energy norm) and n is the dimension of the discretized vector space or simply the number of unknowns (degrees of freedom). h–adaptivity is able to improve the constant c involved drastically for an effective minimal h and additionally does raise the effective regularity α to the H^2–regular value (e.g. $\alpha = 2$) . A problem with low regularity will be solved by an h–adaptive process with lower n and lower computational expense, although overall convergence remains the algebraic (double precision $= 2^d$ times the work). The other way round this means, that for very regular problems adaptivity does not pay off. To improve convergence, one has to use better local approximations, to be able to further raise α with regularity assumptions, that do not hold in general, of course.

We briefly want to mention the convergence results known for h–p and pre-asymptotic p version of finite elements. We now get new quality of convergence of type

$$||\epsilon||_a = \mathcal{O}(e^{-c\sqrt[\beta]{n}})$$

with $\beta = 2$ in one dimension, $\beta = 3$ in two dimensions [BG88, BD81a] and $\beta = 5$ in three dimensions [Guo93], see also [BS94]. This holds under an assumption on the grid and refinement control being optimal. The (sub)-exponential convergence is clearly visible in numerical experiments. The resulting method is at least as fast as usual finite elements being an optimal superposition of existing methods compared to. Now adaptive control does not only improve constants, but it facilitates this improved exponential convergence rates.

Implementations so far

There is a huge amount of finite element codes, both in research and commercially. Only some use refined approximations and only one part of them adaptivity. h–refinement is common in multi-grid and multilevel context, hence there are many codes implementing the h–version. At present there are some codes arising, which use the p method of finite elements like

- PROBE from Noetic
- PEGASYS from ESRD, both based on work of Babuška and B. Szabó [BSK81, Sza85, Sza86] meanwhile distributed as
- MSC/ PROBE from MacNeal Schwendler including an announced update of
- MSC/ NASTRAN from MacNeal Schwendler
- the p-adaptive STRIPE from Aeronautical Research Institute of Sweden
- MECHANICA from Rasna
- POLY FEM from IBM (Almaden Research Center) [MTC92, CMTT93]
- and others like Zienkiewicz, Zhu, Craig and Ainsworth [ZZCA89] and the p-adaptive code of Beck [Bec93]

There has been some research towards h–p codes like Gui and Babuška [GB86a, GB86b, GB86c] and [BR87] and applications like [ZW92, SW92, Hoh94], After all there is only one commercial h–p finite element code known

- Computational Mechanics' PHLEX which meanwhile is
- PDA Engineering's P3D/ CFD based on the work of Oden, Demkowicz et al. [DORH89, ODRW89, ROD89]

One can make a distinction between uniform and adaptive distributions of the order p on a grid and procedures refining h and p at once or one after the other, refining h in a feedback / (self)-adaptive manner or by a priori or semi-automatic criteria. We will see that an adaptive p distribution is under certain conditions easy implementable, sometimes much easier than local h refinement. Hence the question whether to adapt p or not depends on the reduction of number of unknowns by adaptation compared to some computational overhead triggered by the adaptation. Refinement control is not as critical and can rely on standard refinement criteria. The decision of simultaneous h and p refinement is a question of implementational complexity and even more a question of a robust refinement control. Hence there is a lack of any available commercial or research finite element code in more than one space dimension doing this.

We want to characterize the properties of our research finite element code as follows: It has to implement an h–p version. It is build upon the existing

h–adaptive multilevel finite element code KASKADE [DLY89, Lei90, Roi89a, Roi89b, ERFA91] and its three dimensional version [BEK93a]. This implies the usage of unstructured triangle and tetrahedra grids without any slave nodes. The simplex elements are chosen for a maximum of flexibility in input geometry and a uniform treatment of elements without transition elements or elements with different convergence properties.

In the spirit of KASKADE the code should be a (self-) adaptive one in the broadest sense. We observe that any robust automatic control will supersede a semi-automatic or manual control on the long run. Of course an expert is able to outperform such an automatic control, but once established it delivers a new freedom to care about things and maybe details which are really worth it, relying on a sub-optimality of the automatic control. Hence any step towards such an automatic control is welcome.

To put it to an end: We want to implement a maximum of adaptivity meaning an adaptive distribution of the polynomial orders p, an adaptive distribution of the step-sizes h (diameter of the elements) and an automatic refinement procedure able to apply both refinements at once. We will see that the 'green closure' [BSW83] avoiding slave nodes implemented in KASKADE not only pays off in h–adaptation but in p–adaptation in conjunction with non-uniform grids, too. But we will also see that usual ideas from shape functions on bricks enabling easy p adaptation do not always carry over to tetrahedra. Extending the whole set of constituents of an adaptive finite element code to an h–p adaptive one, such as error estimation and refinement and fast linear algebra, we are able to present some numerical results.

Acknowledgement

I would like to express my deep gratitude to P. Deuflhard whose support and encouragement made this work possible. He had initiated the KASKADE project at the Konrad–Zuse–Zentrum Berlin (ZIB) some years ago, which had been a subject of my diploma thesis in Munich and he proposed the h–p topic, when I joined the KASKADE team at the ZIB. Without his insistence on the importance of the symmetric polynomial topic I would just have skipped it, probably undiscovered further on. He also suggested the applications area of elasto mechanics leading to some interesting and challenging industrial cooperation.

I also want to thank I. Babuška for discussion on topics of h–p refinement control, B. A. Szabó for a brief history and some remarks on p–version and M. Griebel for discussion on p–version topics and 'multi-p'. I want to thank both J. T. Oden and L. Demkowicz for providing some literature on h–p–methods and P. Carnevali (meanwhile Rasna Corp.) for his paper on IBM's p–version FEM. Furthermore I thank E. Rank and V. Shaidurov for some discussions on h–p procedures, and J. Xu for discussing preconditioners for high polynomial degrees p during their visits at the ZIB.

Many thanks to the staff at the ZIB who provided a very comfortable and productive atmosphere, in particular our 'theorist' F. A. Bornemann for his lectures especially the one on FEM and some bibliographic comments, K. Gatermann for discussion on issues of symmetry and polynomials, A. Hohmann for his thesis' acknowledgement and some topics of object orientedness and on h–p strategies and R. Beck never failing to point out 'the whole absurdity' of h–p methods, linear elasto mechanics and so on. I also want to thank the other members of the KASKADE team for several contributions, just to mention R. Roitzsch, B. Erdmann, J. Lang and R. Kornhuber. Last but not least I want to thank our computing center and service staff, especially the computer graphics team with H.-C. Hege and D. Stalling and the REDUCE team of H. Melenk and W. Neun.

We have done the symbolic computations including the management of polynomials and permutation groups with list-oriented features of REDUCE [Hea93] (about 700 lines of code, 14kB), which triggered an optimization procedure written in C++ (also about 700 lines of code, 14kB), using a matrix library written by A. Hohmann. The final results were computed and drawn by MATLAB [Mat92]. Some illustrations were drawn using UNIDRAW and IRIS SHOWCASE. The Nurbs pictures were drawn with the IRIS INVENTOR package.

A first finite element code was based on the C version of KASKADE by R. Roitzsch and on 3D extensions by B. Erdmann, about 30,000 lines of code (800kB). The final finite element code was written in C++, about 36,000 lines of code, (1.0MB) based on on an early C++ version of KASKADE written by R. Beck. Finite element pre- and post-processing was done with SDRC's I-DEAS.

1 Theoretical Derivation of Special Shape Functions

In this chapter we will discuss the topic of shape functions on simplex shaped finite elements. Due to linearity of FEM, the solution does not depend on shape functions but on function spaces spanned by them. Hence we theoretically construct a families of shape functions fitting to some implementational constraints. The final form of the shape functions will be derived by use for a special iterative solver. Of course any other standard set of polynomial shape functions can be used instead, causing some trouble in implementation.

A reader not interested in properties of simplexes but in quadrilateral or hexahedral elements is referred to references like [DORH89, BCMP91] and may proceed with the next chapter 2 on page 53. There implementational details such as error estimation, refinement control, parallelization or graphical representation are discussed, which are widely independent from a special choice element shapes. The first part of the numerical experiments chapter 3 on page 74 deals with simplex elements and special shape functions, whereas the second and third parts contain comparisons of different FEM versions (and the same initial grid) for Poisson equation and for linear elasto mechanics, certainly valid for other element shapes, too.

The main purpose of this chapter is the derivation of a special family of shape functions for simplex elements in d–dimensions. Although the shape functions in 2D on the triangle are new and combine some nice properties, they really pay off in higher dimensions. These shape functions in 3D on the tetrahedron were used in the numerical experiments part's FEM code.

The derivation of shape functions relies on two properties: The elements in an arbitrary simplex mesh cannot be oriented, which will be illustrated later. Focussing on fast linear algebra and iterative solvers, we want to avoid static condensation of the global stiffness matrix, which is absent in the case of usual linear and (p–hierarchic) quadratic shape functions and which contradicts the lightweight iterative solver strategy for example with on the fly (implicit) matrix multiplication but appears due to non-orientedness (compatible shape functions). The second property stems form variable order (p) approximation leading to p–hierarchy.

However if one decides to implement static matrix condensation (some costly linear combinations of shape functions), like the description for hexahedral elements by [DORH89], our choice of shape functions is not necessary any longer, but there appear alternatives. Although the orientation demand for hexahedral grids may be relaxed for simple structured meshes, even for hexahedral grids it cannot be avoided in general.

1.1 Standard Shape Functions

We consider the shape functions spanning the discrete vector spaces used in the Galerkin approach of the finite element method. These shape functions are formed by composition of local shape functions ϕ_i on each finite element.

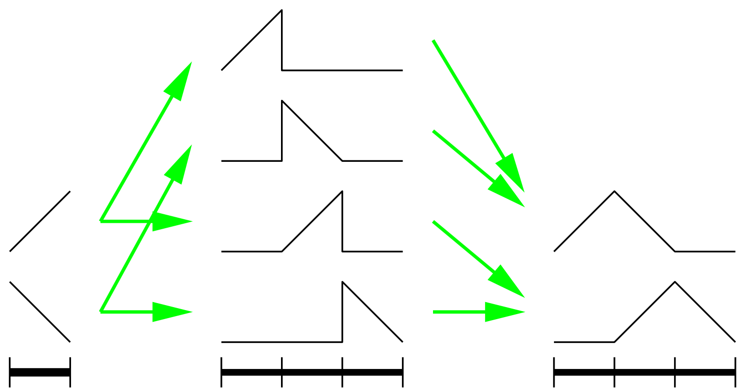

Figure 1: Composition of shape functions in 1D, a local shape functions, b shape functions in the global space, c coupled to global continuous functions for Galerkin procedure

In the case of non-scalar partial differential equations one could think of dedicated systems of coupled shape functions like some elements in structure and fluid mechanics, whose convergence has to be proven by a kind of patch test [IR72]. This is especially valid for non conforming finite elements. There are special shape functions for higher order differential operators [CT65], non symmetric shape functions for non symmetric differential operators [BH82, Web88] and variants of the finite element method using functions with higher continuity like splines [CL90], some kinds of eigenfunctions, trigonometric functions or rational functions [Coo68, Wac75]. We do not consider these variants here, but we just wanted to show alternative ways of shape functions.

We introduce the barycentric coordinates $(b_0, b_1, \ldots, b_d)$ in a d dimensional space with respect to a d-simplex, sometimes called area or volume coordinates or homogeneous coordinates. They may be characterized by an affine transform with coordinates $(0, \ldots, 0, 1, 0, \ldots, 0)$ coresspond to a vertex of the simplex. Hence coordinate $\frac{1}{d}(1, \ldots, 1)$ is the barycenter of the simplex. Using multi index notation we define the vector space $\mathcal{P}_p^d$ of polynomials of degree p in d variables by the linear span of

$$\mathcal{P}_p^d = \langle \bigcup_{|\alpha| \leq p} b^\alpha \rangle, \ \alpha \in \mathbb{N}_0^{d+1}$$

Sometimes shape functions are written in terms of $\{x, y, 1 - x - y\}$ on a reference triangle. This is equivalent to a function in $\{b_1, \ b_2, \ b_0\}$.

1.1.1 Lagrange Polynomials

Figure 2: Lagrange polynomials in 1D, degree 5

The Lagrange polynomials (c.f. figures 2 and 3 on the facing page) are interpolation polynomials on a set of equidistant points called 'control points' x_i (figure 4 on page 16). The polynomials are defined by the orthogonality

Figure 3: Lagrange polynomials on the triangle, degree 5:

a	b
c	d
e	f

a all together, b point functions, c,d edge functions, e,f inner functions

relation

$$f_i(x_j) = \delta_{i,j}$$

The polynomials of degree p are defined on a d–simplex by the equidistant distribution of $\binom{p+d}{d}$ control points on the simplex, spanning the space $\mathcal{P}_p^d$. The set of Lagrange polynomials implements the interpolation property of the linear shape functions $p = 1$ often used for conforming finite elements.

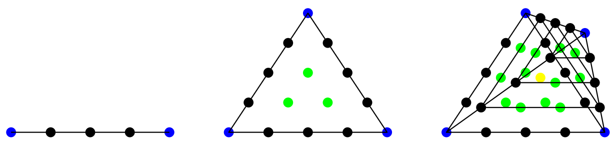

Figure 4: Equidistant control points for the interval, the triangle and the tetrahedron

Global continuity can be achieved for a uniform polynomial degree p identifying shape functions of all elements sharing one geometric control point (coupling of element matrices). Implementation of Dirichlet boundary conditions is easy, too. Shape functions are symmetric/ affine invariant due to symmetry/ affine invariance of the control points.

Some less convenient properties are: Coupling of different degree elements or elements in a non-conforming mesh is local quite expensive. Besides the values given at the control points for a function evaluation inside an element all shape function of the element are involved. The polynomials are highly oscillatory without connection to the differential operator.

There are some slight modifications, moving the position of the control points and using points of the numerical integration formula. There is another proposal for un-symmetric modifications of edge shape functions in the inner of a triangle [DKO92].

1.1.2 Bernstein Polynomials

The Bernstein polynomials are defined by [Far90]:

$$f_\alpha = \binom{|\alpha|}{\alpha} b^\alpha, \;\; \alpha \in \mathbb{N}_0^{d+1}$$

The Bernstein polynomials of degree $|\alpha| = p$ (c.f. figures 5 on the facing page and 6 on page 18) generate the vector space $\mathcal{P}_p^d$. They use the equidistant control points α/p. They implement the interpolation property of the linear shape functions $p = 1$. Global continuity can be achieved for a uniform polynomial degree p identifying shape functions of all elements sharing one geometric control point (coupling of element matrices). This set of polynomials is symmetric.

Some less convenient properties are: Coupling of different degree elements or elements in a non-conforming mesh is quite expensive. Implementation

Figure 5: Bernstein polynomials in 1D, degree 5

of non-constant Dirichlet boundary condition is expensive. For any function evaluation inside an element all shape function of the element are involved.

1.1.3 Legendre Polynomials

We define the orthogonal Legendre polynomials (c.f. figure 7 on page 19 on the interval $[-1,1]$ by

$$f_j(x) = \frac{1}{2^j j!}\frac{d^j}{dx^j}((x^2-1)^j)$$

The Legendre polynomials (c.f. figures 7 on page 19 and 8 on page 20) are orthogonal with respect to the scalar product $\langle .,. \rangle$ on $[-1,1]$. They are hierarchical in their polynomial degree p and symmetric to the origin. The symmetry behavior is alternately odd and even. To exploit the orthogonality in the case of a 1-D problem for the Laplace operator (i.e. $a_0 \equiv 0$ and $a_{11} \equiv 1$) one has to use integrated polynomials as shape functions: $\int_{-1}^{x} f_j(t)dt$

Figure 6: Bernstein polynomials on the triangle, degree 5:

a	b
c	d
e	f

a all together, b point functions, c,d edge functions, e,f inner functions

[SB91]. Now the bilinear form $a(u, v) = \langle u', v' \rangle$ operates on the same terms as the scalar product in the previous case. The integrated polynomials are orthogonal with respect to the new bilinear form.

Following B. Szabó and Babuška [SB91] we construct shape functions on

Figure 7: The integrated Legendre polynomials

the simplex using the Legendre polynomials:

$$f_{i,j}^{p} := \sqrt{8(2p-1)}\frac{b_i\, b_j}{1-(b_i-b_j)^2}\int_{-1}^{b_i-b_j} f_{p-1}(x)dx$$

edge function, point i towards point j.

$$f_{i,j,k}^{p_1,p_2} := b_i b_j b_k f_{p_1}(b_j - b_i) f_{p_2}(2b_k - 1)$$

triangle function, points i, j and k.

$$f_j^p := (\prod_{i\geq 2} b_{j_i}) f_{p_1}(b_{j_1} - b_{j_0}) \prod_{i\geq 2} f_{p_i}(2b_{j_i} - 1)$$

generalized inner function, points $j_0, j_1, j_2, \ldots$.

The Legendre polynomials on the simplex (c.f. figure 8 on the next page) even for Laplace operator are not orthogonal any longer. They are not fully symmetric any longer (or skew symmetric), but they prefer one coordinate.

Figure 8: Using Legendre polynomials on the triangle, pictures and degrees:

a	b	
c	d	e
f	g	h
i	j	k

a=1 (point function), b=2 (edge function), c=3 (edge function), d=3 (inner function), e=4 (edge function), f,g=4 (inner function), h=5 (edge function), i,j,k=5 (inner function)

They contain the linear shape functions and extend them p–hierarchically. p–hierarchy maintains easy implementation of varying degrees p and coupling of different order elements.

Global continuity in 2D induces coupling of skew-symmetric edge functions, subtraction of local stiffness matrices. Coupling in 3D or higher d even for conforming grids may be locally expensive, depending on orientation of the elements. Function evaluation in the interior of an element as usual involves all shape functions an is costly.

The preference of one specific direction can be exploited for directed refinement using an anisotropy of polynomial degrees which may be useful in [KR90, BS94].

1.1.4 Monomials

Figure 9: Monomials on an 1D edge, up to degree 5

We present a early version of p–hierarchic finite element shape functions by A. Peano [Pea76] using monomials. Originally the monomials were defined

Figure 10: Monomials on the triangle, pictures and degrees:

a	b	c
d	e	f
g	h	i

a=0 (constant function), b=1 (point function, rotate only once), c=2 (edge function), d=3 (edge function), e=3 (inner function), f=4 (edge function), g=4 (inner function, rotate only once), h=5 (edge function), i=5 (inner function)

on the triangle. We present a recursive version (c.f. figure 10):

$$
\begin{aligned}
f_2^0(b_0,b_1,b_2) &:= 1 \\
f_{2,i}^1(b_0,b_1,b_2) &:= b_i, \qquad 0 \le i < 2, \quad \text{point functions} \\
f_{2,i}^p(b_0,b_1,b_2) &:= b_i^{p-1}\, b_{i+1(\mathrm{mod} 3)}, \qquad 0 \le i \le 2, \quad \text{edge functions} \\
f_{2,i+3}^p(b_0,b_1,b_2) &:= f_{2,i}^{p-3}(b_0,b_1,b_2)\, b_0 b_1 b_2, \qquad 0 \le i, \quad \text{inner functions}
\end{aligned}
$$

We generalize the function on the d–simplex by (c.f. figures 9 on the page before and 10) and use a recursion over d and p

$$
\begin{aligned}
M_d^0 &:= \Big\{ f((b_j)_{j=0}^d) = 1 \Big\} \\
M_d^1 &:= \Big\{ f((b_j)_{j=0}^d) = b_k \mid k < d \Big\} \\
M_1^p &:= \Big\{ f(b_0, b_1) = b_0^p \Big\} \\
M_d^p &:= \Big\{ f = g((b_{\pi_0,\pi_1,\dots,\pi_k})) \prod_{j=0}^{k} b_{\pi_j} \mid \\
&\qquad g \in M_k^{p-k-1},\ 1 \le k \le d,\ \text{combination } (\pi_j)_{j=0}^k \subseteq (j)_{j=0}^d \Big\},
\end{aligned}
$$

The polynomials are p–hierarchic, the first used explicitly in finite elements, but they are not fully symmetric, some transformations of the polynomials are missing for symmetry ('rotate only once' in figure 10 on the preceding page). They contain the standard linear shape functions. Monomials were chosen to achieve cheap function evaluations. With proper code optimization this advantage may not pay off (see section 2.3 on page 63). Full definition is expanded from original lists for a comparison with other shape functions.

1.1.5 Hermite Polynomials

We define some special sort of Hermite polynomials in 1D by

$$
f^p(x) := \begin{cases} \frac{1}{2}(1 \mp x) & p < 2 \\ \frac{1}{p!}(x^p - 1) & p \text{ even} \\ \frac{1}{p!}x(x^{p-1} - 1) & p \text{ odd} \end{cases}
$$

on the interval $x \in [-1, 1]$

They contain the linear shape functions on $[1, 1]$. Additionally they use derivatives $\frac{d^p}{dx^p}|_{x=0}$ at the origin (c.f. figure 11 on the next page). The polynomials are p–hierarchic. Following Zienkiewicz and R.Taylor [ZT89] we construct higher dimensional shape functions on the triangle by

$$
f_{i,j}^p := \begin{cases} \frac{1}{p!}\Big((b_i - b_j)^p - (b_i + b_j)^p\Big) & p \text{ even} \\ \frac{1}{p!}(b_i - b_j)\Big((b_i - b_j)^{p-1} - (b_i + b_j)^{p-1}\Big) & p \text{ odd} \end{cases}
$$

as edge functions, i to j, $p \ge 2$.

The edge functions equal the 1D Hermite polynomials on the edge (i, j). The first inner function, $p = 3$ is given by

$$
f_{k_0,k_1,k_2}^3 := b_{k_0} b_{k_1} b_{k_2}
$$

Figure 11: Hermite polynomials in 1D, up to degree 5

Fourth order inner functions are given by

$$\{b_0^2 b_1 b_2,\ b_0 b_1^2 b_2,\ b_0 b_1 b_2^2\}$$

although higher order shape functions or any systematic approach is missing. Continuing the construction of inner functions by monomials, we end up with symmetric, non–p–hierarchic shape functions.

Modifying the cases $p \geq 4$ in the way like in section 1.1.4 on page 21 of [Pea76] (which was cited in [ZT89]), we have p–hierarchic shape functions, which are not fully symmetric. They are depicted in figure 12 on the next page. Scaling of the shape function could be improved, depending on the purpose of use.

1.1.6 Other Polynomials

We already have seen orthogonal polynomials in 1D, giving rise to optimal condition numbers for linear algebra (section 1.1.3 on page 17). One could

Figure 12: Using Hermite polynomials on the triangle, pictures and degrees:

a	b	c
d	e	f
g	h	i

a=1 (point function), b=2 (edge function), c=3 (edge function), d=3 (inner function), e=4 (edge function), f (inner function, rotate only once), g=5 (edge function), h=5 (inner function, rotate only once), i=5 (inner function, do not rotate)

think of continuing the property of orthogonality in higher dimensions.

There are such p–hierarchic polynomials on simplexes [Jac36, Mys81, GM78, Dub91, SK95] used for construction of integration formulas analogously the 1D Gauß quadrature connected with the Legendre polynomials.

There are some drawbacks: 'Orthogonality' has to be interpreted as orthogonality of spaces of polynomials, not polynomials itself.

$$\langle f_i^p, f_j^q \rangle = 0 \text{ if } p \neq q$$

Now condition numbers are not optimal for $d > 1$, but are supposed to be low. Optimal condition number one can of course be achieved be a orthogonalization procedure, while loosing any properties useful for implementation of finite elements.

Using standard orthogonal polynomials, which maintain some properties of symmetry, is locally expensive. The coupling of local stiffness matrices involves almost all functions of an element, because the polynomials do not vanish on the boundary of the simplex. There has to be done some computational work in finite elements, $d > 1$, either in linear algebra or in handling optimal conditioned shape functions.

Turning to some other polynomials, we will present some graphs in chapter 1.4.3 on page 44 depicting a version of symmetric hierarchic polynomials [Zum93] for comparison. A detailed description will be given in the following sections.

1.1.7 Overview

Here we show some properties of different families of shape functions:

polynomials	reference	*simple* coupling	symm. on the faces	symm. in the element	hierarch. in p
monomials	$\prod_{i=1}^{d} x_i^{\alpha_i}$	–	–	–	$\times$
orthogonal	[AF26, GM78]	–	–	–	$\times$
mod. Legendre	[SB91]	$\times$	2-D only	–	$\times$
mod. monomials	[Pea76]	$\times$	2-D only	–	$\times$
p–hierarch.	mod. [ZT89]	$\times$	2-D only*	–*	$\times$*
original*	[ZT89]	$\times$	$\times$	$\times$	–
Lagrange	[MP72, Nic72]	$\times$	$\times$	$\times$	–
Bernstein	[Far90]	$\times$	$\times$	$\times$	–
symm. hierarch.	(this thesis)	$\times$	$\times$	$\times$	$\times$

The reader should not be surprised that we can separate two classes of shape functions: The p–hierarchical and the symmetrical ones. The only exception are the new symmetric hierarchical polynomials. The asterisk * denotes our modification of the polynomials originally proposed by [ZT89]. The original shape functions on the triangle are not p–hierarchical for the step from degree 3 to 4. We modified the basis to be hierarchical, loosing symmetry.

On the interval, say $[-1,1]$, the classic orthogonal Legendre polynomials are leading to a kind of optimal set of shape functions for the p– and h–p–version of finite elements for the Laplace equation. A pure higher-order h–version may also use interpolation Lagrange polynomials. For the p– and h–p–version on tensor product structures like quadrilaterals and hexahedrals one could generalize these integrated Legendre polynomials [SB91], loosing some of their good transformation properties and investing more degrees of freedom than necessary in the approximation sense. But for the simplex one has to give up some other nice characteristics of the Legendre polynomials and a more complicated approach [Pea76, SB91, ZT89] has to be used. For a pure h–version one can use Lagrange interpolation [MP72, Nic72].

1.2 Properties of Special Shape Functions

In the following we analyze the approximation of functions by an adaptive (multilevel) finite element code. This leads to useful properties of shape-functions on the simplex. Only some properties are compatible with each other. Each version of finite elements differs in exploiting these properties for an efficient implementation. This is the next part of the section. In a second part we construct vector spaces containing sets of polynomials well-suited for the p– and the h–p–version of finite elements. In a third part we finally construct shape functions within these spaces which are optimal in the sense of an optimal condition number of the preconditioned linear system.

1.2.1 Hierarchical Polynomials

We introduce the concept of p–hierarchy or p–extension. If we have a basis $\mathcal{B}_p$ of shape functions spanning the function space $\mathcal{V}_p$ and we want to reach the space $\mathcal{V}_{p+1}$, we simply can add some shape functions to get the enhanced basis $\mathcal{B}_{p+1} = \mathcal{B}_p \cup \mathcal{B}_{p+1}^{\text{ext}}$.

Definition 1

$$\mathcal{V}_{p+1} = \langle \mathcal{B}_{p+1} \rangle = \langle \mathcal{B}_p \rangle \oplus \langle \mathcal{B}_{p+1} \setminus \mathcal{B}_p \rangle$$

P–hierarchy is an integral part of an approximation with varying order p in space. If we choose the order p_1 on one finite-element and a different p_2 on a neighboring element, one can achieve global continuity by linear constraints like [DORH89] or by handling the p–hierarchic excess in a special way (setting it to zero).

An example for p–hierarchic polynomials are the previously mentioned Legendre polynomials. The Legendre polynomials $f_p(x)$ are orthogonal with respect to the scalar product $\langle .,. \rangle$ on $[-1,1]$. They are hierarchical in their polynomial degree p and symmetrical to the origin. The symmetry behavior is alternately odd and even. To exploit the orthogonality in the case of a one dimensional problem for the Laplace operator one has to use integrated polynomials as shape functions: $\int f_j(t)dt$ [SB91]. Then the bilinear form $a(u,v) = \langle \frac{d}{dx}u, \frac{d}{dx}v \rangle$ operates on the same terms as the scalar product does in the previous case. The integrated polynomials are orthogonal with respect to the bilinear form.

Definition 2 *Here we associate the term 'orthogonal' polynomials with a sequence of nested sets of polynomials $\mathcal{B}_1 \subset \mathcal{B}_2 \subset \ldots$ for a specific bilinear form. The polynomials have to be linearly independent. A polynomial $f \in \mathcal{B}_i$ is orthogonal with respect to this bilinear form on the vector space generated by the basis $\mathcal{B}_{i-1}$ (no condition for $\mathcal{B}_1$). The vector spaces generated by $\mathcal{B}_i$ usually are the vector spaces of polynomials $\mathcal{P}^d_{i+1}$. The polynomials in $\mathcal{B}_i \setminus \mathcal{B}_{i-1}$ need not be orthogonal onto themselves.*

Orthogonal polynomials are hierarchical in p by definition. Orthogonal polynomials do not necessarily lead to local matrices with condition numbers equal to one, $\kappa_j(A^{\text{loc}}) = 1$, like the Legendre polynomials do. However, a basis with this desirable property can be constructed. Although the optimal shape functions in 1D are in fact orthogonal polynomials, we will see, that there is no way to use orthogonal polynomials in higher dimensions efficiently. While maintaining p–hierarchy, we do not end up with orthogonal functions.

1.2.2 Coupling

We call the assembly of local finite-element-matrices into a global one coupling, sometimes called 'global assembly'. One line of interpretation is the representation of global FEM ansatz functions, each connected with an degree of freedom, by linear combinations of local shape functions on an element. Coupling means this linear combination, which ideally is a one-to-one relation (local permutation matrix, called 'simple'). This would be the case for so called compatible shape functions. Many FEM codes use this simple form of global matrix assembly, assuming that the local shape functions are suited for it.

Another bottom-up interpretation is that we have to guarantee we are dealing with globally continuous shape functions $\{\psi_i\}$, which are formed by properly connected local shape functions $\{\phi_i\}$. We introduce two new terms: *simple* and *minimal* coupling.

Definition 3 *We call the coupling of the shape functions of two connected elements* minimal, *if the number of shape functions involved is minimal.*

This number $n(E, E^*)$ equals twice the dimension of the polynomial vector space on the intersection $\overline{E} \cap \overline{E^*}$ of both elements E, E^*. Coupling coefficients zero corresponding to vanishing shape functions on the intersection do not contribute to n.

We can express the coupling by an under-determined system of linear equations. Taking a coupling matrix C and the sets of shape functions $\{\phi_i\}$ and $\{\phi_i^*\}$, we can write the constraints as

$$C \cdot (\phi_1,\ \phi_2,\ \ldots,\ \phi_1^*,\ \phi_2^*,\ \ldots)^T = 0 \quad \text{on} \quad \overline{E} \cap \overline{E^*}.$$

By eliminating columns containing only zeros, eliminating linearly dependent rows and permuting we arrive at a reduced matrix $C \in \mathbb{R}^{n \times 2n}$ of rank n.

We introduce a stronger term of coupling by a special kind of minimal coupling which we call *simple*. The under-determined system of linear equations with (reduced) matrix C should facilitate the conversion between the coefficients of the functions $\{\phi_i\}$ and $\{\phi_i^*\}$. We reduce the matrix C to a smaller matrix $\tilde{C}$ by leaving out columns which are linearly dependent or zero.

Definition 4 *We define a coupling of shape functions $\{\phi_i\}$ and $\{\phi_i^*\}$* simple, *if there exists a reduced and permuted matrix $\tilde{C}$ of maximal rank which has block-diagonal form with 1×2 non-zero blocks.*

The reduced matrix looks like this:

$$\tilde{C} = \begin{pmatrix} \tilde{C}_1 & & & \\ & \tilde{C}_2 & & \\ & & \ddots & \\ & & & \tilde{C}_n \end{pmatrix} \quad \text{with } \tilde{C}_i \in \mathbb{R}^{1 \times 2}.$$

Example 1 *We look at the* simple *coupling of two elements E and E^* with 2×2 local matrices A and B and shape functions $\{\phi_1, \phi_2\}$ and $\{\phi_1^*, \phi_2^*\}$. Function ϕ_2 equals function ϕ_1^* on $\overline{E} \cap \overline{E^*}$. No other shape functions of*

E and E^* are correlated. This leads to a matrix $C = \begin{pmatrix} 0 & 1 & -1 & 0 \end{pmatrix}$, a reduced matrix $\tilde{C} = \tilde{C}_1 = (1\ -1)$ and to

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{pmatrix}$$

coupled with

$$\begin{pmatrix} b_{11} & b_{12} \\ b_{12} & b_{22} \end{pmatrix}$$

adds up to

$$\begin{pmatrix} a_{11} & a_{12} & 0 \\ a_{12} & a_{22} + b_{11} & b_{12} \\ 0 & b_{12} & b_{22} \end{pmatrix}.$$

Simple coupling may also appear as blocks of $\tilde{C}_i = (1\ 1)$, in general as $\tilde{C}_i = (1\ \lambda)$, $\lambda \neq 0$ or as small blocks simply invertible.

The Lagrange interpolation polynomials for equidistributed interpolation points (as in chapter 1.1.1 on page 14) of degree $|\alpha| = p$ generate the vector space $\mathcal{P}_p^d$. This set of polynomials is symmetrical. Any affine transform of the simplex onto itself will cause a permutation of the shape functions, but no linear combinations are necessary. The polynomials facilitate a *minimal* and *simple* coupling of blocks $\tilde{C}_i = (1\ -1)$ by identifying the proper functions being identical on the common boundary. The same holds for the Bernstein polynomials. On the other hand, it is harder to apply Dirichlet-boundary conditions. One has to interpolate the prescribed values using the polynomials non-vanishing on the boundary.

Although the efficient implementation of boundary condition is only of minor interest, because $\partial\Omega$ is of lower complexity compared to Ω, we want to remark, that in general, only Lagrange polynomials permit a simple implementation of (non-homogeneous) Dirichlet boundary conditions. The situation changes in the case of other boundary conditions like Neumann conditions. Leaving Lagrange polynomials, one may invest some more computational effort in calculating interpolation conditions on the boundary for gaining some nicer properties in the inner domain Ω.

Remark 1 *We conclude that there are shape functions with* minimal *and* simple *coupling. Some are symmetrical, too.*

1.2.3 Symmetry

Finite Element methods often use a simple set of shape functions defined on a reference element. In the case of simplexes each shape function is transferred to a real element using a linear transformation . There are different possibilities to realize this transformation. The transformation is unique only modulo permutation of the corner points. Hence one has to be able to couple any face of one element with any face of another one, where faces can be points, edges, triangles and so on.

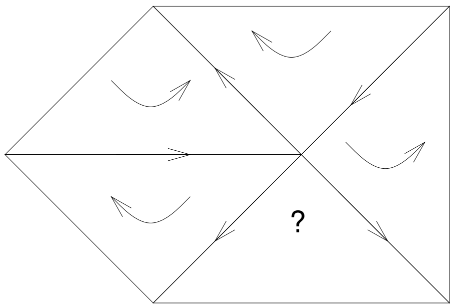

Figure 13: Problems in orientating a tessellation in 2D

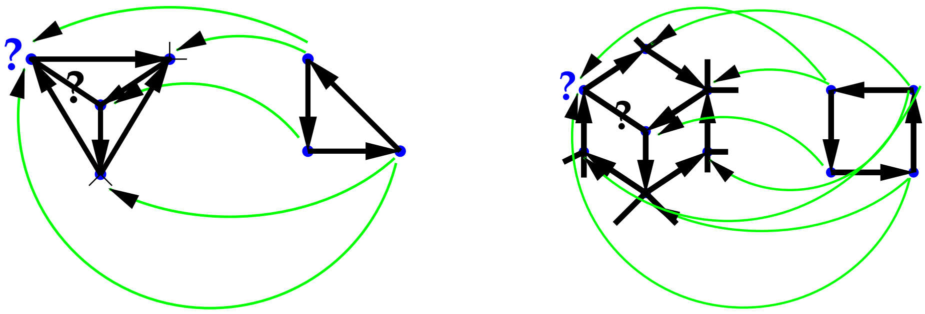

Figure 14: Non-oriented tessellations and mapping to standard elements in 2D

One can think of a completely oriented tessellation where the coupling is restricted to only some distinguished combinations of faces. But in general there is no such orientation (figure 1.2.3).

Hence there is no way out of having a deeper look into symmetry and coupling properties.

Definition 5 *We denote the group of permutations of d elements with* S_d *and the subset of the alternating group with* S_d^+.

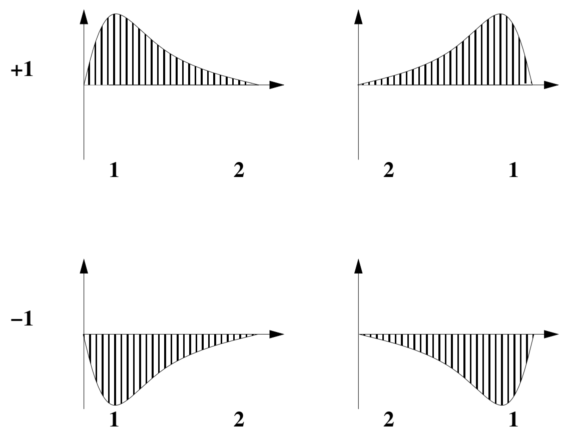

Figure 15: $\pm\mathrm{S}_2$ reflections of a 1–D function

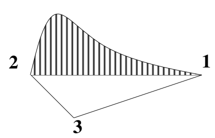

Figure 16: A 2–D function f on the triangle

Definition 6 *We define the action of a group* $\mathrm{S} \subset \mathrm{S}_{d+1}$ *on a set of polynomials* $\mathcal{B}$ *in d variables by the set of polynomials resulting from permuting the input variables (by the permutations of the group) in barycentric representation. This covers the definition of the action on a single polynomial and on a whole vector space of polynomials.*

$$\mathrm{S}\mathcal{B} = \bigcup_{f \in \mathcal{B}, s \in \mathrm{S}} f(s^{-1}(b_0, b_1, \ldots, b_d))$$

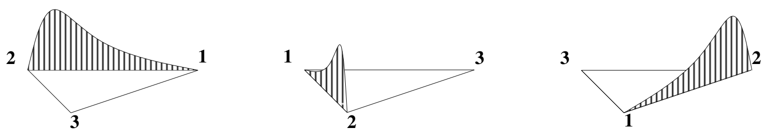

Figure 17: S_3^+ reflections of a 2–D function f

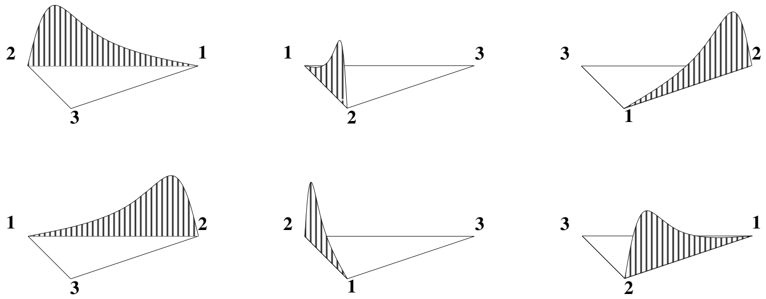

Figure 18: S_3 reflections of a 2–D function f

Definition 7 *We call a polynomial f, a set of polynomials $\mathcal{B}$ and a vector space $\mathcal{V}$ of polynomials* S*–symmetrical, if it is invariant with respect to the action of* S

$$f = \mathrm{S}f,\ \ \mathcal{B} = \mathrm{S}\mathcal{B}\ \text{ and } \mathcal{V} = \mathrm{S}\mathcal{V}.$$

It immediately follows that

- a set of S–symmetrical polynomials is an S–symmetrical set of polynomials and
- a vector space generated by an S–symmetrical set of polynomials is S–symmetrical itself.

Additionally we introduce point-symmetry which is *not* covered by the previous definitions.

Definition 8 *We define a set of polynomials $\mathcal{B}$ to be $\mathrm{S}_{d+1}^{\pm}$–symmetrical in d variables by*

$$\forall s \in \mathrm{S}_{d+1} \text{ and } \forall f \in \mathcal{B} \text{ holds } sf \in \mathcal{B} \text{ or } -(sf) \in \mathcal{B}.$$

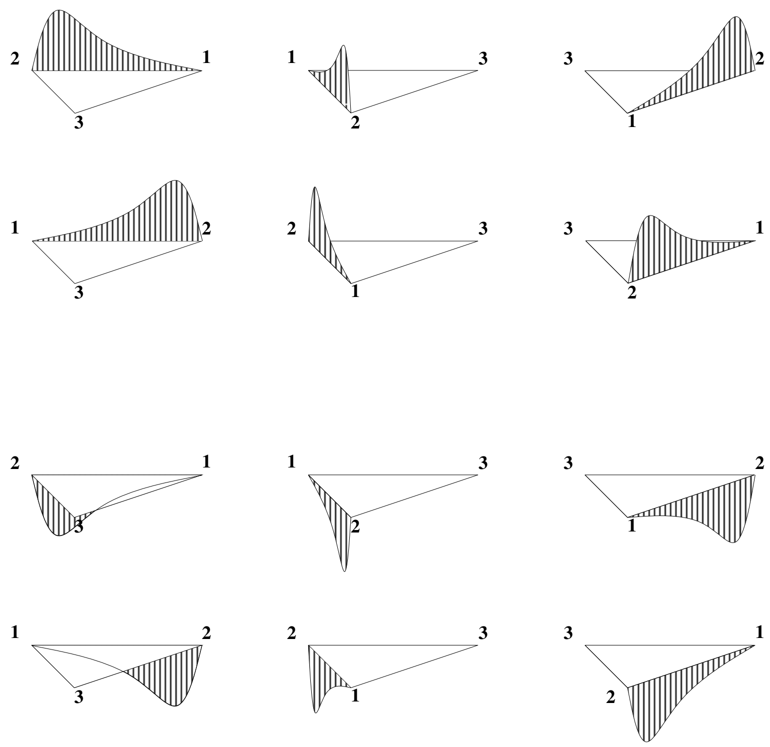

Figure 19: $\pm\mathrm{S}_3^{\pm}$ reflections of a 2–D function f, correlated with $\mathrm{S}_3^{\pm}$

Remark 2 *Defining symmetry by* $\forall s \in \mathrm{S}_{d+1}$ *and* $\forall f \in \mathcal{B}\ \exists \lambda \in \mathbb{R} \setminus \{0\}$ *with* $\lambda(sf) \in \mathcal{B}$ *leads to* $\lambda = \pm 1$*, too.*

Lemma 1 *An* $\mathrm{S}^{\pm}$*–symmetrical set* $\mathcal{B}$ *of polynomials is* S^{+}*–symmetrical.*

Remark 3 *We conclude that there are* $\mathrm{S}_d^{\pm}$*– and* S_d*–symmetrical shape functions.*

1.2.4 Symmetry and Coupling

Finite Element methods often use a simple set of shape functions defined on a reference element. In the case of simplexes each shape function is transferred to a real element using an affine transformation. There are different possibilities to realize this transformation. The transformation is unique only modulo permutation of the corner points. Hence one has to be able to couple any face of one element with any face of another one, where faces can be points, edges, triangles and so on.

One can think of a completely oriented tessellation where the coupling is restricted to only some distinguished combinations of faces. But in general there is no such orientation.

Hence there is no way out of having a deeper look into symmetry and coupling properties.

Theorem 1 *A set of shape functions for a general conforming tessellation of d–simplexes will permit a* simple *coupling with blocks* $\tilde{C}_i = (1 \;\; -1)$ *if and only if the shape functions permit* minimal *coupling and are* S_{j+1}*–symmetrical on each j–dimensional face of a simplex.*

We can relax this condition a little by requiring only $\psi_i = \pm\psi_k$ on the common boundary which leads to addition and subtraction of local matrices.

Corollary 1 *A set of shape functions for a general conforming tessellation of d–simplexes will permit a* simple *coupling with blocks* $\tilde{C}_i = (1 \;\; \pm 1)$ *if the shape functions permit* minimal *coupling and are* $\mathrm{S}^{\pm}_{j+1}$*–symmetrical on each j–dimensional face of a simplex.*

Remark 4 S_{d+1}*–symmetry is correlated with* simple *coupling of* $(1 \;\; -1)$ *and* $\mathrm{S}^{\pm}_{d+1}$*–symmetry is correlated with* simple *coupling of* $(1 \;\; \pm 1)$.

1.2.5 Symmetry and Hierarchy

We now want to derive the correlation of symmetry and p–hierarchy. The Legendre polynomials for example are p–hierarchic and $\mathrm{S}^{\pm}_2$–symmetrical, which simply means point and axial symmetry in one dimension. For d dimensions we get the following main result:

Theorem 2 *There is no p–hierarchical* S^{+}_{d+1}*–symmetrical polynomial basis on the d–simplex for* $d > 1$.

Proof. We look at the p–hierarchical step from polynomial degree $j(d+1)$ to $j(d+1)+1$ with $j \in \mathbb{N}_0$. Note that the dimension of symmetrizations of the set $\{b_0 - b_1, b_1 - b_2, \ldots, b_{d-1} - b_d\}(b_0 \cdot b_1 \cdots b_d)^j$ is at least $d+1$, but the vector space is of dimension d. $\square$

Corollary 2 *There is no p–hierarchical* $\mathrm{S}^{\pm}_{d+1}$*–symmetrical polynomial basis on the d–simplex for* $d > 1$.

Theorem 3 *There is no p–hierarchical* S_{d+1}*–symmetrical polynomial basis on the d–simplex for* $d \geq 1$*.*

Corollary 3 *There are no* S^+_{d+1}*–symmetrical orthogonal polynomials on the d–simplex for* $d > 1$*.*

Remark 5 *Symmetry and simple coupling on the one hand and p–hierarchy for* $\mathcal{P}^d_p$ *on the other hand exclude each other.*

1.3 Construction of New Polynomial Spaces

We want to construct a family of p–hierarchical shape functions for the d–simplex. It has to facilitate a simple coupling which implies symmetry (chapter 1.2.4). It should be suitable for a p– and h–p–version of finite elements with variable order p which means p–hierarchy in some sense. Both properties are not possible at the same time (chapter 1.2.5).

We have to cope with the limitations of theorem (2). We shall enlarge the polynomial vector spaces $\mathcal{P}^d_p$ slightly and construct new S_{d+1}–symmetrical vector spaces which avoid the irreducible subspaces of the proof.

1.3.1 Symmetry S_d

We recursively construct a basis for the new vector space $\mathcal{P}^{d,\mathrm{sym}}_p$ by the span of the vector space $\mathcal{P}^{d,\mathrm{sym}}_{p-1}$ one degree lower and additional functions. These functions are internal functions formed by the product of the "bubble" function $\prod_{j=0}^{d} b_j$ with functions of degree $(p-d-1)$ and boundary functions defined on the faces. The boundary functions are S_{d+1}–permutations (symmetrizations) of such (lower-) i–dimensional functions $(\mathcal{B}^i_{p-i-1} \cdot \prod_{j=0}^{i} b_j)$, $i < d$. The only difference to the standard polynomial spaces $\mathcal{P}^d_p$ is the beginning of the recursion. We start with $\{b_0\}$ for $\mathcal{B}^0_0$ which actually has degree 1. If we want to get the standard polynomial spaces $\mathcal{P}^d_p$, we should have taken $\{1\}$. We have enlarged the vector space. This enlargement spreads to the higher dimensions and the higher degrees.

Definition 9 *We recursively construct a basis for the new vector space in barycentric coordinates based on lower dimensions and lower degrees using the group of permutations* S_d*:*

- $\mathcal{B}^0_0 = \{b_0\}$
- $\mathcal{B}^0_p = \emptyset,\ p > 0$

- $\mathcal{B}_p^d = \bigcup_{i=0}^d \ \mathrm{S}_{d+1}(\mathcal{B}_{p-i-1}^i \cdot b_0 \cdot b_1 \cdots b_i),\ d \geq 1.$

Sometimes shape functions are written in a form like $\{x, y, 1-x-y\}$ on a reference triangle. This is equivalent to $\{b_1,\ b_2,\ b_0\}$.

Remark 6 *In the previous definition we can substitute the action of* S_{d+1} *by the combinations without repetition of* $i{+}1$ *elements of the set* $\{b_0, b_1, \ldots, b_d\}$.

Definition 10 *We now define the new polynomial vector spaces as the span of the basis functions in d dimensions:* $\mathcal{P}_p^{d,\mathrm{sym}} = \langle \bigcup_{i=0}^p \mathcal{B}_i^d \rangle$

Remark 7 *The vector spaces* $\mathcal{P}_p^{d,\mathrm{sym}}$ *are* S_{d+1}*–symmetrical. Their bases* $\mathcal{B}_p^{d,\mathrm{sym}}$ *are p–hierarchical, facilitate* minimal *and* simple *coupling with blocks* $(1\ -1)$ *and are enlarged* $\mathcal{P}_p^d \subseteq \mathcal{P}_p^{d,\mathrm{sym}} \subseteq \mathcal{P}_{p+1}^d$.

These polynomial spaces are well-suited for the coupling $(1\ -1)$, but they have got a high dimension (= too many shape functions). If we relax the coupling to $(1\ \pm 1)$, we can reduce this high dimension, but we have to consider the group $\mathrm{S}_{d+1}^{\pm}$ (chapter 1.2.4).

1.3.2 Symmetry $\mathrm{S}_d^{\pm}$

Definition 11 *We recursively construct a basis for the new* $\mathrm{S}_{d+1}^{\pm}$*–symmetrical vector space, taking the same* 0*–dimensional space and the action of the alternating group* S_d^{+} *otherwise, modifying the one dimensional basis:*

- $\mathcal{B}_0^{0,\pm} = \{b_0\},\ \ \mathcal{B}_p^{0,\pm} = \emptyset,\ p > 0$
- $\mathcal{B}_0^{1,\pm} = \{b_0,\ b_1\},\ \ \mathcal{B}_1^{1,\pm} = \emptyset,\ \ \mathcal{B}_p^{1,\pm} = \{(b_1 - b_0)^p\},\ p > 1$
- $\mathcal{B}_p^{d,\pm} = \bigcup_{i=0}^d \ \mathrm{S}_{d+1}^{+}(\mathcal{B}_{p-i-1}^{i,\pm} \cdot b_0 \cdot b_1 \cdots b_i),\ d > 1.$

Remark 8 *In the previous definition we can substitute the action of* S_{d+1}^{+} *by the even combinations without repetition of* $i+1$ *elements of the set* $\{b_0, b_1, \ldots, b_d\}$. *Watch out for a systematic interpretation of "even"!*

Definition 12 *We now define the new polynomial vector spaces as the span of the basis functions in d dimensions:* $\mathcal{P}_p^{d,\pm} = \langle \bigcup_{i=0}^p \mathcal{B}_i^{d,\pm} \rangle$

Example 2 *In zero dimension we get the following sequence of polynomials, which are only useful for the construction of higher dimensional ones:*
$\mathcal{P}_0^{0,\pm} = \mathcal{P}_1^{0,\pm} = \mathcal{P}_2^{0,\pm} = \ldots = \langle \{b_0\} \rangle$

Starting with the one dimensional $\mathrm{S}_2^{\pm}$*–symmetrical polynomials we get the following sequence:*

$$
\begin{array}{lllll}
\mathcal{P}_0^{1,\pm} & = & \mathcal{P}_1^{1,\pm} & = & \langle\{b_0,\ b_1\}\rangle \\
\mathcal{P}_2^{1,\pm} & = & \langle\mathcal{P}_1^{1,\pm} & \cup & \{(b_1-b_0)^2\}\rangle \\
\mathcal{P}_3^{1,\pm} & = & \langle\mathcal{P}_2^{1,\pm} & \cup & \{(b_1-b_0)^3\}\rangle \\
\vdots & & & &
\end{array}
$$

The spaces $\mathcal{P}_p^{1,\pm}$ *are equal to the former spaces* $\mathcal{P}_p^1$ *for* $p > 0$*. Thus they are smaller than the spaces* $\mathcal{P}_p^{1,\mathrm{sym}}$*. The one dimensional basis is not enlarged any more. Inserting this into the definition for two dimensions we get a sequence of* $\mathrm{S}_3^{\pm}$*–symmetrical polynomials:*

$$
\begin{array}{lllll}
\mathcal{P}_0^{2,\pm} & = & \mathcal{P}_1^{2,\pm} & = & \langle\{b_0,\ b_1,\ b_2\}\rangle \\
\mathcal{P}_2^{2,\pm} & = & \langle\mathcal{P}_1^{2,\pm} & \cup & \{(b_1-b_0)^2,\ (b_2-b_1)^2, \\
 & & & & (b_0-b_2)^2\}\rangle \\
\mathcal{P}_3^{2,\pm} & = & \langle\mathcal{P}_2^{2,\pm} & \cup & \{(b_1-b_0)^3,\ (b_2-b_1)^3, \\
 & & & & (b_0-b_2)^3\}\ \cup\ \{b_0(b_0b_1b_2), \\
 & & & & b_1(b_0b_1b_2),\ b_2(b_0b_1b_2)\}\rangle \\
\mathcal{P}_4^{2,\pm} & = & \langle\mathcal{P}_3^{2,\pm} & \cup & \{(b_1-b_0)^4,\ (b_2-b_1)^4, \\
 & & & & (b_0-b_2)^4\}\rangle \\
\vdots & & & &
\end{array}
$$

On the triangle the polynomial sets for a degree p which is not divisible by 3 *are identical to* $\mathcal{P}_p^2$*, all other vector spaces are generated by* $\mathcal{P}_p^2$ *and* 2 *additional polynomials.*

Remark 9 *The usual linear Lagrange polynomials are contained in both* $\mathcal{B}_1^d$ *and* $\mathcal{B}_1^{d,\pm}$*. The associated hierarchical quadratic polynomials are contained in* $\mathcal{B}_2^{d,\pm}$*, too.*

Remark 10 *The linear Lagrange polynomials can be interpreted as symmetrization of the canonical basis of* $\mathcal{P}_1^d$*:* $\{1\} \cup \{b_1, b_2, \ldots, b_d\}$*.*

Remark 11 *The bases* $\mathcal{B}_p^{d,\pm}$ *are* $\mathrm{S}_{d+1}^{\pm}$*–symmetrical, p–hierarchical and facilitate* minimal *and* simple *coupling with blocks* $(1\ \pm 1)$*. The spanned vector spaces* $\mathcal{P}_p^{d,\pm}$ *are only slightly enlarged (*$\mathcal{P}_p^d \subseteq \mathcal{P}_p^{d,\pm} \subseteq \mathcal{P}_p^{d,\mathrm{sym}} \subseteq \mathcal{P}_{p+1}^d$*) and have got an even lower dimension than* $\mathcal{P}_p^{d,\mathrm{sym}}$*.*

1.3.3 The Enlargement

We saw that the enlarged polynomial spaces fulfill

$$\mathcal{P}_p^d \subseteq \mathcal{P}_p^{d,\pm} \subseteq \mathcal{P}_{p+1}^d$$

which limits the enlargement of $\mathcal{P}_p^{d,\pm}$ (note 11). Actually we can show that with $p \to \infty$ this enlargement vanishes in the following sense:

$$\frac{dim\mathcal{P}_p^{d,\pm} - dim\mathcal{P}_p^d}{dim\mathcal{P}_{p+1}^d - dim\mathcal{P}_p^d} \leq O(p^{-1}) \quad \text{for fixed } d$$

We prove this also giving the dependence on d by

Lemma 2

$$dim\mathcal{P}_p^{d,\pm} - dim\mathcal{P}_p^d \leq d\binom{p+d-2}{p}$$

Proof. $dim\mathcal{P}_p^d$ may also be written by the recursive equation of $dim\mathcal{P}_p^{d,\pm}$ obtained form definition of $\mathcal{P}_p^{d,\pm}$. The difference obeys the same equation. It can be majorized by a recursion of $d \cdot dim\mathcal{P}_p^{d-2}$. □

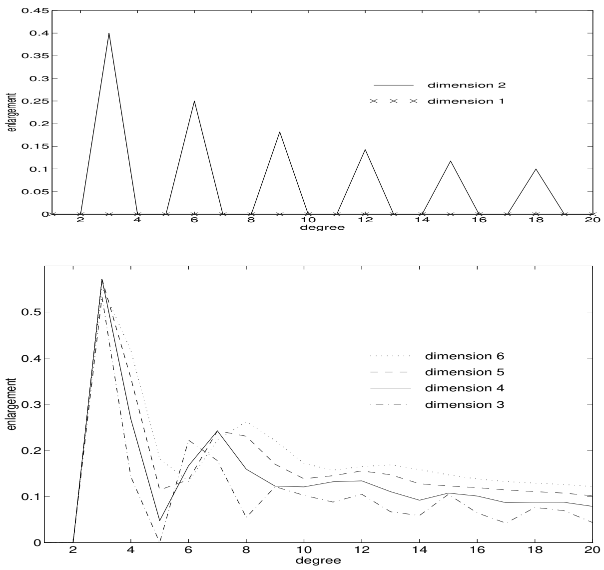

Figure 20: Enlargement of $\mathcal{P}_p^{d,\pm}$: values of $(dim\mathcal{P}_p^{d,\pm} - dim\mathcal{P}_p^d)/(dim\mathcal{P}_{p+1}^d - dim\mathcal{P}_p^d)$

With $dim\mathcal{P}_p^d = \binom{p+d}{p}$ the proof of formula (1.3.3) is completed. Figure (20) shows the actual values of the quotient. $\mathcal{P}_p^{1,\pm}$ is not enlarged and $\mathcal{P}_p^{2,\pm}$ is enlarged by 2 polynomials only for every third p. It indicates in conjunction

Table I: Dimensions of $\mathcal{P}_p^d$ and $\mathcal{P}_p^{d,\pm}$

d	dim $\mathcal{P}_5^d$	dim $\mathcal{P}_5^{d,\pm}$	dim $\mathcal{P}_{10}^d$	dim $\mathcal{P}_{10}^{d,\pm}$
1	6	6	11	11
2	21	21	66	66
3	56	56	286	294
4	126	130	1001	1045
5	252	276	3003	3192
6	462	546	8008	8757

Table II: Dimensions of $\mathcal{P}_p^3$ in $\mathbb{R}^3$, full and serendipity spaces, S-symmetric, S$^\pm$-symmetric and S$^\pm$-symmetric on the boundary only, extensions/ reductions (serendipity)

p	serendip S	S$^\pm$	S$^\pm$ on boundary	original $\mathcal{P}_p^3$	extension S$^\pm$ on boundary	S$^\pm$	S
1	4	4	4	4	4	4	4
2	4	10	10	10	10	10	16
3	16	16	16	20	28	28	28
4	28	34	35	35	35	38	44
5	44	56	56	56	56	56	68
6	68	80	80	84	92	92	104
7	104	116	120	120	120	128	140
8	140	164	165	165	165	168	192

with the actual numbers in table I on the facing page that the values decrease asymptotically with p^{-1}.

In table II on the preceding page we have added the dimensions for full symmetric and (generalized) serendipity polynomial spaces obtained by the recursion. Serendipity in this context are incomplete polynomial spaces due to symmetrization analog the previous extension of spaces. The spaces of polynomials with symmetry on the boundary of the element only do not differ much from symmetry for all polynomials including the inner functions. The $S^{\pm}$-symmetric serendipity polynomials are an attractive lower dimensional counterpart of the $S^{\pm}$-symmetric full polynomials $\mathcal{P}_p^{d,\pm}$ and the S-symmetric serendipity polynomials are connected with full S-symmetric polynomials $\mathcal{P}_p^d$. We also remark the drop-out of full symmetric serendipity polynomials at degree $p = 2$ and via recursion inherited minor drop-outs. In the further we will use the polynomials $\mathcal{P}_p^{d,\pm}$ only.

1.4 Shape Functions for Iterative Solvers

1.4.1 Condition of the Stiffness Matrix

It is well known that the condition number of the global stiffness matrix for a fixed set of shape functions depends on the extension of the elements h. For elements of uniform size h, we get a sharp estimate

$$\kappa(A) \le C(a, p, \gamma) h^{-2}.$$

The constant C depends on the interior angles γ of the elements, on the differential operator a, on the set of shape functions and the associated polynomial degree p. In the case of non-uniform h and simplexes we get a lower estimate J. Xu [Xu89, Xu92] with n denoting the number of simplexes

$$\kappa(A) \le C(a, p, \gamma) \cdot \begin{cases} n(1 + \log \frac{\max h}{\min h}) & \text{for } d = 2 \\ n^{2/d} & \text{for } d \ge 3 \end{cases}$$

Let us consider the p–dependence of the constants C. In the one-dimensional case there are shape functions with C independent from p, namely the integrated Legendre polynomials (chapter 3.1.1 on page 74). Next we can conclude from sharp estimates in [BCMP91], that there are shape functions in two dimensions with global condition number

$$\kappa(A) \le C(a, \gamma) h^{-2} (1 + \log^2 p)$$

for uniform h. The required functions are split into point-, edge- and internal shape functions in the usual way and the edge-functions have to be discrete harmonic with respect to the operator a. For dimensions higher than two ($d > 2$) an analogous construction delivers rapidly growing condition numbers in p [BCMP91].

If we now split the local condition numbers $\kappa_0(A_{\text{loc}})$ and $\kappa_1(A_{\text{loc}})$ into maximal and minimal eigenvalue $\lambda_{\max}(A_{\text{loc}})$ and $\lambda_{\min}(A_{\text{loc}})$ and look for the maximal $\lambda_{\max}$ and the minimal $\lambda_{\min}$ on all elements of a tessellation, we get a connection of local and global condition numbers. We conjecture an estimate for uniform h of the kind

$$\kappa(A) \leq C(\text{coupling})h^{-2}\frac{\max\lambda_{\max}(A_{\text{loc}})}{\min\lambda_{\min}(A_{\text{loc}})}$$

The constant C may also be written as a function of the minimal interior angle $\gamma_{\min}$ and the family of shape functions. The conjecture is supported by numerical experiments and the estimates in section 3.1 on page 74.

1.4.2 Preconditioning

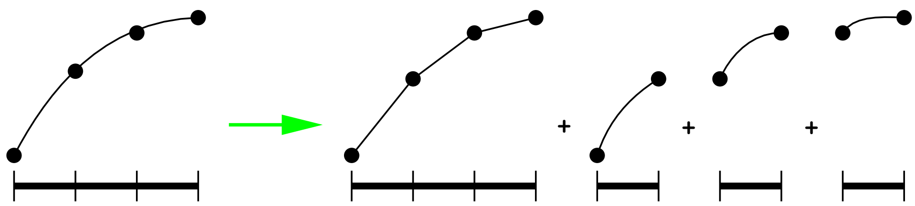

Figure 21: Domain decomposition preconditioner in 1D, global space of piecewise linear functions and local spaces of higher order polynomials

In the context of the iterative solution of the linear systems another tool comes into play. It is the convergence rate of the iteration. It can be estimated for some classic iterations like conjugate gradients and Richardson-iteration in terms of the condition number. But for sake of efficiency linear systems are often preconditioned, so we have to consider the condition number of the preconditioned matrix instead. There are two similar approaches for preconditioning linear systems of p–version [BCMP91, Man90a, Man90b] and [Pav92, Pav94b, Pav94a], both leading to estimates independent from h and with a rather mild increase in $\log^2 p$. We also have to mention an earlier experimental approach by [BGP89] in connection with the idea of 'multi-p' (Griebel) as an analogon to the h–version multi-grid, a physical domain decomposition method [OPF93] (in contrary to decomposing only function spaces) and a more practical implementation by [FP93] for $p = 2$ with 'coarse' grid $p = 1$ and SOR iteration (a two-level multi-p method).

It is well known from domain decomposition, that any construction of a preconditioner as splitting into a linear or piecewise constant global function space and several higher order local spaces leads to such an h–independence under the condition of *minimal coupling* (see figure 21 on the facing page). Coupling comes into play localizing the global higher order function spaces. Hence we construct our preconditioner as the splitting into the global linear space and additional local spaces. To keep them local, we have to separate spaces for each edge, triangle, tetrahedron etc. Now we can interpret this preconditioner B as a block-diagonal version of the stiffness-matrix A. Calculating the preconditioned condition number $\kappa(B^{-1}A)$ we can see that it is majorized by the maximum of the local condition numbers $\tilde{\kappa}$ $(\frac{\max\lambda}{\min\lambda\neq 0})$ of the generalized eigenvalue problem $B_{\mathrm{loc}}x = \lambda A_{\mathrm{loc}}x$. $\tilde{\kappa}$ is calculated from the generalized eigenvalues orthogonal to the common eigenfunction of the eigenvalue 0. This means that we only have to optimize and calculate local condition numbers and the condition number is independent from h with the aid of a good preconditioner for the linear h–version problem.

1.4.3 Condition of the Preconditioned Stiffness Matrix

We want to construct the final version of our shape functions by using the polynomial vector spaces $\mathcal{P}_p^{d,\mathrm{sym}}$ and $\mathcal{P}_p^{d,\pm}$ of chapter 1.2.4 on page 34. The set of functions should maintain the symmetry and coupling properties of the original basis $\mathcal{B}_p^d$ and $\mathcal{B}_p^{d,\pm}$. P–hierarchy is guaranteed by the nesting of the vector spaces. The only missing property is a low condition number of the preconditioned system.

We make a general approach to optimization of the local condition numbers. The optimal polynomials are in a linear vector space $\mathcal{V} = \langle f_1, f_2, \ldots \rangle$. Every optimized polynomial v_k has a representation of

$$v_k := \sum_{i=1}^{k} q_{ki} f_i, \qquad k = 1, \ldots$$

We have to determine the coefficients q_{ki} that v_k has got the desired properties. We did use some direct procedures for minimization of the condition numbers.

For example we construct the cubic edge shape function on a triangle. The raw edge function f_9 is $b_0 b_1 (b_1 - b_0)$. We write the optimized edge

Table III: Condition numbers of the preconditioned stiffness matrix for $\mathcal{P}_p^{d,\pm}$

d	$p=2$	$p=3$	$p=4$	$p=5$	$p=6$	$p=7$	$p=8$	$p=9$
2	6.00	19.2	21.5	28.8	29.0	39.7	54.5	64.3
3	12.3	125.	132.	235.	467.	577.		
4	21.1	336.	467.	882.				

shape function v_9 as

$$v_9 = q_0 b_0 b_1 (b_1 - b_0) + q_1 b_0 b_1 + q_2 b_0^2 b_1 b_2 + q_3 b_0 b_1^2 b_3 + q_4 b_0 b_1 b_2^2$$

which implies hierarchy (from degree 2 to degree 3) and coupling (with triangles with common edges). The condition of $\mathrm{S}_3^{\pm}$ symmetry reads as

$$q_2 + q_3 = 0, \qquad q_1 = q_4 = 0$$

Normalizing the function v_9, there is only one parameter left, which can be obtained by an optimization procedure for condition number. The final result will be

$$v_9 \;=\; b_0\, b_1\, (\,-\,16.635193\, b_0\, b_2 + 7.277900\, b_0 + 16.635193\, b_1\, b_2 \,-\, 7.277900\, b_1)$$

All optimization procedures have in common the necessity of a correct management of the polynomials, their symmetry and their coupling properties. This includes the construction of the appropriate basis functions for each optimized shape function set. The optimized shape functions are a linear combination of the basis functions. The combination itself depends on the optimization. The actual basis functions $\tilde{f}_i$ are in some cases (optimized) shape functions v_k of previous optimization steps and in some cases symmetrizations of them.

We now compare the resulting local condition numbers. We choose the Laplace operator on the equilateral simplex. The condition numbers shown in table III were evaluated numerically.

We do not prove a special kind of asymptotics in p but we simply present the actual numbers of interest. We think that a prove would not only be intricated because of the structure of the function spaces $\mathcal{P}_p^{d,\pm}$, but dropping the constants it also would be of less practical worth for low p.

Nevertheless we obtain low local condition numbers. Hence an additional acceleration of an iterative solver would be obsolete. But our main result still are the new polynomial spaces $\mathcal{P}_p^{\mathrm{sym}}$ and $\mathcal{P}_p^{\pm}$ including their properties.

We present the shape functions, optimized for domain decomposition preconditioned stiffness matrices of chapter 1.4.3 on page 43. We show the functions belonging to $\mathcal{P}_p^{2,\pm}$.

Figure 22: *Symmetric hierarchic* polynomials f_0 and f_3 of $\mathcal{P}_1^{2,\pm}$ and $\mathcal{P}_2^{2,\pm}$

Figure 23: *Symmetric hierarchic* polynomials f_6 and f_9 of $\mathcal{P}_3^{2,\pm}$

1.4.4 Iteration Counts for Iterative Solution

The local condition numbers for the preconditioned system of equations are depicted in chapter 1.4.3 on page 43. They serve as an upper bound for the global condition numbers. We now present iteration counts for the iterative solution of global linear system of equations. We have to keep in mind the dramatic increase of un-preconditioned condition numbers for increasing p at high degrees p.

We already have presented an iterative two-level domain decomposition solver while constructing an optimal set of shape functions similar to [Man90b, BCMP91]. We now compare this one with another approach due to [Deu94]

Figure 24: *Symmetric hierarchic* polynomials f_{12} of $\mathcal{P}_4^{2,\pm}$ and f_{15} and f_{18} of $\mathcal{P}_5^{2,\pm}$

Figure 25: *Symmetric hierarchic* polynomials f_{21}, f_{24} and f_{27} of $\mathcal{P}_6^{2,\pm}$

only exploiting the history of refined grids called CCG. Optimal complexity $\mathcal{O}(n)$ of the solution up to discretization error has been proven for linear finite elements in [Sha94, BD95]. This of course carries over to higher order h–version with other (worse) constants. For p–version and h–p–version with

Figure 26: *Symmetric hierarchic* polynomials f_{30} and f_{33} of $\mathcal{P}_7^{2,\pm}$

Figure 27: *Symmetric hierarchic* polynomials f_{36}, f_{39} and f_{42} of $\mathcal{P}_8^{2,\pm}$

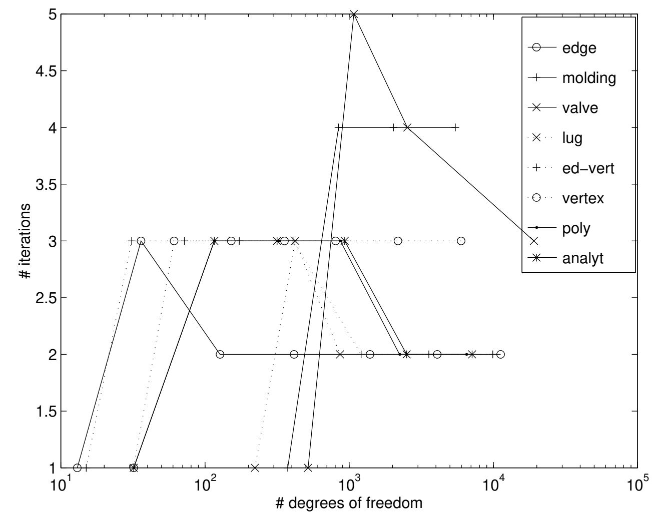

Figure 28: CCG iterations with diagonal scaling for adaptive h–version, estimated error .1

limited $p_{\max}$, we can obtain constants of the corresponding higher order h–version, which probably are of limited use for practical considerations and are not fully satisfying theoretically.

The CCG algorithm is originally equipped with a diagonal scaling. Hence we combine the pure CCG algorithm with the domain decomposition preconditioner derived. We choose a multi-grid $V_{2,2}$ cycle with 3×3 block symmetric Gauss-Seidel smoother for the approximate solution of the global system of linear elements arising in domain decomposition and use direct solvers for

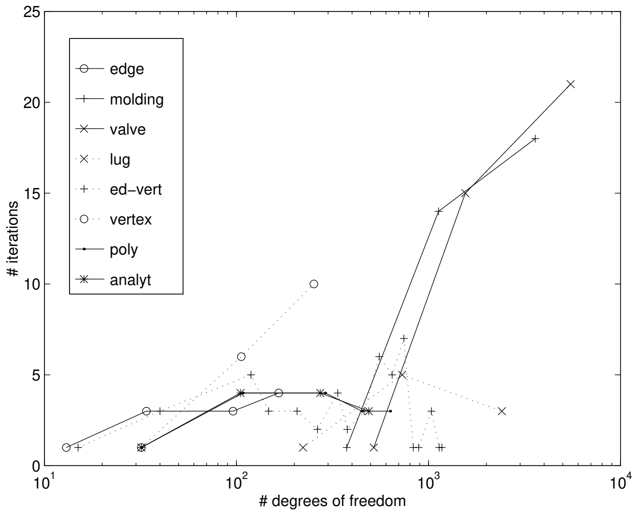

Figure 29: Ccg iterations with diagonal scaling for adaptive p–version, estimated error .1

the local linear systems connected with the geometric entities such as edges, triangles and tetrahedra.

We present iteration counts for different solvers, FEM versions and some examples described in chapters 3.2 on page 88 and 3.3 on page 100. Figures 28 on the page before for adaptive h–version, figures 29 and 30 for adaptive p–version and figures 31 and 32 for adaptive h–p–version (h–p ratio at $.7 \max h - err/p - err$). The figures show the number of cg-iterations per level. Each level is drawn at the number of degrees of freedom involved. Iteration counts start at 1 because of the 'one' iteration of the direct solver used on the initial coarse grid.

Figure 28 on the preceding page shows the iteration counts for adaptive h–version and Ccg solver with diagonal scaling. The number of iteration is at least 3 for the Poisson equation problems and at least 5 for elasto mechanics problems. The elasto mechanics test cases need higher iteration counts because of the 3 components of the solution vector, the 3×3 blocks in the stiffness matrix and the spectral properties at Poisson ratio .29 used. It is interesting to observe that the genuine 2D problem of the attachment lug ('lug') requires lower iteration counts than the full 3D elasto mechanics problems 'molding' and 'valve' although all problems are computed with

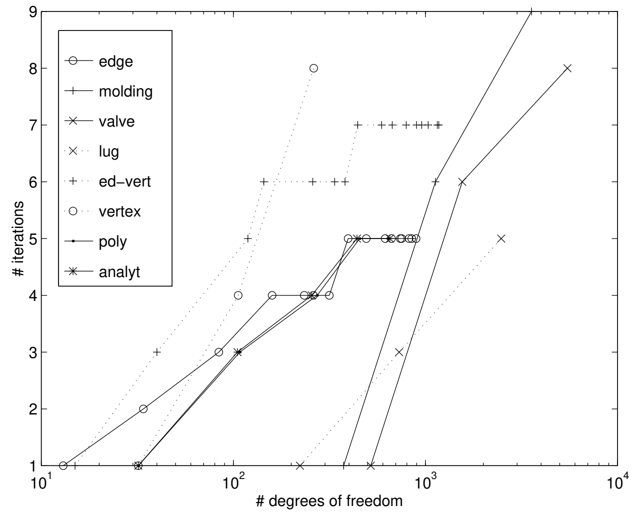

Figure 30: Ccg iterations with domain decomposition preconditioner for adaptive p–version, estimated error .1

3D FEM and full 3D refinement. All graphs show the typical behavior of a short increase of the number of iterations, an high peak at certain number of unknowns and a slow decay down to a ridiculously low number of iterations, also demonstrated in [BD95]. For h–version we do not apply domain decomposition preconditioning, because of the lack of higher p–extensions for space decomposition.

The next two figures show iteration counts for adaptive p–version with a forced minimum refinement of .3. The graphs for diagonal preconditioning, figure 29 on the preceding page show maximal iteration counts about 20 for elasto mechanics examples and below 5 for most of the Poisson equations. The almost polynomial solution and the analytic solution do not differ much. The vertex-singularity example requires more iterations than usual problems while approximation of the singularity remains bad during p–adaption. Even uniform p–version cannot resolve the singularity properly, as show in chapter 3.2.5 on page 97. This effect spreads out onto the behavior of the iterative solution procedure. The edge-vertex-singularity is also affected from the approximation drawback, but its linear algebra in conjunction with refinement shows an oscillatory behavior. The standard test examples show the typical peak behavior of Ccg, while the pure 3D elasto

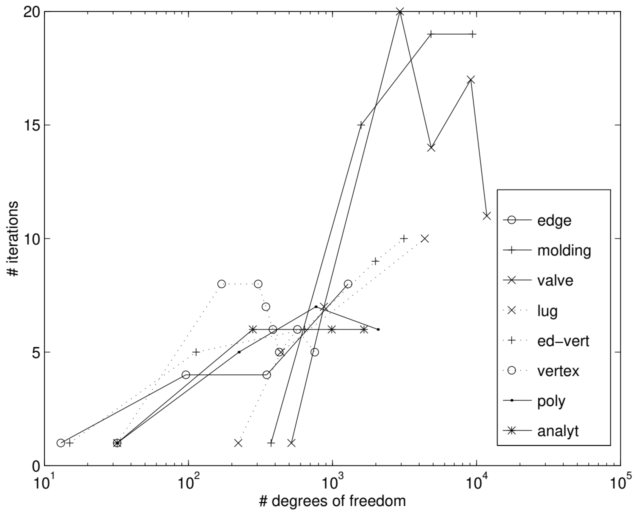

Figure 31: CCG iterations with diagonal scaling for adaptive h–p–version (ratio .7), estimated error .1

mechanics examples seem to be in a pre-asymptotic increase phase for .1 global energy error. Iteration counts are usually higher than for h–version because of worse condition numbers of the global stiffness matrix due to better approximation properties.

Figure 30 on the page before shows the number of iterations for domain decomposition preconditioned CCG. Iteration counts for elasto mechanics are generally cut down to 9 by preconditioning. The 'valve' problems reaction to preconditioning is better than for the 'molding' example, because the loads act in all three directions in contrast to the uni-axial loading in the 'molding' case. The vertex-singularity and the edge-vertex-singularity problems show high iteration counts as in the previous case without preconditioning. The graphs do not show the typical peak behavior of CCG any longer, but a monotone increase. Preconditioning has the effect of reduction of the condition number especially for low p, whereas the condition numbers increase with p. Hence the usually high iteration counts in the middle of the refinement history meet low condition numbers, lowering them and the usually low iteration numbers at the tail are increased by large increase of condition numbers. The computational grids (h and p distribution) may differ slightly, because the approximate solution does affect error estimation

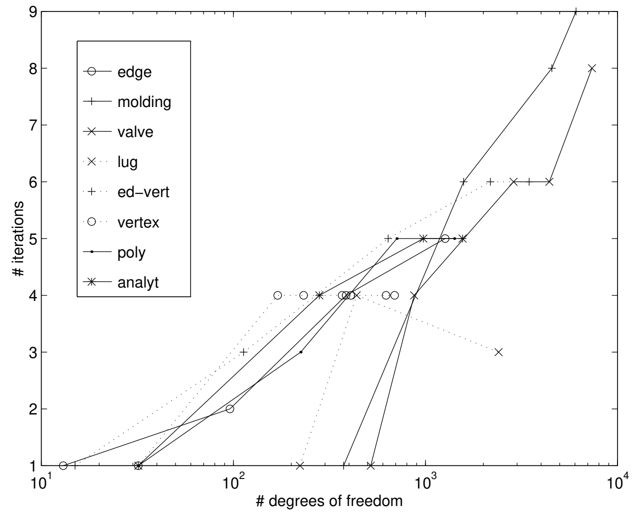

Figure 32: CCG iterations with domain decomposition preconditioner for adaptive h–p–version (ratio .7), estimated error .1

and therefore grid refinement.

The last two figure are on h–p–version. We used a .7 max ratio strategy. Iteration counts for diagonal scaling (without preconditioning) for elasto mechanics are below 20 and for Poisson problems are below 10. The genuine 2D mechanics example 'lug', it actually is mechanics, and the edge-vertex-singularity examples, challenging approximation problem, show relatively high numbers of iterations. The other test examples show typical peak-behavior of CCG. There is a small oscillation for the 'valve' example probably due to the refinement procedure. Absolute number of iterations are comparable or a little bit higher than for adaptive p–version, which is higher than for adaptive h–version, due to higher p's bad condition numbers and the improved approximation properties of the grids.

The domain decomposition preconditioned CCG for adaptive h–p version is presented in figure 32. For elasto mechanics examples the number of iterations is reduced down to a maximum of 9 and the attachment lug example does benefit, too. The positive effect of preconditioning seems to be comparable to preconditioning pure adaptive p–version. Graphs show the monotonic increase pattern already discussed. The different graphs look very similar, there are no drop outs due to approximation problems as for the p–

version. Preconditioning generally has an effect of smoothing the graphs and making the graphs of different examples approaching each other. Iteration history is regularized in this sense.

2 Algorithmic Constituents

In this chapter we are going to discuss some components of an adaptive h–p FEM code which deal with implementation, but which are of general use beyond simplex shaped finite elements. Issues like a posteriori error estimation, adaptive control and parallelization address rather general directions of actual research. They are reviewed from the h–p point of view. There are two more technical topics arising in the use of higher order elements. They are on the evaluation of polynomials in the code and export of polynomial solutions from the code, the first related to symbolic computation and the second one related to computer graphics.

2.1 Error Estimation and h–p–Refinements

2.1.1 Error Estimation

Putting this new shape functions into a framework of an adaptive or feedback finite element code, we have to consider some other details. We use an a posteriori error estimator to indicate those elements and regions to refine in the next step of the adaptive procedure. Assuming that our initial grid is fine enough, we can use an a posteriori error estimator. For a theoretic review of a posteriori error estimators c.f. [BEK93b]. Some famous error estimators can be considered as an extension of the finite element space by some shape functions and the approximate solution of the enlarged system by localizing domain decomposition preconditioning techniques. This holds for error estimators of Babuška and Rheinboldt [BR78], Bank and Weiser [BW85], Babuška and Miller[BM87] and Deuflhard,Leinen and Yserentant (DLY) [DLY89]. Other concepts of a posterior error estimates contain dual methods (c.f. comparison [ODRW89]) or asymptotic exact estimators [BY87].

We consider the edge based error estimators of DLY type. The original version is the extension of the space of linear functions by p–hierarchic quadratic edge functions. The stiffness matrix is splitted into blocks for linear functions (L) and quadratic functions (Q):

$$\begin{pmatrix} A_{LL} & A_{LQ} \\ A_{LQ}^t & A_{QQ} \end{pmatrix}$$

The linear system of equations is approximately solved by one iteration, using the preconditioner

$$\begin{pmatrix} A_{LL} & 0 \\ 0 & \mathrm{diag}(A_{QQ}) \end{pmatrix}$$

We want to calculate the coefficients of the quadratic shape functions, hence we only use the diagonal of A_{QQ} and the actual solution of the linear system. The spectral equivalence of the preconditioner and the stiffness matrix may be obtained by standard domain decomposition analysis techniques. For each edge we have to solve linear systems with one unknown. The most expensive part is the assembly of the required parts of the stiffness matrix and the right hand side terms. For more than one unknown per node, we either can use a block diagonal of A_{QQ}, or we can use the diagonal with less computational work, but less precision.

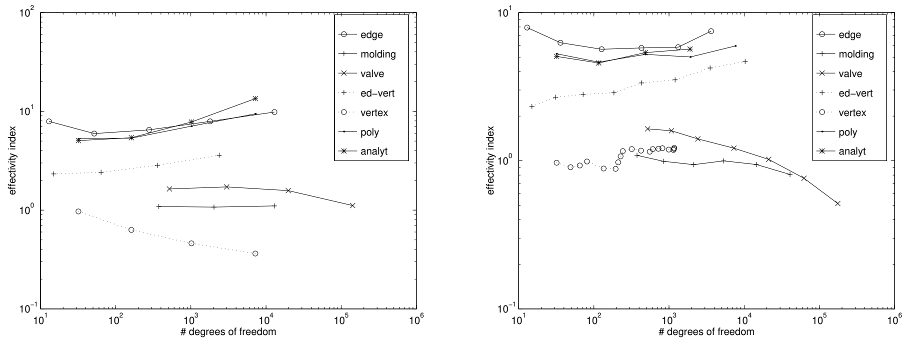

Figure 33: h–version's error estimator efficiency index for uniform version (left) and adaptive version (right), different test cases

The error estimator can be used in any space dimension d, and originally was proposed for 2D. For 3D there exists a comparison with another generalization of the error estimator [BEK93b], generalizing the property of bubble functions being applied to the boundary of the element which now are triangle faces. Hence that version uses quadratic edge functions in conjunction with cubic triangle functions. Both functions types have to be combined in the right manner, because they have different approximation properties. Numerical experiments show the superior performance and the higher cost of this estimator. It seems difficult to judge. We do not consider this variants further, because we do not use the pure cubic triangle shape functions needed and we want to have a concept applicable for different p.

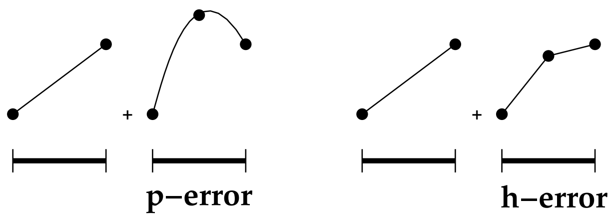

Figure 34: Edge oriented p– and h–error estimator

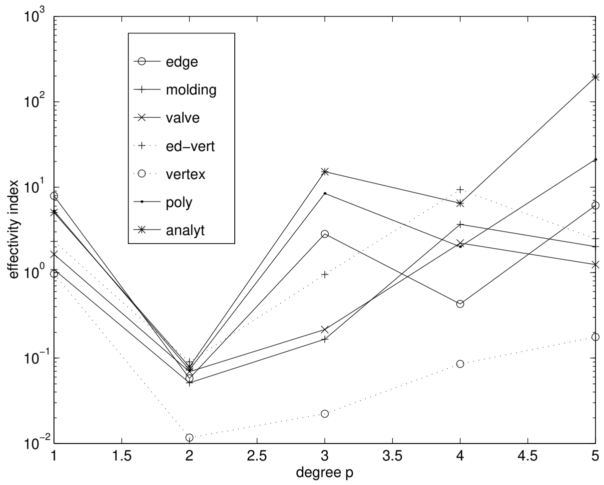

Figure 35: Error estimator's efficiency index for p–version (without scaling), different test cases

We have to generalize the concepts of error estimation to higher degrees p and to variable p. We keep the property of local one dimensional edge extensions. The existing function space is now called L and the space of edge functions, each one one degree higher than in L is denoted by Q. The whole mechanism of construction, proof and implementation is analogous. Although we have to keep in mind the constants for spectral equivalence for high p. We already have mentioned the increase of condition numbers for high p in the case of preconditioning. The same holds for the constants for error estimation. Nevertheless error estimates for high p in regular parts of the domain are quite precise because of the good approximation properties. The error estimator constructed will be named the p–error, for p–extension.

It has been mentioned in [BY87] that odd p FEM errors are dominated by jumps of derivatives across element boundaries, whereas even p FEM errors essentially are interior errors. This means that for odd p the edge oriented error estimators are supposed to work sufficiently well and for even p they might even underestimate the true error. This effect is numerically visible for degrees one and two in figure 35 on the page before. We will scale even order edge error estimates by some factors $f(p) > 1$ decreasing in p.

We present error estimators' efficiency indices (ratio of estimated error to true energy error) for some examples described in chapters 3.2 on page 88 and 3.3 on page 100. Figures 33 on page 54 present some numbers for uniform and adaptive h–version error estimation and figure 35 on the page before presents some for uniform p–version.

Left figure 33 on page 54 shows indices for uniform h–version. The lines are quite flat denoting rather constant deviation of the estimation from the true values. Hence for each test case there is a tuning parameter improving the error estimator just by scaling. Standard Laplace examples overestimate the error by a factor about 7. The elasto mechanics examples meet the error quite well. The vertex- and the edge-vertex-singularities deviate form the general outline. For the vertex-singularity additionally there is a dangerous underestimation of the error. This is due to bad estimates in the vicinity of the vertex-singularity, containing a major part of the global error.

Right hand side of figure 33 on page 54 contains analogous graphs for the adaptive h–version. The lines are smooth and constants are slightly better (nearer to 1). Generally indices are often lower than for the uniform refinement counterpart. This is because of the adaptive refinement, which can be interpreted as an optimization procedure for minimization of error estimates. The vertex-singularity graphs shows some oscillation due to arithmetic progression of the number of nodes. Adaptation does improve the error estimate even in this case, converging to 1. It is interesting to note that the minimization of error estimates by grid refinement does improve the quality of error estimates besides the reduction of the true error. A minimization of a measure totally uncorrelated to the true error would not cause this effect.

We already have mentioned some difficulties with even order FEM error estimation in figure 35 on the preceding page. This especially holds for quadratic elements. We generally obtain an oscillatory behavior in p. The trend of underestimation continues for quartic elements ($p = 4$) and even solutions such as the analytic, polynomial and edge-singularity problem. However in general there is no trend for higher order even p. These figures do not use any scaling applied to further tests.

Along the same line we construct another error estimator, called the h–error: We choose the space Q as one dimensional space extension located at the edges. Now this should be h–refinement which is bisection or the h–hierarchic hat-function. The constants of spectral equivalence are not as good as for the p–extensions, but we will use the estimations as indicators for decisions, whether to refine an element in h– or p–way. For refinement decision the p–error is used.

2.1.2 Standard Refinement Control

The goal for any adaptive refinement control is the minimization of computational work while guaranteeing a prescribed global precision. We already obtained local and global error estimates. We have to decide, whether to refine an element or not. This is sometimes called error indicator $\in \{0, 1\}$ in contrary to more precise error estimators $\in \mathbb{R}^+$.

First we consider pure h–version and pure p–version with binary decisions whether to refine an element. Minimum work is often relaxed to a condition of a low number of degrees of freedom with an estimated global error ϵ below a given tolerance tol. More precisely one has to take the number of estimation-refinement-solution cycles into account, which should be low, too. The condition of low number of degrees of freedom can be transformed into a low number of elements. Looking at the variation of an optimal grid , or introducing local optimality like Pareto optimality [Rhe80] one ends up with a requirement for an equidistribution of local (estimated) errors ϵ_i:

$$\max_{\text{grid } i} \epsilon_i \leq \min_{\text{father element } i} \epsilon_i$$

or more vague for n elements

$$\epsilon_i \approx \frac{1}{n}\epsilon = \overline{\epsilon_i}$$

This characterization directly leads to a very effective greedy algorithm, refining a suitable amount of elements carrying the biggest local errors. The elements are selected by a certain threshold θ,

$$\epsilon_i \geq \theta$$

There are many suggestions for the selection of the threshold θ.

- refine all, $\theta = 0$ is often used for p–version FEM [Sza85], if the global error ϵ does not match the required precision.

- fixed percentage c of elements is refined, choose θ with $\#\{\epsilon_i \geq \theta\}/n \geq c$. This guarantees a geometric progression of number of elements and hence some complexity bounds on the whole FEM algorithm [Ban94]. The median of the error distribution is $c = 1/2$.

- a fraction c of the maximum error, $\theta = c \cdot \max_i \epsilon_i$. Refining only a small number of elements, high $c < 1$ leads to optimal distributions of errors. We reduce only the biggest errors [Roi89a].

- the mean value of local errors, $\theta = c \cdot \epsilon/n$, which does not need a problem dependent tuning parameter using $c = 1$.

- fixed percentage c of the total energy error is refined, choose θ with $\sum_{\epsilon_i \geq \theta} \epsilon_i \geq c\epsilon$. This guarantees in conjunction with a proper initial grid the convergence of the whole adaptive FEM process due to [Dör94].

- some combinations of the criteria, e.g. a linear combination of $\max_i \epsilon_i$ and ϵ/n, while preserving a geometric prescribed progression of nodes (three parameters) [Lei90].

There is a modification, not just using the estimated local errors ϵ_i, but estimations for the gain of refinement [BR78]. This can be accomplished by means of extrapolation [ERFA91]:

$$\tilde{\epsilon}_i = \epsilon_i^2 / \epsilon_i^{\text{father}}$$

introducing another component for decision $\tilde{\epsilon}_i$. Now the maximum criterion also does not need a parameter

$$\tilde{\epsilon}_i \geq \theta = \max_i \epsilon_i$$

The transfer of error estimates evaluated on edges to elements has a smoothing property. The procedure of green closure of triangulations to avoid slave nodes smoothes the triangulation in a discrete sense, too. For p–version some $\max p$ conditions on common element boundaries smooth, too. The procedures enhance the approximation quality of the grids, leading locally to more regular grids with more accurate solution and error estimates again. Local failure of error estimates may be overcome by forcing refinement because of neighbor refinement. Any resulting slight over-refinement does degrade performance of the adaptation process.

Asymptotically $\epsilon \to 0$ the different strategies all deliver the same local optimal grids [Rhe80]. The actual performance is guided by the quality of error estimation and the adaptation of tuning parameters towards the test case. There are also considerations about grid optimization instead of error estimation like [Jar86].

2.1.3 h–p Refinement Control

Besides indication of refined elements, we have to perform another task, identifying h refined elements and p refined ones. First we have to state, that the criterion of optimality may be modified. The derivation of the equidistribution of local errors uses an assumption of equal work per element. A generalization is given in [ROD89] using an equidistribution of error per work

$$\hat{\epsilon}_i = \epsilon_i / P(w_i), \ P \text{ polynomial}$$

for which we have to introduce some measures for work load depending on the local polynomial degree p_i:

$$w_i \approx \binom{p_i + d}{d} = \mathcal{O}((1 + p_i)^d) \text{ for varying } p_i$$

w_i is the number of unknowns, w_i^2 is the number of non-zeros in the stiffness matrix and w_i^3 is the complexity of local matrix diagonalization. Following [Deu83] we instead have to consider terms of

$$\log \hat{\epsilon}_i = \log \epsilon_i / P(w_i), \ P \text{ polynomial}$$

We now can apply the previous strategies with threshold θ and estimated local errors for each local h–refinement $\tilde{\epsilon}_i^h$ and p–refinement $\tilde{\epsilon}_i^p$, considering the local work w_i^h and w_i^p for both possible decisions. Actually there is a third possibility of combined refinement $\tilde{\epsilon}_i^{hp}$ and w_i^{hp}. The workloads may be estimated by

$$\begin{array}{rcl} w_i^h & = & w_i 2^d \\ w_i^p & = & w_i (\frac{2+p_i}{1+p_i})^d \\ w_i^{hp} & = & w_i (2\frac{2+p_i}{1+p_i})^d \end{array}$$

We can take different extrapolations of old and actual h–error and p–error for ϵ_i^h, ϵ_i^p and ϵ_i^{hp}. We now employ the usual θ threshold strategy with iteration over all elements $i \in \{1, \ldots, n\}$ and no, h, p and $h-p$ refinement for each element.

We also introduce a different approach, without history and extrapolation with old error estimates. History of local errors is not available in a startup phase of the algorithm. In 3D we do not perform many refinement steps, so history is of limited benefit. It may be inaccurate and misleading, if one tries to compare errors for a previous p–refinement with a previous h–refinement, or h–refinement with different orders p. Hence we do not use any history of local errors. We have to base the decision on estimated p– and h–errors and on the known workloads. The h–error is of low global quality, so we use the p–error for standard refinement decision. Afterwards we decide, whether to refine an element the h– or the p–way. Now we employ a quotient of h–error and p–error, using usual threshold techniques.

For other considerations about h–p grid generation see Gui and Babuška [GB86a, GB86b, GB86c], Oden [Ode94], Rachowicz, Oden and Demkowicz [ROD89] and A. Hohmann [Hoh94]. Similar work on best bases selection is done in wavelet context c.f. [CW90].

2.2 Parallelization

A great part of all parallel finite element research efforts are directed towards parallel linear algebra. Domain decomposition methods are considered as decomposing the physical computational domain into pieces mapped to processors. However it turned out, that standard multilevel and multi-grid methods parallelize quite good [BPX90, Zum91, Gri94] and many implementations like [Bas93, Lem94] and are often faster than excessive cutting lines of information by lower communication methods like domain decomposition. Hence we only consider parallel implementations of multilevel finite element methods, which are identical to the scalar ones (up to rounding errors).

Let us assume there is an efficient parallel implementation (data parallel) of an adaptive multilevel h–version finite element code available. We now want to add the p–version capabilities described (like [BEM92]), towards implementing an adaptive h–p–version. For polynomial degree $p > 1$ we add local work and we increase the local stiffness matrices, which are dense. For uniform p global and local work increases. Meanwhile data at the boundary of elements or element patches decreases relatively. Hence parallel efficiency will get better. If we now consider varying degrees p, we introduce some imbalance of work. Hence an adaptive partitioning algorithm has to take the different local work into account, which is only a minor modification in most cases.

Properties of a parallel adaptive h–p code strongly depend on the efficiency of the underlying parallel adaptive h code. We want to mention some steps towards such a parallel adaptive code. We want to make some distinctions, sorted form 'easy' to 'still to do':

- non parallel codes: all kinds of adaptive and uniform h and p codes, h–p codes with restrictions mentioned (usual scalar codes)
- fully adaptive parallel codes on shared memory computers, without data partitioning but with coloring of data [Lei90, Bir93], performance is limited by computer hardware restrictions (not scalable)
- parallel codes on (uniform refined) structured grids: main part of all parallel finite element codes, grid partition is easy (just use High-Performance-Fortran, Parmacs, ... , excellent efficiencies)
- parallel codes on (uniform refined) unstructured grids: grid partition is calculated (often scalar) before actual computation (use good heuristics like [PSL90])
- parallel codes on a set of non-uniform refined unstructured grids, given a priori: grid partition for the whole set is calculated before actual computation (for block structured grids, with adaptivity in the scalar case [Lem94], with interfaces for dynamic repartitioning and with adaptivity in the scalar case [Bas93])
- parallel codes on adaptively derived non-uniform refined unstructured grids: grid partition for the initial grid can be calculated before actual computation, following grid partitions have to be done while parallel processing (without repartitioning, no multi-grid linear algebra [BSS91, Wil92, JP92])

We conclude that there are efforts towards adaptive parallel codes, but existing ones are either parallel or adaptive. Looking at parallelization of unstructured finite element codes with different grids or evolution turns out the enormous implementational efforts to maintain data and data structures correct in parallel. Of course it is easy to use a very high level of abstraction, partitioning single elements with a comfortable numeric library sacrifying performance and efficiency. However gains of parallel computing for this kind of problems are supposed to be so small, that full optimization of the parallel code has to be applied. This means a very intricated and error prone implementation. But there is a more general problem unsolved. Following [Zum91], we mention some complexity bounds for h–version involved. n denotes the number of unknowns, P is the number of processors. The nested grids are assumed to have a geometric progression of nodes, we have $\mathcal{O}(\log n)$ different grids.

We assume a typical two-level non uniform memory access computer architecture. Local memory access is assumed to have a linear access patern and it is sufficient to count the number of operations performed ('computation'). Global memory access is sensitive to both, volume of data accessed ('data to transfers') and the number of block read or write operations ('number of transfers'). This pattern can be motivated either by message-passing or by a cache coherence protocol for shared memory and is found to be very precise [Zum91]. All numbers are complexity bounds neglecting constants, where n or π turns to infinity ($n \geq \pi$).

description	scalar	parallel computation (π processors)	number of local transfers	local data to transfer
assembly of local matrices	n	n/π	0	0
coupling to global matrices	n	n/π	1	$\leq n/\pi$
matrix multiply	n	n/π	1	$\leq n/\pi$
scalar product	n	$n/\pi + \log \pi$	$\log \pi$	$\log \pi$
multilevel preconditioner	n	n/π	1	$n/\pi + \log n$
multi-grid preconditioner	n	n/π	$\log n$	$n/\pi + \log n$
fixed number of precond. cg iterations	n	$n/\pi + \log \pi$	$\log \pi$ $(+\log n)$	$n/\pi + \log n$ $+\log \pi$
error estimation	n	n/π	1	$\leq n/\pi$
marking elements for refinement	n	$n/\pi + \log \pi$	$\log \pi$	$\log \pi$
refining grid without 'green' closure	n	n/π	1	$\leq n/\pi$
nested iteration cycle, without repartitioning	n	$n/\pi + \log n \log \pi$ $(+\log^2 n)$	$\log n \log \pi$ $(+\log^2 n)$	$n/\pi + \log^2 n$ $+\log n \log \pi$
one grid bisection, by Lanczos iteration	n	$n/\pi + \log \pi$	$\log \pi$	$n/\pi + \log \pi$
one grid partitioning, nested bisection	$n \log \pi$	$n \log \pi/\pi$ $+\log^2 \pi$	$\log^2 \pi$	n/π $+\log^2 \pi$
partitioning of all grids	$n \log \pi$	$n \log \pi/\pi$ $+\log^2 \pi \log n$	$\log^2 \pi \log n$	n/π $+\log^2 \pi \log n$

The grid partitioning algorithms known have a complexity $\mathcal{O}(n \log \pi)$ for recursive spectral bisection [PSL90] and modifications such as [WCE95] with references therein and even higher $\mathcal{O}(n \log n)$ for Hilbert curve index based partitioning like [PO93] and [KOR95] (in this reference, adaptive means variable computational environment), a complexity which is larger than of the complete numerical algorithm itself. Just looking at complexities may be misleading here, because a parallel version of a one dimensional index based method with a high scalar and parallel complexity of $\mathcal{O}(n \log \pi + n/\pi \log n)$ (parallel sort) often is much faster than the very good partitions obtained by recursive bisection algorithms. Although for massively parallel computations the algorithms can not be used without special care. Clustering techniques [Bas93, WCE95] trace the problem to lower granularity and put the limits of $\log \pi$ to higher π. Another improvement is the consideration of 'scaling up' a problem, denoting n/π is fixed while n turns to infinity. For a parallel computer with fast communication hardware, the number of transfers may be neglect-able. There is a hope that fast communication is a future direction of parallel computing. Depending on the problem solved and the hardware used, one may trade off granularity versus quality of partitioning. But at the end, a general solution for the $\mathcal{O}(n)$ partitioning problem (if ever possible) still has to be found.

2.3 Evaluation of Shape Functions

We want to mention one implementational detail concerning the numerical evaluations of the shape functions. [Pea76] used monomials for performance reasons, using less arithmetical operations per evaluation. However we use a fixed set of integration formulas in the finite element code and therefore a (greater) fixed set of evaluation points (in barycentric coordinates) a priori known. We simply store the values of the shape functions at these points in a table. Restoring these values is of course cheaper than calculating them. The table may be set up without any connection to a finite element run.

Actually we have to compute the table once. We could have done this completely in the symbolic algebra package used for computing the shape functions itself, put we did this in the code itself, by means of symbolically generated code. But at the first time the was a drawback concerning a bad code optimization of the code generator. Consider the following example, a general polynomial of degree 3 in two variables:

$$f = \sum_{i+j \leq 3} a_{i,j} x^i y^j$$

or in computer input

```
f:=a30*x**3+a03*y**3+a21*x*x*y+a12*x*y*y+
```

```
a20*x*x+a11*x*y+a02*y*y+a10*x+a01*y+a00;
```

We get generated C source code without optimization like this:

```
h=pow((double)x,3.0)*a30+pow((double)x,2.0)*y*a21+
pow((double)x,2.0)*a20+x*pow((double)y,2.0)*a12+
x*y*a11+x*a10+pow((double)y,3.0)*a03+
pow((double)y,2.0)*a02+y *a01+a00;
```

using the amount of arithmetic operations: 5 power, 12 mult, 9 add.

Turning on optimization, we are able to drop the power operations:

```
double a00,y,a02,a12,x,a03,a11,a21,a20,a30,a01,a10,h;
h=a00+y*y*(a02+a12*x+a03*y)+y*x*(a11+a21*x)+
x*x*(a20+a30*x)+a01*y+a10*x;
```

using the amount of arithmetic operations: 12 mult, 9 add.

Simply using Horner's scheme (by hand) for polynomial evaluation recursively for each variable, we are able to reduce computational effort to

```
h=((a30*x+(a21*y+a20))*x+((a12*y+a11)*y+a10))*x+
(((a03*y+a02)*y+a01)*y)+a00;
```

which is an amount of arithmetic operations: 9 mult, 9 add.

The Horner option has been added to the new release of the symbolic package used.

2.4 Post-processing

Some pictures showing results from finite element calculations are made by a standard finite element post processor. It uses linear interpolation between given data at control points connected with nodes or finite elements. For h–version of finite element this is an exact representation of the solution because of the linear shape functions used. p–extensions are not used in graphics. This may be dangerous in the presence of large elements with high p used in p– or h–p–version.

It is of course possible, to project the calculated solution onto a refined h grid for export to graphics. However the structure of the calculation grid would be lost in the post-processor and we applied p– and h–p–version for the sake of efficiency, which we do not want to sacrify for graphics. To fix the later one, one could reduce precision for post-processing, using less refined h–grids. But there is no convenient way of determining a priori, what part of the data has to be precise for interactive post-processing.

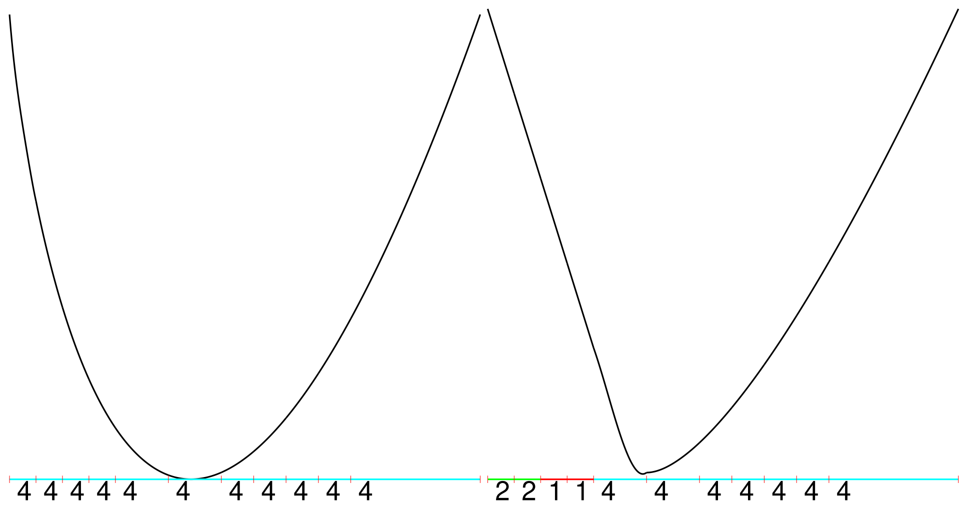

Figure 36: two components of a solution of an h–p adaptive procedure, common h–grid and different distribution of p, numbers denote polynomial degrees

We will present some steps towards post-processing including higher p and using standard formats of data from computer graphics in 2D following [Zum94]. In one dimension we just show an output by straight lines (figure 36). The step size for graphics (polynomials $p \geq 2$) is chosen in dependence of the resolution of the output device.

If the graphical results of the piecewise linear representations are not satisfying and we want to increase precision, we have to supply additional data. This can be done by using patches with a higher level of precision. In order to improve the patches, we choose higher order B-splines and rational B-splines, which are both heavily used in computer graphics. Any component of a solution is a surfaces, that has to be global continuous. The surfaces are analytic on each local finite-element.

2.4.1 NURBS

We now concentrate on the exact graphical representation of higher order polynomial shape functions stemming from a finite element code. We will use one complete description of a NURBS for each finite element ($p \geq 2$) according to a standard computer graphics data format. We will see that every polynomial of degree p on a triangle is exactly contained in a NURBS of degree p.

If we use central projection for representing the surface, which is in $\subset \mathbb{R}^3$, starting with polynomials, we have to do the remaining parts of computation with rational functions. Hence we can operate on rational functions anyway. They transform to rational functions of the same degree under central perspective transformations. Orthographic perspective is considered as a special case of central perspective. Rational functions sometimes are efficiently implemented in graphic subsystems.

We introduce the projective space $\mathbb{E}^4$ based on $\mathbb{R}^4$ by the relation

$$\vec{x} = \lambda \vec{y} \mod \mathbb{R}^4 \quad , \lambda \in \mathbb{R}, \lambda \neq 0 \quad \Rightarrow \quad \vec{x} = \vec{y} \mod \mathbb{E}^4$$

Now we can write central projection in projective coordinates, which is the way it is typically implemented in graphics hardware

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{pmatrix} \mapsto A\vec{x} = \begin{pmatrix} \vec{a}_1 \vec{x} \\ \vec{a}_2 \vec{x} \\ \vec{a}_3 \vec{x} \\ \vec{a}_4 \vec{x} \end{pmatrix} \equiv \begin{pmatrix} \vec{a}_1 \vec{x} / \vec{a}_4 \vec{x} \\ \vec{a}_2 \vec{x} / \vec{a}_4 \vec{x} \\ \vec{a}_3 \vec{x} / \vec{a}_4 \vec{x} \\ 1 \end{pmatrix}$$

with transformation matrix

$$A = \begin{pmatrix} \vec{a}_1 \\ \vec{a}_2 \\ \vec{a}_3 \\ \vec{a}_4 \end{pmatrix} \in \mathbb{R}^{4,4}$$

We now want to define NURBS. Using B-splines in 1D (curves), we can define every spline by its set of control points x_i and its weights ω_i.

$$f(t) = \sum_{i=0}^{n} \omega_i B_{x_i}^n(t)$$

The B-Splines may be implemented using de Casteljau's algorithm ([DH93], for further reading [Far90]). This may be generalized to two parameter surfaces on regular grids of control points. The simplest one is a tensor product approach for quadrilaterals:

$$f(t,u) = \sum_{i=0}^{n} \sum_{j=0}^{n} \omega_{i,j} B_{x_i}^n(t) B_{y_j}^n(u)$$

Inserting this B-spline surface into central projection,

$$\vec{f}(t,u) = \sum_{i=0}^{n}\sum_{j=0}^{n} \vec{\omega}_{i,j} B_{x_i}^n(t) B_{y_j}^n(u) \in \mathbb{R}^4$$

leads to rational B-splines. If the control points x_i, y_i are not equidistant, we have got 'non-uniform rational B-splines', called NURBS.

We want to represent one polynomial of degree p by a NURBS. Hence we transform the piecewise rationals into a meromorph function choosing in 1D the set of $p+1$ control points

$$(x_0, x_1, x_2, \dots, x_p) = (0, 1/p, 2/p, \dots, 1)$$

which corresponds to a set of spline 'knots' actually used

$$(k_{-p}, k_{-p+1}, \dots, k_0, k_1, \dots, k_{p+1}) = (0, 0, \dots, 0, 1, \dots, 1)$$

We turn the meromorph functions into a polynomial by the weights

$$(\vec{\omega}_{i,j})_4 = 1$$

For uniform interpolation we set

$$(\vec{\omega}_{i,j})_1 = x(b_1 = i/p, b_2 = j/p), \quad (\vec{\omega}_{i,j})_2 = y(b_1 = i/p, b_2 = j/p)$$

with x and y being the coordinate system of the finite element We are left with a transformation of the actual solution to a set of

$$(\vec{\omega}_{i,j})_3$$

which is unique and linear.

2.4.2 Trimming Bivariate NURBS

Unfortunately in most cases only the bivariate NURBS for the quadrilaterals are implemented in graphical subsystems (Iris GL, OpenGL, Hewlett-Packard Starbase), but we need them on the triangle. NURBS on the triangle are determined by a lower number of coefficients (control points), but they are completely contained in the space of NURBS on the quadrilaterals of the same degree and the double area, analogously to the polynomial spaces. Hence we can compute the coefficients of the NURBS on the quadrilateral instead of the ones on the triangle. For graphical representation we

Figure 37: A bivariate NURBS on a square, automatically resolved into patches and a triangular trim curve for it, hardcopy from screen.

have to remove one half of the quadrilateral which can be done by 'trimming'. We restrict the domain of parameters to a subset of $[0,1]^2$ bounded by NURBS curves called trimming curves. This is shown in the sequence of figures 37 and 38 on the next page. The NURBS is defined in parameter space on $[0,1]^2 \rightarrow \mathbb{E}^4$. We can restrict this space to a domain $T \subset [0,1]^2$ defined as the interior of an oriented trimming loop. In our case T will be the triangle $T = \{(x,y)|x, y, (1-x-y) \geq 0\}$.

2.4.3 Adaptive Tessellation

There is a general problem concerning higher degree polynomials: We cannot render an higher order surface directly, if the degree p of the polynomials exceeds three. Hence we apply adaptive subdivision of patches until the area of display is resolved fine enough (display space), or the object itself is resolved fine enough (object space) (figure 39 on page 70). We postponed this problem to the graphical software package (here Iris GL). We only supply the coefficients of the polynomial in a NURBS representation and we supply a parameter $c \in [0,1]$ for adaptivity in display space, called 'geometric complexity' (figures 40 on page 71 and 41 on page 72).

Figure 38: The bivariate NURBS trimmed to a NURBS on a triangle, additionally with texture mapping in an interactive scene viewer tool

We now show the influence of the polynomial degrees onto the tessellation (figure 41 on page 72). The graphical object of the figures itself comes from h–p–version finite element calculation on a 2D L-shaped domain with non-uniform degree p and is a rather rare one in graphical representation. Up to rounding errors, the functions on the boundary of the patches are equal, because the global function is approximated continuously. Hence the dense subdivision of quadratic elements which have an edge in common with a linear element do not produce errors in continuity. A key point is either a stable (unique) representation of the polynomials or a high precision of the coefficients. Relying on precision, excessive subdivision or excessive enlargement of the picture will result in display errors. Such errors will always occur on edges of two triangles with different represented polynomials or different subdivided polynomials, which is the case for different polynomial degrees.

A general solution for this problem is not a change of the NURBS basis. There are two alternatives: One can use the original representation of data of the finite element method and do the adaptive tessellation in a global conforming way. This can be done in a approach like [WR92]. Another way

Figure 39: A solution on an L-shaped domain with quadratic triangle shaped elements. It is automatically resolved into patches. We show a global view and a detail of the adaptive tessellation. The picture enlargement triggers a finer tessellation.

is used in [RR93a, RR93b], where the NURBS basis is retained, but additional information concerning topology of the tessellation is supplied. The renderer 'knows', that the surface is continuous and that specific functions have to be equal on an edge. In the case of critical decisions for drawing, an algorithm can use this information by some kinds of modified skyline algorithms, which cannot miss a pixel.

2.4.4 Texture Mapping

We want to show results using a second technique of computer graphics, which does not depend directly on NURBS but is easily used in conjunction with NURBS, which is texture mapping. Texture mapping is algorithmically simple, but computationally intensive. It can be supported by hardware [Sil] and it has the same measure of complexity $\mathcal{O}(n)$, like usual hidden line Z-buffer algorithm. We are able to transmit additional information via texture mapping. It is a natural generalization of the linear color lookup tables, which are widely used.

Figure 40: Quadratic triangle shaped elements, geometric complexity for adaptive control of the tessellation is set to 0.01 (detail) and to 0.9.

We use a linear mapping of coordinate space $\mathbb{R}^2$ into the periodic parameter space of the texture defined in $[0, 1]^2$. Some pictures like the brick example (figure 42 on page 73) give a realistic impression of an artificial object in $\mathbb{R}^3$ depicting the solution surface. The feature texture mapping extends the imagination of the surface plots. Here we can easily perceive curvature and relative height. In the square example (also figure 42 on page 73) we are able to observe a widening and a thickness of the lines depending on their distance to the observer. This is a very intuitive effect similar to a textured membrane (a balloon) put onto the surface. Choosing a proper texture, one add text and annotations onto the texture, inscriptions like name of the axis and numerical values, and graphics like contour lines.

We continue with linear texture mapped views of the geometric object, modifying the parameterization of the texture $T \subset [0, 1]^2 \to \mathbb{R}^2$. We can distort the map in the x, y–plane which leads to line plots instead of grid plots. The lines can have different thickness and focal points in the picture (figure 43 on page 73). We also introduce additional components of a vector valued solution. They are used as a component of parameterization by a similar NURBS representation, x left and $x\&y$ right in figure 43. Ge-

Figure 41: L-shaped domain with linear, quadratic and cubic triangle shaped elements. Linear shape functions are not resolved into smaller patches because they are drawn exactly. Adaptive tessellation.

ometrically this is a generalization of one dimensional color lookup table, introducing a 2D table which is the texture.

Figure 42: L-shaped domain with quadratic triangle shaped elements, textured with a squared pattern and with bricks.

Figure 43: L-shaped domain with quadratic triangle shaped elements, texture mapping is solution dependent.

3 Numerical Experiments

In this chapter we present some numerical data and experiments concerning the constituents of the adaptive multilevel h–p–finite element code. We will show numerical comparisons of condition numbers for different families of shape functions and compare the approximation of different FEM versions for some synthetic test cases and some applications to linear structural mechanics.

3.1 Comparison of Condition Numbers

We compare condition numbers of the local stiffness matrices for different families of shape functions discussed in chapter 1.1 on page 13. The numbers increase dramatically for higher degrees p. With this background knowledge we look at the condition of the preconditioned system. The local condition numbers were obtained in chapter 1.4.3 on page 43 already, and we have presented resulting effective iteration counts for the global linear system of equations in chapter 1.4.4 on page 45.

3.1.1 1D Condition Numbers

We now compare the resulting local condition numbers like in [BGP89, CMTT93]. We choose the Laplace operator Δu (i.e. $a_0 \equiv 0$ and $a_{ik} \equiv I$) and vary the geometry of the elements. Equivalently we could have changed the differential operator keeping the element fixed. The absolute scaling of the elements is not relevant, hence we only change angles and aspect ratios. The condition numbers are evaluated numerically.

We compare the condition numbers of the local matrices for the Laplace operator on the interval $[0, 1]$. We may draw conclusions from this 1-D case for edge based shape functions in higher dimensions.

The polynomials compared are (see chapter 1.1 on page 13):

- the monomials x^p for $p \in \mathbb{N}_0$
- p–hierarchical polynomials proposed by [ZT89], which are of Hermitean type till degree two

$$f_p(x) = \begin{cases} \frac{1}{p!}(x^p - 1) & p \text{ even} \\ \frac{1}{p!}(x^p - x) & p \text{ odd} \end{cases}$$

- normalized polynomials ϕ_i. ϕ_i is substituted by $\frac{\phi_i}{\sqrt{\langle \phi_i, \phi_i \rangle}}$, if the (semi-) norm of the polynomial is not zero ($\sqrt{\langle \phi_i, \phi_i \rangle} \neq 0$).

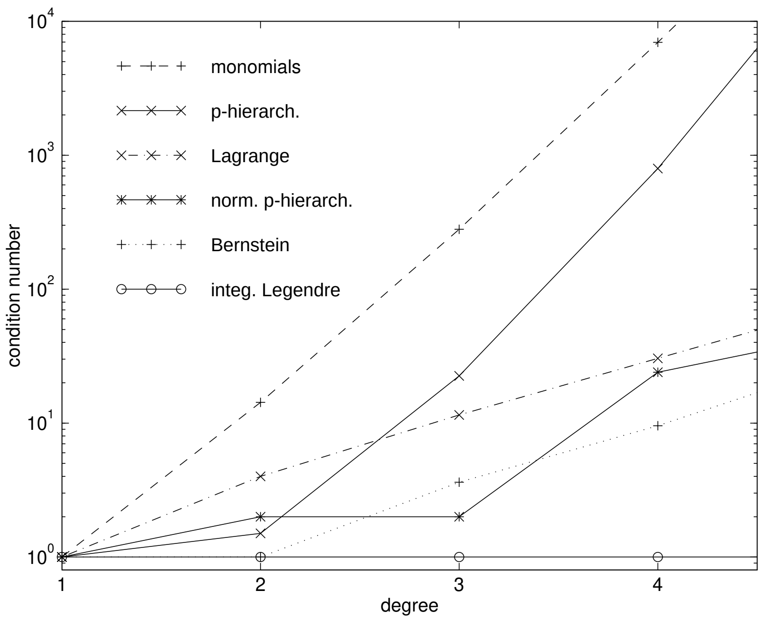

Figure 44: Local condition numbers for the Laplace operator on the interval for different polynomials of degree 1 to 4 (detail)

- Lagrange polynomials which are interpolation polynomials for the equidistributed interpolation points $x_i = i/p$, $i \in \{0, 1, \dots, p\}$ on the interval $[0, 1]$.

$$f_{ip}(x_i) = \begin{cases} 1 & \text{for} \quad i = p \\ 0 & \text{for} \quad i \neq p \end{cases}$$

- Bernstein polynomials of degree p [Far90]

$$f_{ip}(x) = \frac{p!}{i!(p-i)!}(1-x)^i x^{p-i}$$

- the integrated Legendre polynomials on $[-1, 1]$ [SB91]

$$f_p(x) = \begin{cases} \frac{1 \pm x}{2} & \text{for} \quad p < 2 \\ \frac{1}{2(2p-1)}\left(f_p(x) - f_{p-2}(x)\right) & \text{for} \quad p \geq 2 \end{cases}$$

with Legendre polynomials f_p defined in example (1.1.3 on page 17).

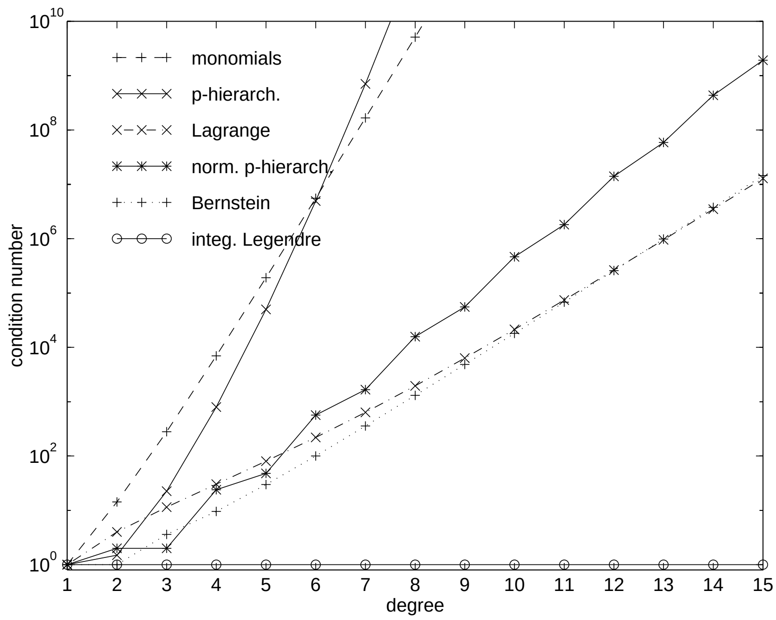

Figure 45: Local condition numbers for the Laplace operator on the interval for different polynomials of degree 1 to 15

We observe the following: For degree one all condition numbers start with a low number. For degree two, the quadratic case, the condition numbers diverge, but not with the same pattern as in the asymptotic case. The Bernstein polynomials keep optimal for degree two. The p–hierarchical polynomials are slightly better than the normalized ones in the quadratic case, but are outperformed for higher degrees. The integrated Legendre polynomials keep the optimal constant condition number due to their construction.

The numbers of the monomials and the p–hierarchic polynomials increase drastically. At first the monomials are worse, but are caught at degree 6, getting unacceptable for higher degrees. The normalized p–hierarchic polynomials perform better. One can obtain a "staircase effect" for odd and even degrees. The non-hierarchical Bernstein and the Lagrange polynomials asymptotically show the same behavior.

3.1.2 2D Condition Numbers

Figure 46: Local condition numbers for the Laplace operator on the *equilateral triangle* for different polynomials of degree 1 to 10

We compare the condition numbers of the local matrices for the Laplace operator on the triangle. The results only depend on the angles of the triangle. The symmetric hierarchic shape functions in this section and the next one are optimized for pure condition numbers [Zum93], instead of preconditioned condition numbers of the last chapter.

The polynomials compared are (see chapter 1.1 on page 13):

- the *monomials* $x^i y^j$ with $i+j=p \in \mathbb{N}_0$
- the orthogonal *(ortho. 1)* polynomials proposed by [GM78] for the d–simplex:

$$f_\alpha = \sum_{\beta \leq \alpha} (-1)^{|\alpha|+|\beta|} \frac{(d-1+|\alpha|+|\beta|)!}{(d-1+2|\alpha|)!} (\alpha-\beta)! \Big(\frac{\alpha!}{(\alpha-\beta)!\beta!} \Big)^2 b^\beta, \alpha, \beta \in \mathbb{N}_0^d$$

- the orthogonal *(ortho. 2)* polynomials proposed by [AF26] for the d–simplex

$$f_\alpha = \partial^\alpha \Big((1 - \sum_{j=1}^{d} b_j)^{|\alpha|} b^\alpha \Big), \alpha \in \mathbb{N}_0^d$$

Figure 47: Local condition numbers for the mass-matrix on the *equilateral triangle* for different polynomials of degree 1 to 10

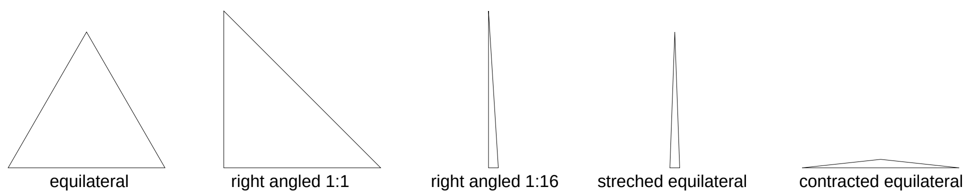

Figure 48: The different triangle shapes

with derivatives ∂^i with respect to b_i.

- *normalized* polynomials ϕ_i. ϕ_i is substituted by $\frac{\phi_i}{\sqrt{\langle \phi_i, \phi_i \rangle}}$, if the (semi-) norm of the polynomial is not zero ($\sqrt{\langle \phi_i, \phi_i \rangle} \neq 0$).

- a modified version of the p–*hierarchical* polynomials proposed by [ZT89], which are a generalization of the one-dimensional polynomials. We have the linear functions $\{b_0,\ b_1,\ b_2\}$ and the edge-functions gener-

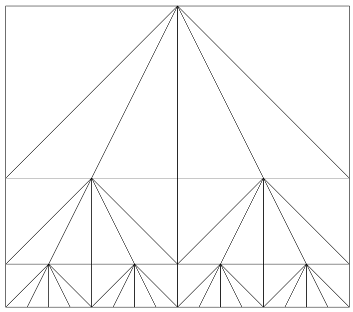

Figure 49: triangles with large angles

Figure 50: Local condition numbers for the Laplace operator on the *right-angled triangle* with short edges of length 1 : 1 and 1 : 16 for different polynomials of degree 1 to 10

ated by rotations (cyclic permutations) of:

$$f_p(x) = \begin{cases} \frac{1}{p!}\left((b_2 - b_1)^p - (b_2 + b_1)^p\right) & p \text{ even} \\ \frac{1}{p!}\left((b_2 - b_1)^p - (b_2 - b_1)(b_2 + b_1)^{p-1}\right) & p \text{ odd} \end{cases}$$

Additionally, the internal function $b_0 b_1 b_2$ is proposed for degree 3 and

Figure 51: Local condition numbers for the Laplace operator on the *distorted equilateral triangle* stretched and contracted by a factor of 16 for different polynomials of degree 1 to 10

the functions $\{b_0(b_0b_1b_2),\ b_1(b_0b_1b_2),\ b_2(b_0b_1b_2)\}$ for degree 4 as a substitution for it. Hence the functions are not p–hierarchical any longer. We modify this step using the the usual monomials multiplied by the "bubble"-function $b_1^i b_2^j(b_0b_1b_2)$ with $i+j+3=p$. Hence we enforce p–hierarchy and violate symmetry on the triangle. Thus the polynomials are defined for all degrees p, not only till degree 4.

- the modified monomials for the triangle proposed by [Pea76]:

$$\begin{aligned} f_1^0 &= 1 \\ f_{1,2}^1 &= b_1,\ b_2 \quad \text{point functions} \\ f_{1,2,3}^p &= b_0^{p-1}\,b_1,\ b_1^{p-1}\,b_2,\ b_2^{p-1}\,b_0 \quad \text{edge functions}, \quad p>1 \\ f_{i+3}^p &= f_i^{p-3}\,b_0b_1b_2 \quad \text{internal functions}, \quad i \geq 0 \end{aligned}$$

  After construction of the polynomials we substitute the usual linear polynomial b_0 for f_1^0.

- Lagrange polynomials which are interpolation polynomials for the equidistributed interpolation points $x_i = i/p$ and $y_j = j/p$, $i,j \in \{0,1,\ldots,p\}$

on the reference triangle $(x + y) \leq 1$, $(i + j) \leq p$ (for details see [MP72, Nic72]).

$$f_{ij}(x_k, y_l) = \begin{cases} 1 & \text{for } i = k \text{ and } j = l \\ 0 & \text{else} \end{cases}$$

- Bernstein polynomials of degree p [Far90]

$$f_{ijp}(x, y) = \frac{p!}{i!j!(p-i-j)!}\; x^i\; y^j\; (1 - x - y)^{p-i-j}$$

- a construction using Legendre polynomials on the triangle [SB91]: point functions $\{b_0,\ b_1,\ b_2\}$ as usual,

$$F_p = \sqrt{8(2p-1)}\frac{b_0\, b_1}{1-(b_0-b_1)^2}\int_{-1}^{b_0-b_1} f_{p-1}(x)dx$$

  the edge function for the edge $(0, 1)$ together with its cyclic permutations, and internal functions

$$F_{p,q} = b_0 b_1 b_2\; f_p(b_1 - b_0)\; f_q(2b_2 - 1)$$

  with Legendre polynomials f_p defined in example (1.1.3 on page 17).

- the *symmetric hierarchical least squares* polynomials of the polynomial vector space $P_p^{2,\pm}$, optimized consecutively by a least squares method.

- the *symmetric hierarchical step-wise/* consecutively (by a Gauß-Seidel method) optimized polynomials of the polynomial vector space $P_p^{2,\pm}$. Each polynomial generating a symmetrical subset is optimized with respect to the previous optimized polynomials.

- the *symmetric hierarchical* polynomials of the polynomial vector space $P_p^{2,\pm}$, optimized by a Gauß-Seidel method for each space $P_p^{2,\pm}$, $p = 1, 2, \ldots$

We observe the following: For degree one all condition numbers start with a low number. For degree two, the quadratic case, the condition numbers diverge, but not in the same pattern as in the asymptotic case. For non-symmetric triangles we could get up to six different condition numbers depending on the orientation of the triangle, but actually we do get only three different numbers. We see that a suitable orientation pays off in this case. The symmetric polynomials have got only one unique condition number independent of orientation.

We can separate the polynomials into three different groups (figure 46 on page 77). The highest condition numbers are produced by the monomials, the modified monomials, the orthogonal polynomials and the p–hierarchical polynomials. The condition numbers are not acceptable for higher p and grow dramatically. The second group contains the normalized orthogonal polynomials number 2, the modified Legendre polynomials and, asymptotically, the normalized modified Legendre polynomials. The slowest growth of condition numbers show the Bernstein and Lagrange polynomials and the symmetric hierarchical polynomials. The different versions of optimization procedures do not differ much from each other. We have chosen the best optimization as a reference for the following tests. Looking at the details for low degrees we see a cluster until $p = 3$ or 4. An optimal choice in these sections may differ from a choice for higher p. In this range the condition numbers are generally not very high, hence other properties than condition numbers may become a criterion of higher priority.

The comparison of the condition numbers for the Laplace-operator and the pure mass-matrix (i.e. $a_0 \equiv 1$ and $a_{ik} \equiv 0$) on the equilateral triangle shows that the numbers are approximately of the same size but are not equally clustered (figure 47 on page 78). Some polynomials perform significantly better for the mass-matrix and some perform in a similar way. The orthogonal polynomials number 2 do benefit from this operator, whereas the normalized ones (normalized for the Laplace-operator) are slightly better in the beginning until $p = 5$, but are asymptotically worse. The Lagrange polynomials have got smaller condition numbers, too. In this case they generally differ from the numbers of the Bernstein-polynomials. The symmetric hierarchical polynomials perform similarly to the modified Legendre-polynomials. Besides the similarity of condition numbers for the different differential-operators for the equilateral triangle one has to mention the fact that the eigenvalues for the mass-matrix scale in a different way in h than they do for the Laplace-operator.

Let us look at some other triangles. A “canonical” one is the right-angled triangle with short edges of length 1 and the hypotenuse of length $\sqrt{2}$ (figure 50 on page 79). The triangle is no longer symmetric which means that non-symmetric polynomials are sensitive for orientation. We can see this effect for the modified monomials which split into two different paths of condition numbers. We can see that the Bernstein-polynomials perform slightly worse for small p, but in general there is not much difference to the equilateral triangle.

Things change for a distorted right-angled triangle with short edges of length 1 and 16 (figure 50 on page 79). The condition numbers are approximately a factor of 16^2 higher than for the undistorted case. We obtain a splitting of condition number histories not only for the modified monomials (up to 3 branches) but also for the p–hierarchic polynomials and the modified Legendre-polynomials. The difference between different orientations diverges for the Legendre–polynomials starting with $p = 5$, which means that a proper orientation of the polynomials depending on the geometry does pay off. Both for the normalized and the original modified Legendre-polynomials the best condition numbers (lowest branch) do not suffer as much from the distortion as the other polynomials do.

Going further into details we look at two other triangles distorted by a factor of 16 which gives a stretched and a contracted equilateral triangle (figure 51 on page 80). The condition numbers for the stretched triangle do not differ much from the long right-angled triangle. In the contracted case the splitting of the condition numbers for the modified Legendre-polynomials converges for high p, which was not the case for the previous triangles. The normalized Legendre-polynomials perform better than before in a medium range of p.

We can summarize this by saying that the lowest condition numbers are gained with the Lagrange and Bernstein polynomials. They are not far away from the numbers for the new symmetric hierarchical polynomials and the modified Legendre polynomials, which depend on orientation. The exact ranking depends on the triangle and the differential operator itself.

3.1.3 3D Condition Numbers

We compare the condition numbers of the local matrices for the Laplace operator on the tetrahedron. The results only depend on the angles of the tetrahedron.

The polynomials compared are (see chapter 1.1 on page 13):

- the *monomials* $x^i y^j z^k$ with $i + j + k = p \in \mathbb{N}_0$
- the orthogonal *(ortho. 1)* polynomials proposed by [GM78] for the d–simplex defined in chapter (3.1.2 on page 77).
- the orthogonal *(ortho. 2)* polynomials proposed by [AF26] for the d–simplex defined in chapter (3.1.2 on page 77).
- a modified version of the *p–hierarchical* polynomials proposed by [ZT89], which are a generalization of the one- and two-dimensional polynomials.

Figure 52: Local condition numbers for the Laplace operator on the *equilateral tetrahedron* for different polynomials of degree 1 to 7

We take the usual linear functions $\{b_0,\ b_1,\ b_2,\ b_3\}$ and the edge-functions generated by permutations of:

$$f_p = \begin{cases} \frac{1}{p!}\left((b_2 - b_1)^p - (b_2 + b_1)^p\right) & p \text{ even} \\ \frac{1}{p!}\left((b_2 - b_1)^p - (b_2 - b_1)(b_2 + b_1)^{p-1}\right) & p \text{ odd} \end{cases}$$

Analogously to the two dimensional case we enforce p–hierarchy and violate symmetry by constructing internal functions and functions on the triangular faces with standard monomials and "bubble" functions $b_0 \cdot b_1 \cdots b_d$.

- the *modified monomials* originally proposed by [Pea76] for the triangle, generalized for the d–simplex:

$$\begin{aligned} F_0^d &= \{1\} \\ F_1^d &= \{b_1, b_2, \ldots, b_d\} \\ F_p^1 &= \{b_0^p\} \\ F_p^d &= \bigcup_k \text{permutations of } F_{p-k-1}^k \cdot b_0 \cdot b_1 \cdots b_k \end{aligned}$$

Figure 53: Local condition numbers for the Laplace operator on the *right-angled tetrahedron* with short edges of length 1 : 1 : 1 and 1 : 1 : 16 for different polynomials of degree 1 to 7

After the construction of the polynomials we substitute the usual linear polynomial $\{b_0\}$ for F_0^d.

- Lagrange polynomials which are interpolation polynomials for the regular equidistant interpolation points x_α on the reference simplex:

$$f_\beta(x_\alpha) = \begin{cases} 1 & \text{for} \quad \alpha = \beta \\ 0 & \text{for} \quad \alpha \neq \beta \end{cases}$$

- Bernstein polynomials of degree p for the d–simplex [Far90]

$$f_\alpha = \binom{|\alpha|}{\alpha} b^\alpha$$

- a construction using Legendre polynomials on the tetrahedron [SB91]: point functions $\{b_0,\ b_1,\ b_2,\ b_3\}$ defined as usual,

$$F_p = \sqrt{8(2p-1)} \frac{b_0\, b_1}{1-(b_0-b_1)^2} \int_{-1}^{b_0-b_1} f_{p-1}(x)dx$$

Figure 54: Local condition numbers for the Laplace operator on the *right-angled tetrahedron* with short edges of length $1:16:16$ and $1:16:16^2$ for different polynomials of degree 1 to 7

as edge function for the edge $(0,1)$ together with permutations for the other edges,

$$F_{p,q} = b_0 b_1 b_2 \; f_p(b_1 - b_0) \; f_q(2b_2 - 1)$$

as triangle function for the triangular face $(0,1,2)$ together with permutations for the other faces,

$$F_\alpha = b_0 \cdot b_1 \cdots b_d \; f_{\alpha_1}(b_1 - b_0) \; f_{\alpha_2}(2b_2 - 1) \cdot f_{\alpha_3}(2b_3 - 1) \cdots$$

as general internal functions for a d–simplex with $|\alpha| = d$ and with Legendre polynomials f_p defined in example (1.1.3 on page 17).

- the *symmetric hierarchical least squares* polynomials of the polynomial vector space $P_p^{3,\pm}$, optimized by a step-wise least squares method.
- the *symmetric hierarchical step-wise*/ consecutively (by a Gauß-Seidel method) optimized polynomials of the polynomial vector space $P_p^{3,\pm}$. Each polynomial which generates a symmetrical subset is optimized with respect to the previous optimized polynomials.

Figure 55: Local condition numbers for the Laplace operator on the *distorted equilateral tetrahedron* stretched and contracted by a factor of 16 for different polynomials of degree 1 to 7

- the *symmetric hierarchical* polynomials of the polynomial vector space $P_p^{3,\pm}$, optimized by a Gauß-Seidel method for each space $P_p^{3,\pm}$, $p = 1, 2, \ldots$.

We observe the following: For degree one all condition numbers start with a low number. For degree two, the quadratic case, the condition numbers diverge, but not with the same pattern as in the asymptotic case. For non-symmetric tetrahedra we could get up to 24 different condition numbers depending on the orientation of the tetrahedron, but actually we get up to six different condition numbers. We see that a suitable orientation pays off in this case. The symmetric polynomials have got only one unique condition number independent from orientation.

In more detail we can see in figure (52 on page 84) an analogous pattern of condition numbers for the equilateral tetrahedron. We can split the polynomials into groups of lower or higher growth of condition numbers. The monomials, the orthogonal polynomials, the p–hierarchical polynomials, the modified monomials and the modified Legendre polynomials belong to the group with rapidly growing condition numbers. On the triangle the modified Legendre-polynomials were in the group with better condition numbers, but this is not the case for the tetrahedron. Now the Bernstein polynomials perform best and the Lagrange polynomials are slightly worse. In general the lowest condition numbers are of the same magnitude like on the triangle, but other ones may be much higher. We have to keep in mind that the number of polynomials involved for each p grows an order of p faster in 3D than in 2D.

Looking at right-angled tetrahedra we have to consider different situations concerning a factor 16 of distortion (figures 53 on page 85 and 54 on page 86). The distortions are applied analogously to the 2-D case (figure 48 on page 78). We take the edges $1 : 1 : 1$, $1 : 1 : 16$, $1 : 16 : 16$ and $1 : 16 : 16^2$ preserving the right-angle. The $1 : 1 : 1$ situation produces only slightly higher condition numbers than the equilateral tetrahedron. The other distorted tetrahedra have a factor of 16^2 and the twice distorted ones $(1 : 16 : 16^2)$ an even higher factor. The condition numbers for the $1 : 16 : 16$ case are slightly better than for the $1 : 1 : 16$ one. In each case the symmetric hierarchical polynomials have the lowest condition number of all hierarchic polynomials. Only Bernstein- and Lagrange-polynomials have lower ones, but are not far away. The different distortions lead to a splitting of the condition number histories, which diverge for modified monomials and modified Legendre-polynomials. For high p a proper orientation pays off for these polynomials.

Proceeding to distortions of an equilateral tetrahedron (figure 55 on the page before), we can observe a similar pattern but with different scaling. We have contracted and stretched a tetrahedron by a factor 16. The contracted one leads to a smaller divergence of non-symmetric polynomials and at some points to a slightly lower condition number. The general behavior and the division into groups remains the same. At degree $p = 3$ we can see small deviations for some polynomials.

We can summarize the results saying that the lowest condition numbers arise in the case of Bernstein and Lagrange polynomials. They are not far away from the numbers for the new symmetric hierarchical polynomials. The modified Legendre polynomials have got higher condition numbers, which additionally depend on orientation. The exact ranking depends on the tetrahedron shape and the differential operator itself.

3.2 Comparison of h–, p– and h–p–Versions

We present some numerical experiments concerning the performance of different finite element versions. Depicted are the error measured in energy norm versus the number of unknowns in the linear system of equations (logarithmic scale). The examples shall confirm that the uniform p–version is faster than the uniform h–version, different behavior of the adaptive versions and performance of the h–p adaptation for some threshold parameters. The adaptivity is controlled by a posteriori error estimators and error indicators described in chapter 2.1 on page 53. For a detailed explanation of such kind of figures we refer to [SB91].

For Poisson equation

$$-\Delta u = f, \text{ in } \Omega, \qquad \Omega \subset \mathbb{R}^d$$

with smooth right hand side

$$f \in C^\infty(\Omega)$$

we obtain smooth solutions in the interior of the domain Ω and several kinds of singularities at the boundary $\partial\Omega$, sometimes referred as pollution effect. In one dimension $d = 1$ there are no such singularities. In two dimensions all singularities are vertex singularities of the type

$$u = \Psi(\phi) r^\alpha$$

written in polar coordinates (r, ϕ) with a smooth function Ψ. α determines the regularity of the solution u and depends on the angle ω of the vertex and the kind of boundary conditions, $\alpha = \pi/\omega$ or $\alpha = \pi/2\omega$ (D-D and N-D).

In three dimensions $d = 3$ there are three different types of singularities, called edge, vertex and edge-vertex singularities. Following [Gri92, Dau88, BS94] and more general [NP94], we denote the asymptotic expansions like this:

Edge singularities are genuine 2D vertex singularities, written in cylinder coordinates (r, ϕ, x) with smooth c

$$u = c(x)\Psi(\phi) r^\alpha$$

Vertex singularities are real 3D singularities, in polar coordinates (r, θ, ϕ) with smooth c

$$u = c(\theta, \phi) r^\beta$$

and exponent β depending on the lowest eigenvalue of the Laplace-Beltrami operator on a infinitesimal sphere around the vertex (in Ω).

Edge-Vertex singularities denote a mixture of pure edge and vertex singularities, which often is the case

$$u = \Psi(\phi) r^\beta \theta^\alpha$$

There is another kind of pollution driven by some non-smoothness of the right hand side f or jumps in the coefficients of the differential operator, e.g. change of material. These discontinuities often are resolved by the initial grid. An adaptation scheme suitable for this singularities mainly is a proper adaptive integrator.

3.2.1 Polynomial Solution

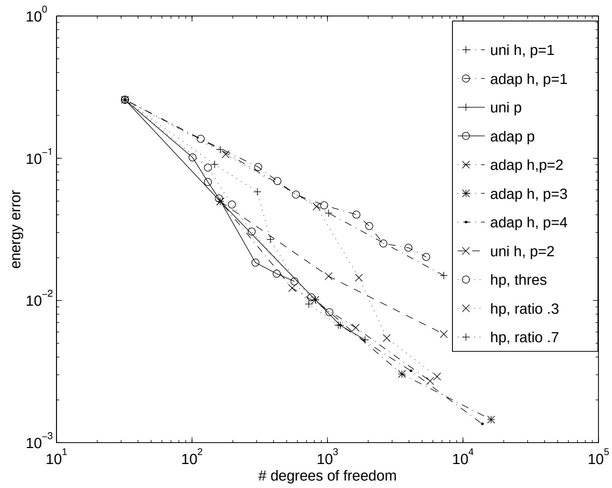

Figure 56: Nearly polynomial solution

First example is a problem with a nearly quadratic solution,

$$\Delta u(x) = 1, \quad x \in [-1, 1]^3$$

with homogeneous Dirichlet boundary conditions.

This example seems to be a little bit unfair for linear element $p = 1$ h–version FEM. For exact polynomial solution an higher order FEM is of course able to reproduce this solution exactly. Distorting the quadratic polynomial by the introduction of vertices of the cube, higher order FEM inherits an advantage, but is not exact any longer.

The convergence results are depicted in figure 56, with a computational domain of only 1/8th of the cube. The energy error of the discretization versus the number of unknowns are given, with neglect-able error of a linear algebra solution procedure. It shows a comparison of uniform and adaptive h–versions with different orders p, uniform and adaptive p–versions on the initial grid and some h–p–version runs. For linear elements $p = 1$ adaptivity does not pay off. We see the rare case of uniform h–version being superior over adaptive versions at some stages. In general the h–adaptive control is very robust, but of course it is easy to trap an adaptive procedure by bad error indicators, even delivering no convergence at all. Here the error estimator has got good equivalence constants, preventing this worst case.

Things completely change for quadratic elements. Now the adaptive version gains an extraordinary factor of at least 4 (number of unknowns) compared to the uniform h–version. General approximation by quadratic elements is good and we have to resolve the vertex and to a certain extend the edges with the aid of adaptivity. We want to point out, that this is no effect of error estimation, but a question of approximation, which can be verified looking at the other examples.

When we now have a look on the bulk of other tests, we do not expect great differences between them. The adaptive p–version seems to be preferable at some stages, but uniform p, adaptive h with $p = 2$ or $p = 3$ do not deteriorate much. The h–p–tests are among them, after some initialization phases. The ratio of .7 means more p–refinement (and less h–refinement) than the ratio of .3, which is faster approaching the p–version's convergence history in this example.

3.2.2 Analytic Solution

Next example is an analytic one, also rewarding higher order approximations:

$$\Delta u(x) = \cos x_1 \cos x_2 \cos x_3, \qquad x \in [-\pi/2, \pi/2]^3$$

with homogeneous Dirichlet boundary conditions.

This example illustrates the behavior of smooth solution in the interior of a domain and an analytic right hand side. Even a solution with singularities at the boundary may be thought of a linear combination of such a smooth solution and some approximations of the singularities. Hence such an example isolates the performance of the method for the interior of the domain.

There is another challenge of the problem, which is connected with higher order derivatives. The solution is an even function. Hence even and odd derivatives behave differently at each point in the domain. This exaggerates any oscillations of an error estimator during p–refinement, because the higher

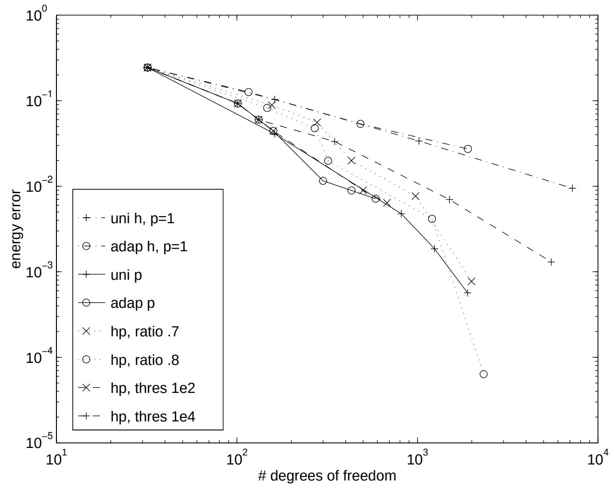

Figure 57: Analytic Solution

order approximation $p \to p+1$ need not improve the solution. This also challenges the p–adaptive control and the h–p–version control.

The results are depicted in figure 57, calculations done on a domain of 1/8th with different scaling in x. The h–adaptivity does not pay off in this example. Conversely the moderate increase of number of unknowns requires a higher number of matrix assemblies and solution steps than uniform (factor 8) refinement. For p–version we slightly can see an exponential convergence with the adaptive version oscillating around the smoother convergence history of the uniform p–version. The threshold h–p–versions perform between pure p–version and h–version, depending on the threshold parameter (parameter times maximum). The fixed thresholds (without using maxima) need some startup, but do outperform the p–version on the long run. The precisions reached in this example are extraordinary high for 3D calculations due to the smoothness.

3.2.3 Edge Singularity

We consider the edge singularity (in cylinder coordinates) of the type

$$u = \sin(\alpha\phi)r^{\alpha}, \qquad \alpha = 2/7$$

Figure 58: Edge singularity: geometry with boundary conditions and perspective view of the solution (h version)

analogous the 2D singularity example 'circle' of [Ban94] and [Lei90]. We use homogeneous Neumann and varying Dirichlet boundary conditions and a right hand side $f = 0$. The problem serves as an example for one kind of singularities caused by the geometry of the domain and by boundary conditions. Figure 58 shows geometry and boundary conditions and the solution. Although the solution would not be H^2–regular in 2D ($\mathrm{H}^{1\frac{2}{7}+\epsilon}(\Omega)$), it is almost H^2–regular in 3D ($\mathrm{H}^{25/14+\epsilon}(\Omega)$). Hence it is not a very strong singularity and performance of h–version FEM of order $p = 1$ should not be seriously deteriorated by the singular behavior.

This is why the adaptive h–version as shown in figure 59 on the following page only gains a constant factor less than 2 compared to the uniform h version. This too gives rise to the good performance of the p version although adaptive p version control is oscillating a little bit. The p–versions remain the best ones until the maximal degree p is reached. Beyond an error of 2% the adaptive h–version with quadratic elements seems to be preferable. The cubic elements probably will be superior for even higher precision demands. The h–p–version show a whole spectrum of convergence behavior: One threshold version near the p–version, one near the h–versions and an-

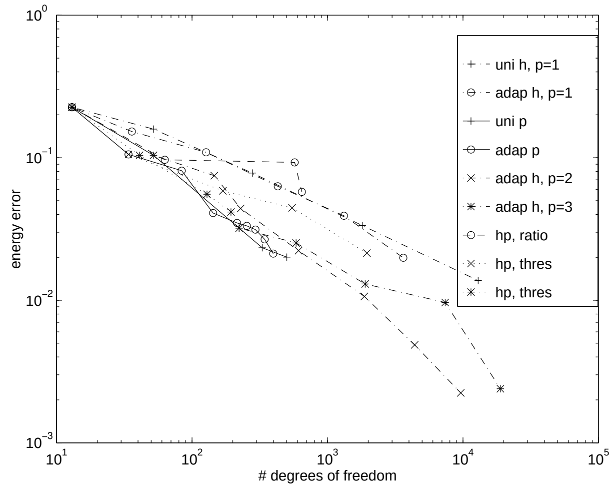

Figure 59: Edge Singularity

other version which has a drop out, refining all h at 10% error, leading to an disastrous amount of work before recovering.

There is another general drawback in performance of the adaptive h–version here, which is isotropic refinement. Some kind of anisotropy as proposed by [BS94] would certainly be more suitable for the structure of the singularity than the isotropic one. While solution changes rapidly perpendicular to the edge, it is smooth along the edge.

There are such anisotropic refinements in computational fluid dynamics, even for triangular grids in 2D ('blue refinement' proposed by [KR90]. However there are certain implementational difficulties, performing this in the context of an (self-) adaptive procedure and only based on local information. First there is the decision to be made whether to refine in one direction or isotropically in both directions. For means of efficiency, elements have to be rotated and distorted. Last un-refinement has to be considered, because one wrong isotropic / anisotropic decision made in the first steps of the algorithm may deteriorate the overall performance of the adaptive procedure.

These difficulties may explain, why we did not consider anisotropy in 3D, although the shape functions do facilitate anisotropic distribution of degrees of freedom, too. But the possible gain of a reduction of degrees of freedom from $n = h^{-3}$ down to $n = h^{-2}$ seems to be neglect-able in view of h–p

version asymptotic convergence. With respect to the regularity of many 3D problems and the good performance of higher order FEM versions, this decision seems to be justified.

3.2.4 Edge-Vertex Singularity

Figure 60: Edge-vertex singularity, geometry with boundary conditions and perspective view of the solution

The edge-vertex singularity problem is constructed by means of homogeneous Neumann and varying Dirichlet boundary conditions, right hand side $f = 0$ analogous the edge-singularity example in the previous section 3.2.3. Figure 60 shows the geometry and the boundary conditions for the edge-vertex singularity problem.

The appropriate solution u is depicted in figures 61 on the next page. The edge-vertex singularity is not a very strong one as explained in section 3.2.3 on page 92. The results seem to be comparable at a first glance.

We obtain a factor of 2 improved convergence by using adaptive h–version instead of uniform h–version. Up to an error of 3% the adaptive p–version performs best, when the maximal polynomial degree is reached. Uniform p–version is slightly less efficient, about a factor of 1.3. For a higher precision

Figure 61: Edge-vertex singularity, cross-section of the solution

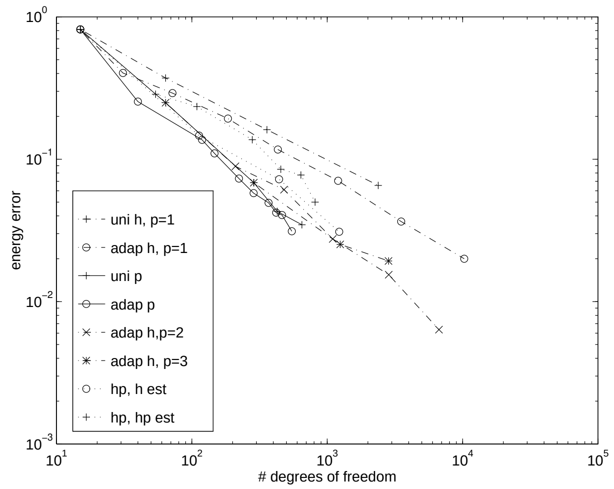

Figure 62: Edge-Vertex Singularity

beyond the 1% mark, quadratic adaptive h–version seems to be the method of choice. Cubic elements are to expensive below a precision of 2%. The different h–p–versions mimic the p–version's behavior in a succesfull way. The version using h and p estimates uses slightly to much h–refinement in the startup phase.

The overall good performance of the equal p–version and higher order h–version are generally due to regularity of the solution. Any effect of the use of anisotropic refinement probably would at least be disturbed by the additional presence of vertex-singularity components in the solution and in view of the regularity without great effect.

3.2.5 Vertex Singularity

Figure 63: Vertex singularity, perspective view of the solution and close up

Unfortunately the construction of pure vertex singularities with zero right hand side $f = 0$ via boundary conditions is difficult in 3D, so we present a vertex singularity of r^{α}–type induced by the right hand side (in spherical coordinates)

$$f = r^{\alpha-2}, \quad \alpha = -.3$$

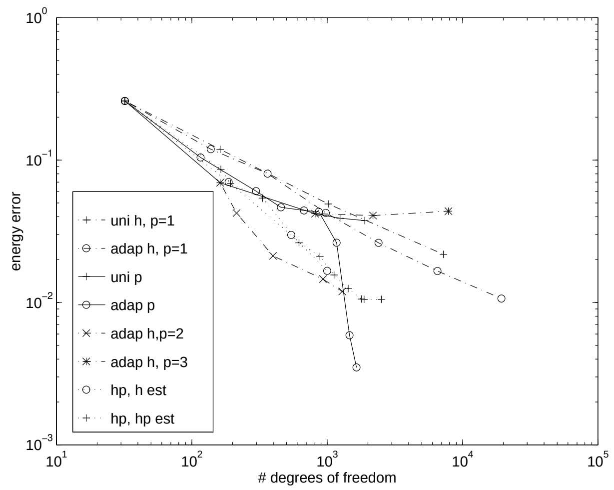

Figure 64: Vertex Singularity for another (more regular solution) exponent $\alpha = .3$

with homogeneous Dirichlet boundary conditions on the unit cube with a computational domain of 1/8th. The right hand side is not analytic any longer. Nevertheless this does not destroy the exponential convergence of a p– or h–p–version, because of the spherical r^α type solution.

First we look at the vertex singularity of exponent $\alpha = .3$, which is not a very strong one and belongs to $\mathrm{H}^{1.8+\epsilon}(\Omega)$ as explained in chapter 3.2.3 on page 92. The results depicted in figure 64 seem to be comparable to the previous regular solution ones at a first glance.

Due to regularity, the adaptive h–versions is able to gain a factor of only 2.5 compared to uniform h–version. However uniform p–version performs poor below an error of 5%. Performance degrades in this asymptotic phase worse than uniform h–version. Such is cubic element adaptive h–version, with a rare effect of increasing error while refinement. This increase is not possible for nested FEM spaces, only stagnation would be possible. The effect is do to green closure of triangulations, which introduces a small effect of non nested-ness of spaces in some distance away from the singularity. While quadratic adaptive h–version seems to be preferable up to an error of 2%, it is overtaken by the adaptive p–version. After a startup phase the adaptive p–version reaches the pre-asymptotic exponential convergence phase down to .4%. This is in sharp contrast to uniform p and in fact a rare case of a drastic improvement by p–adaptivity. One has to mention that the improved approximation at the last adaptive p–version steps in this case is also visible at the increase of iterations during solution. The two h–p–versions perform in a smoother fashion than the superior quadratic h–version in the 1% error area. Different refinement control has got no great effect onto convergence.

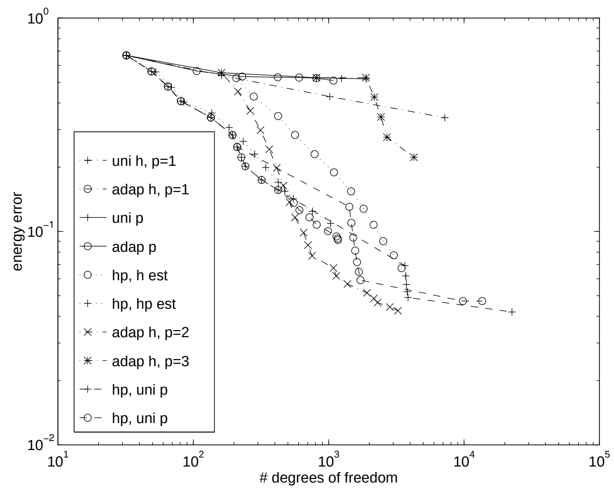

Figure 65: Vertex singularity for (original) exponent $\alpha = -.3$

We now consider the other exponent example. The vertex singularity with parameter $\alpha = .3$ are almost $\mathrm{H}^2(\Omega)$–regular ($\mathrm{H}^{1.8+\epsilon}$), whereas exponent $\alpha = -.3$ gives rise to a strong singularity only contained in $\mathrm{H}^{1.2+\epsilon}(\Omega)$, shown in figure 65. This completely changes the role of adaptive h–version. Adaptivity now has to restore the usual $\mathrm{H}^2(\Omega)$ convergence properties of the h–version (with respect to the amount of work), which is $\mathcal{O}(1/\sqrt[3]{n})$ and now is not only a gain of a constant factor any longer. The history of grids during refinement becomes an arithmetic progression of nodes in a pre-asymptotic phase instead of geometric progression for regular solutions. Now any $\mathcal{O}(n)$ implementation instead of $\mathcal{O}(\#\text{levels})$ in the FEM code pays off in this phase of convergence.

Figure 65 depicts the convergence for exponent $\alpha = -.3$. We do not obtain the regular convergence slope of $.5 : 2^d$ for uniform h–version, but can clearly see the degradation down to approximately $.5^{\cdot 2} : 2^d \approx .87 : 8$. Hence uniform h–version becomes unacceptably slow. We also observe the feigned convergence of the p–versions suffering from pollution of the singularity.The p–versions do not only fail to converge at an observable speed, but they fake convergence in conjunction with a posteriori error estimators. For this example and this very coarse initial grid, p–version is by no means reliable. This is in contrast to the overall good performance in other examples and illustrates the potential danger of the method.

The different h–p–versions are in certain trouble, too, because they should not raise p in the first steps and perform h–refinements only very selectively. We can see the $p = 2$ adaptive h–version overtaking the $p = 1$ version at about .15 error and reaching an asymptotics at .07 error with overall best performance. The $p = 3$ h–version seems to be much to costly in this phase. Hence any h–p version avoiding to much p–refinements is comparable to the $p = 1$ h–version and will be overtaken by the $p = 2$ version at this point of .15, while raising p at this point (like for the uniform p h–p–versions) is much to costly and introduces un-smooth convergence histories, because the whole grid is not adapted for higher p and h–de-refinements at additional cost seem to be appropriate but we would expect to be unmanageable to control.

We can draw two conclusion out of this: In the vicinity of strong singularities p–version is extremely dangerous while different adaptive h–versions perform well. And additionally any h–p control of comparable quality in the presence of strong singularities can not be a pure local strategy based on local error estimates as proposed by the Kaskade principle (look for the uniform p h–p–versions). Nevertheless higher order FEM pays off even in this case, for precision better than 15%.

In general we have seen the potential danger of pure p–versions for singularities and a comparable performance of higher order adaptive h–versions overcoming this danger. The optimal polynomial degree depends on the final precision required. In our examples we obtained an optimal switch form linear to quadratic elements at an error level of about 10%. For extraordinary high precisions, solution dependent methods like adaptive p–versions or h–p–versions have to be applied.

3.3 Application to Structural Mechanics

Before presenting some examples, we briefly want to summarize some basic equations of structural mechanics.

3.3.1 Foundations of Structural Mechanics

We want to derive the basic equations from elasto mechanics in brief, using notation of [Bra92, Ran78], for further reading see [Cia88, Gur81]. We consider a bounded mechanical body at an un-deformed position, called reference configuration, $\Omega \subset \mathbb{R}^3$. It is subject to forces and is thereby deformed to a new position. The transformation

$$x \to x + u(x) = (u + id)(x) \in \mathbb{R}^3$$

describes the position of every point of the body $x \in \Omega$ after deformation. The field

$$u : \Omega \to \mathbb{R}^3$$

is called the displacement field. The symmetric tensor

$$C = \nabla(u + id)^t \nabla(u + id)$$

is called the right Cauchy-Green strain tensor related to the local change of scales. The (symmetric) strain tensor itself is defined by

$$E = \frac{1}{2}(C - Id)$$

sometimes also called 'kinematic balance', yielding

$$E_{ij} = \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}) + \frac{1}{2}\sum_{k=1}^{3} \frac{\partial u_i}{\partial x_k}\frac{\partial u_j}{\partial x_k}$$

Looking at the eigenvalues and eigenvectors of E gives rise to the main directions of strain. One can also extract geometric shears.

Stress is defined as force per area, introduced by Cauchy (1823) stating the existence of a symmetric tensor T, which fulfills the static balance

$$\mathrm{div} T(x) + f(x) = 0$$

with $f(x)$ being forces (dead loads) applied to the body. Unfortunately the strain tensor T is given in the coordinate system of the deformed body. Transforming the Cauchy stress tensor back to the reference system by Piola transform leads to an un-symmetric first Kirchhoff stress tensor. The symmetric version is called (second) Kirchhoff stress tensor defined by

$$\Sigma = \det(\nabla(u + id))(\nabla(u + id))^{-1} T(\nabla(u + id))^{-t}$$

For completeness we include the definition of the term 'constitutive equation' describing properties of the material:

$$\Sigma = \Sigma(x, E(x), \ldots)$$

A constitutive equation has to be 'objective', which means independent of the coordinate system, it may be homogeneous (independent of position x) or isotropic (independent of orientation). A homogeneous material is called 'elastic', iff the stress tensor does only depend on the strain

$$\Sigma(x) = \Sigma(E(x))$$

Applying proper boundary conditions, including prescribed displacements or forces onto the boundary, the problem description is completed.

3.3.2 The Linear Elastic Problem

We want to simplify the situation a little bit, looking at the linearized elastic problem. We have to make the assumption of small strains, giving rise to the linear variant of the Cauchy-Green strain E

$$\epsilon_{ij} = \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i})$$

We are able to identify the stress tensors T and Σ writing σ in the latter. Hence the constitutive equation becomes linear. If the material is isotropic, there are only two parameters left (as a consequence of a theorem of Rivlin and Erickson).

$$\sigma = \frac{e}{1+\nu}(\epsilon + \frac{\nu}{1-2\nu}(\mathrm{tr}\ \epsilon)Id)$$

is called St. Venant-Kirchhoff equation or just generalized Hookean law, with constants e Young's modulus of elasticity (in units of pressure) and ν Poisson ratio, $0 \leq \nu \leq 1/2$. It is sometimes written in Lamé constants

$$\lambda = \frac{e\nu}{(1+\nu)(1-2\nu)} \quad \mu = \frac{e}{2+2\nu}$$

The differential equation of the static balance in the variational formulation leads to the energy functional

$$\Pi = \int_\Omega (\frac{1}{2}\epsilon : \sigma - fu)dx$$

The interior forces f may stem from gravity or in one application from thermal expansion. Introducing a temperature τ and a reference system of the body at a reference temperature τ_0, we substitute the strain ϵ by $\epsilon - \epsilon_0$, with thermal strain being

$$\epsilon_0 = \alpha(\tau - \tau_0)Id$$

with a thermal expansion coefficient α [ZT89]. One is able to rewrite the equations, inserting the temperature into the right hand side force $f = \alpha\nabla\tau$. For further reading see [BW60]. The equations are called Navier-Lamé.

We are able to exploit the variational formulation for finite element discretization, called the displacement approach. We discretize the displacements u by $\mathrm{H}^1(\Omega)$ finite elements leading to a second order elliptic partial differential equation in 3D with a system of three unknowns per node. Ellipticity is guaranteed by Korn's inequality, if the Poisson ratio $\nu < 1/2$. In the vicinity of $\nu = 1/2$ we obtain numerical effects of 'locking', the finite element approximation constants become worse, which is not as dramatic for p–version finite elements [Vog83].

3.3.3 Discretizations

Another way to overcome locking is the use of mixed finite element methods, discretizing the stress σ, too. This leads to saddle point problems not considered here. We just want to mention some approaches like the two-field mixed discretization $(u - \sigma)$ named Hellinger-Reissner and the three-field mixed discretization $(u - \epsilon - \sigma)$ named Hu-Washizu, see [ZT91, RT91].

Most of engineering finite element computations are not done with full 3D models, but with less computational expensive 2D ones. Under the assumption of small thickness of a body or homogeneity in that direction, there are reduced models of plain stress and plain strain. Using some higher order differential terms, previously neglected, there are masses of different models for plate bending and shells, depending on what parts of the equations are considered to be small and are neglect-able and depending on some given symmetry. In plate bending we want to mention models of Kirchhoff-Love and Reissner-Mindlin. There are also some approaches of varying approximations for plates and shells in the direction of thickness. In 1D there are analogs named beams and rods [ZT91].

3.3.4 Nonlinear Mechanics

In the derivation of the linear equations we made some derivations about small displacements and small strains. These need not be fulfilled. We are now able to enhance the system of equations by several kinds of non-linearity.

First one is called the geometric non-linearity. We have to consider the second order derivatives of displacement:

$$E_{ij} = \epsilon_{ij} + \frac{1}{2}\sum_{k=1}^{3} \frac{\partial u_i}{\partial x_k}\frac{\partial u_j}{\partial x_k}$$

Taking into account the different stresses T and Σ we have a fourth order partial differential equation, which under some circumstances loses uniqueness of the solution. There are effects of symmetry breaking (e.g. Eulerian rod) and history dependence of the solution. Non-unique solutions are often calculated by means of path following methods, see [Le 94, Cri91] and citations therein and [Hoh94]. These effects generally appear in bending, not only in 3D but for plates and shells in 2D and rods in 1D.

There is another effect of geometric non-linearity, for large deformations not contained in the differential equations. The deformation of the body may be restricted by some other bodies, leading to the problem of contact. Besides modelling of actual contact (friction, slip, adhesive) (see [KO88] and references therein) there come some inequalities into play written as a constraint $\tilde{\Omega}$

$$(u + id) : \Omega \to \tilde{\Omega}, \quad \text{injective}$$

It is not only a question of numerical algorithms here [HHNL88] and [Hop90, HK94], but sometimes is an algorithmic challenge, too, identifying the locations of contact.

If stresses become large, even the elliptic constitutive equations for a specific material may become complicated. The number of parameters increases. But modelling a material as being elliptic need not be appropriate any longer. There are two generalizations: Viscosity means a certain flow of the material which may occur just before melting or under extreme conditions. A constitutive equation may be written as

$$\Sigma \in \partial\phi(\dot{E})$$

with a dissipation potential ϕ. This would be fluid dynamics, but with transitions from elastic response to viscous flow we are at visco-elasticity. The problems are time dependent, see also [Le 90, FLO76]. Pure melting with phase transitions of a material is a special subject where numerical detection of the transition area can be studied in detail [HK90].

Another material property is plasticity. Material has got memory and behavior depends on the history of time at that point [DL76, FLO76]. Perfect plasticity may be written as

$$\begin{aligned} \Sigma &\in C \\ (E - E_{\text{linear}}(\Sigma)) &\in \{\text{normal cone to } Y \text{ at } \Sigma\} \end{aligned}$$

with local linear elasticity tensor E_{linear} denoting the elastic constitutive equation for E and Σ. Sometimes a combination of all of them is used, called elasto-visco-plasticity, where the yield surface Y is time dependent. These problems definitely are time dependent. Integrating one time-step, one has to solve variational inequalities giving rise to rather complex algorithms and often prohibiting the use of advanced strategies of ordinary differential equation solvers.

3.3.5 Experiments

We present some experiments in linear elastic mechanics with a full 3D displacement discretization. For an analogous treatment of singularities for elasto mechanics like in chapter 3.2 on page 88, the reader is referred to Grisvard [Gri89, Gri92].

3.3.6 Attachment Lug

This example is a typical 2D benchmark test, so it seems to be prohibitive for 3D calculations. But we are able to test symmetry of the solution while initial triangulation is not symmetric, and we are able to test loading (Neumann) boundary conditions. The solution has got singularities at the boundaries of the prescribed displacement area (change homogeneous Dirichlet/ Neumann conditions). Only constant loads on triangle faces are applied due to the lack of proper geometric handling of the curved surface (lack of exporting curved surfaces due to I-DEAS). Hence there are some minor singularities in the loaded area, which do not influence convergence very strongly depicted in figure 68 on page 107.

The adaptive h–version is a factor of 2.5 faster than the uniform h–version. This asymptotic behavior is reached in the very first steps. Best performance delivers adaptive p–version up to a precision of .3%. Uniform p–version seems to be a constant factor slower before performing two exponential convergent final steps below .1%. The higher order adaptive h–versions are comparable down to a precision of .5% before reaching an asymptotic phase. It is interesting to mention that cubic h–version performs better than quadratic in this example, while usually it appears to be too costly. The h–p–versions in the first steps do too much h–refinement before recovering down to the

Figure 66: Attachment lug: geometry and boundary conditions, and p–version solution (displacement)

general .5% mark. We conclude that we obtain a very regular solution's behavior facilitating a high final precision and smooth p–convergence properties.

3.3.7 Molding Case

We apply this as an example for complicated while plane geometry. Walls are considered thick, so a real 3D approach is needed. One triangle shaped area on top of the case is loaded by a constant force while zero displacement is prescribed at the base of the case. This may reflect some test of durability of the case. Singularities mainly are placed at the edges of this triangle. The other singularities are at obtuse angle corners and slightly above the prescribed zero deformation at ground level.

Results are shown in figure 71 on page 109 which is a relatively regular solution's behavior. The overall precision is limited by a larger initial grid due to a more sophisticated geometry, compared to the previous example. Adaptive h–version delivers a .5 advantage over uniform h–version. There are too much and too weak singularities. Up to a precision of 8% the

Figure 67: Attachment lug: strain and displacement (left) and stress and displacement(right)

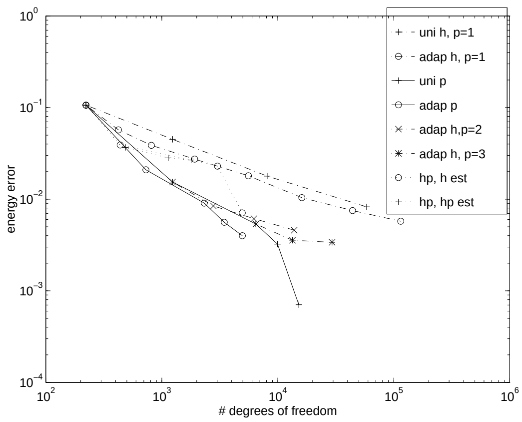

Figure 68: Attachment Lug

Figure 69: Molding case: perspective view of the geometry and boundary conditions

quadratic adaptive h–version performs best, overtaken by cubic adaptive h–version. Both sharply end up in an asymptotic convergence phase. The pure p–versions both perform very well, adaptive p down to 5% and uniform below. Uniform p–version even reaches an exponential convergence phase ending up at the most precise solution about 1.5%. The h–p–versions both perform well, independent form exact control. They should have done some more p–refinement, but converge in a smooth way. To conclude we obtain a very regular behavior with exponential p–version convergence and finial precision guided by geometric complexity of the domain.

3.3.8 Valve Body

This example is considered as a relatively complicated geometry. Interior forces are used, to squeeze the thick pipes of the valve body. One side contains an area with fixed displacement (an attachment) giving rise to some singularities at the boundary of the attachment area.

Convergence history is depicted in figure 75 on page 112, where usual performance for regular solution can be obtained. In detail, adaptive h–version is about a factor of 2 faster than uniform h–version, both delivering usual

Figure 70: Molding case: perspective view of stress and displacement and cross-section

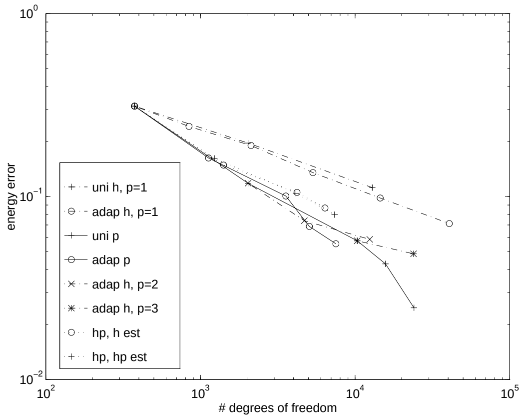

Figure 71: Molding Case

Figure 72: Valve body: perspective view of the geometry and boundary conditions

regular convergence. Adaptive p–version performs best, which is slightly better than uniform p–version. The higher order adaptive h–version are comparable down to a precision of 5% and continue with slower asymptotic convergence. In this case, the cubic version performs better than the quadratic one, which seems to be natural,but look for the singularity examples chapter 3.2.3 on page 92 ff. The h–p–version's performance approaches the p–versions if one increases the number of refined elements at every refinement step. Low numbers of 10% elements marked for refinement are far to specific and tend to use to much high degrees p. We presented 30% refinement for threshold and fixed h–p criterion based and h– and p– estimates and a 60% run, which is difficult to separate form pure p–version although h-refinements were applied.

We conclude for all three elasto mechanics examples a very regular convergence, although there are some singularities present. Final precision is strongly limited by geometric complexity of the domains, while the number of unknowns on the very coarse initial grid seem to be comparable. Hence pure p–versions pay off in this mechanics examples, while higher order h–versions seemed to be preferable for the Poisson equation prototype

Figure 73: Valve body: perspective view of strain and displacement and cross-section

singularities.

Figure 74: Valve body: diagonal cross-section, strain and displacement

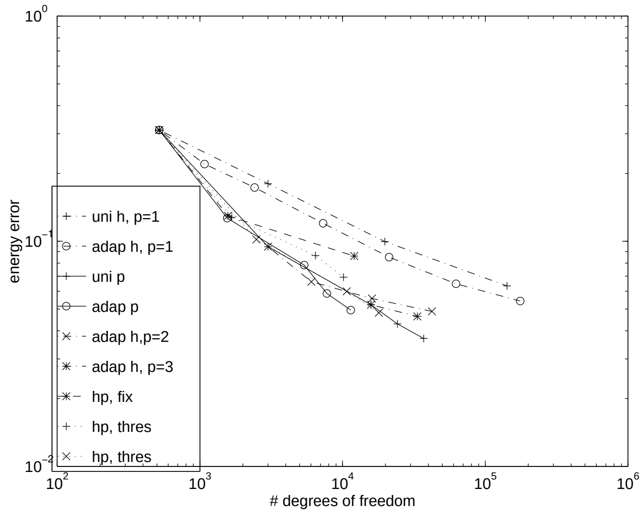

Figure 75: Valve Body

Conclusion

We have presented a complete framework for adaptive h-p finite element methods for second order boundary value problems. Aiming an efficient computational code with fully automatic control, we were led to the h-p version of finite elements ensuring (sub-) exponential convergence in contrast to the standard algebraic one. To generate the full convergence order, well-adapted grids had to be generated by the code.

The demands for efficiency in conjunction with unstructured grids (because of geometry constraints and adaptation), varying polynomial degrees (in space and in adaptation history) and some concerns on robustness required new shape functions. The different polynomial degrees call for the concept of p-hierarchy of the shape functions. The easy assembly of the global stiffness matrix and the load vector on unstructured grids of simplices clearly lead to the requirements of symmetry of the shape functions on the boundary of each individual element. Finally independence of orientation demands symmetry of the shape functions on the whole element.

A short review showed some symmetric and p-hierarchic families of shape functions. But it was proved, that no one could have both properties at once for standard polynomial spaces. Hence the spaces of polynomials spanned by the shape functions were slightly modified and p-hierarchic and symmetric shape functions were constructed.

Therefore, a domain decomposition preconditioner for h-p grids based on a standard multilevel iterative solver for h grids was developed. This construction implied an orthogonalization of shape functions by means of optimization of the resulting condition numbers of the preconditioner. The optimization procedure delivered the uniqueness (modulo symmetry) of the shape functions.

A set of error estimators and refinement strategies was developed for an adaptive construction of h-p grids where h- and p-refinements were applied simultaneously. The most robust strategies turned out to be the ones with different a posteriori error estimators on the finest grid.

The numerical experiments demonstrated the superior convergence properties of pre-asymptotic p-version and the global convergence of h-p-version finite elements which agrees with the theory. This was shown both for the characteristic 3D singularities of the Laplacian and for some 3D examples of linear elasto mechanics. Medium or high precision solutions (less than 5%-20%) for 3D problems could only be obtained by higher order methods. Comparisons with standard h-version implemented in Kaskade were available only for low precision on the very first grids and refinement steps.

However we also have to state the enormous efforts in implementation needed for h-p adaptive codes.

The advances in parallelization of adaptive finite element codes appeared to be slow and there was no complete implementation found. Furthermore complexity bounds were derived for the partitioning of adapted grids (the bottle-neck). Hence there may be only convincing solutions for massive parallelization which depend on the problem and the computer hardware. The computer graphics capabilities for higher order elements were reviewed, stating that appropriate base software is available which only lacks the usage by finite element packages.

In a short introduction to structural mechanics we have shown some future directions of development for the existing linear finite element code, which are different kinds of non-linearities, variational inequalities for primal or dual variables, mixed finite elements for nearly incompressible materials, reduced elements in 2D or in 1D following different theories of mechanics (or dimension adaptation), time dependent problems and path following for (geometric) instabilities (bifurcations). Some of the generalizations were done within the KASKADE project.

We conclude that further finite element projects dealing with piecewise analytic (or at least smooth) solutions or data, should be based on higher order methods. These are the p–version FEM on solution dependent adapted grids and (from the scratch) the h–p–version. For ease of coding p-extensions should be omitted or encapsulated by object oriented coding during development of new numerical procedures.

Symbols

$\mathbb{R}$	real numbers
$\mathbb{N}$	natural numbers
d	number of space dimensions, domain is in $\mathbb{R}^d$
$\Omega \subset \mathbb{R}^d$	polyhedral domain
$\mathrm{L}^2(\Omega)$	space of square integrable functions
$\mathrm{H}^\alpha(\Omega)$	Sobolev space, α-th derivatives are in $\mathrm{L}^2(\Omega)$
n	number of unknowns, number of degrees of freedom
h	step-size, diameter of the finite elements
p	local approximation order, polynomial degree of the shape functions
π	in chapter 2.2 on page 60 only: number of parallel processors
$\mathcal{O}(n)$	Landau order symbol
a	second order selfadjoint elliptic differential operator (introduction on page 3)
$a(.,.)$	weak formulation of a
$(b_0, b_1, \ldots, b_d)$	barycentric coordinates in $\mathbb{R}^d$, normed by $\sum_{i=0}^d b_i(x) = 1,\ \forall x$ (chapter 1.1 on page 13)
$\mathcal{P}_p^d$	polynomials of degree p in d variables
S_{d+1}	group of permutations of the set $(1, 2, \ldots, d, d+1)$ (chapter 1.2.3 on page 31)
$\mathrm{S}_{d+1}^+ \subset \mathrm{S}_{d+1}$	alternating group, even permutations
$\mathrm{S}_{d+1}^\pm$	anti-symmetry of functions $\mathbb{R}^d \to \mathbb{R}$, based on S_{d+1}^+ expressed in barycentric coordinates (chapter 1.2.3 on page 31)
$\mathbb{E}^d$	projection of the Euclidean space $\mathbb{R}^d$ onto the hyper-plane $x_d = 1$ (chapter 2.4 on page 64)

References

[AF26] P. Appell and J. Kampé de Fériet. *Fonctions Hypergéométriques et Hyperspheriques–Polynomes d'Hermite.* Gauthier–Villars, Paris, 1926.

[BA76] I. Babuška and A. K. Aziz. On the angle condition in the finite element method. *SIAM J. Numer. Anal.*, 13:214–226, 1976.

[Bab93] I. Babuška. Courant element: Bofore and after. Technical Report BN-1154, Univ. of Maryland, College Park, MD, 1993.

[Ban82] R. E. Bank. PLTMG Users' guide, june 1981 version. Technical report, Dept. of Mathematics, Univ. of Calif., San Diego, 1982.

[Bän91] E. Bänsch. Local mesh refinement in 2 and 3 dimensions. *IMPACT Comput. Sci. Engrg.*, 3:181–191, 1991.

[Ban94] R. E. Bank. *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations, Users' Guide 7.0.* SIAM, Philadelphia, 1994.

[Bas93] P. Bastian. *Parallel Adaptive Multigrid Methods.* IWR, Universität Heidelberg, 1993. Preprint 93–60.

[BCMP91] I. Babuška, A. Craig, J. Mandel, and J. Pitkäranta. Efficient preconditioning for the p-version finite element method in two dimensions. *SIAM J. Numer. Anal.*, 28(3):624–661, 1991.

[BD81a] I. Babuška and M. R. Dorr. Error estimates for the combined h and p versions of the finite element method. *Numer. Math.*, 37:257–277, 1981.

[BD81b] R. E. Bank and T. F. Dupont. An optimal order process for solving elliptic finite element equations. *Math. Comp.*, 36:967–975, 1981.

[BD95] F. A. Bornemann and P. Deuflhard. Cascadic multigrid method for elliptic problems. *Numer. Math.*, 1995. to appear.

[BDR92] I. Babuška, R. Duran, and R. Rodriguez. Analysis of the efficiency of an a posteriori error estimator for linear triangular finite elements. *SIAM J. Numer. Anal.*, 29(4):947–964, 1992.

[Bec93] R. Beck. *Feldberechnung in dreidimensionalen Leitungsstrukturen der Mikroelektronik mittels p-adaptiver Finite-Elemente-Methoden.* PhD thesis, TU Berlin, 1993.

[BEK93a] F. A. Bornemann, B. Erdmann, and R. Kornhuber. Adaptive multilevel methods in three space dimensions. *Internat. J. Numer. Methods Engrg.*, 36:3187–3202, 1993.

[BEK93b] F. A. Bornemann, B. Erdmann, and R. Kornhuber. A posteriori error estimates for elliptic problems in two and three space dimensions. *SIAM J. Numer. Anal.*, 1993. to appear.

[BEM92] I. Babuška, H. Elman, and K. Markley. Parallel implementation of the hp-version of the finite element method on a shared-memory architecture. *SIAM J. Sci. Statist. Comput.*, 13:1433–1459, 1992.

[BG88] I. Babuška and B. Q. Guo. The h-p version of the finite element method for domains with curved boundaries. *SIAM J. Numer. Anal.*, 25(4):837–861, 1988.

[BGP89] I. Babuška, M. Griebel, and J. Pitkäranta. The problem of selecting the shape functions for a p-type element method. *Internat. J. Numer. Methods Engrg.*, 28:1891–1908, 1989.

[BH82] A. N. Brooks and T. J. R. Hughes. Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Comput. Methods Appl. Mech. Engrg.*, 32:199–259, 1982.

[Bir93] K. Birken. Ein Parallelisierungskonzept für adaptive, numerische Berechnungen. Diplomarbeit, Universität Erlangen-Nürnberg, 1993.

[BKP79] I. Babuška, R. B. Kellogg, and J. Pitkäranta. Direct and inverse estimates for finite elements with mesh refinements. *Numer. Math.*, 33:447–471, 1979.

[BM87] I. Babuška and A. Miller. A feedback finite element method with a posteriori error estimation: Part i. the finite element method and some basic properties of the a posteriori error estimator. *Comput. Methods Appl. Mech. Engrg.*, 61:1–40, 1987.

[Bor91] F. A. Bornemann. *A Sharpened Condition Number Estimate for the BPX Preconditioner of Elliptic Finite Element Problems on Highly Nonuniform Triangulations.* Konrad–Zuse–Zentrum, Berlin, 1991. Preprint SC 91–9.

[BPX90] J. H. Bramble, J. E. Pasciak, and J. Xu. Parallel multilevel preconditioners. *Math. Comp.*, 55:1–22, 1990.

[BR78] I. Babuška and W. C. Rheinboldt. Error estimates for adaptive finite element computations. *SIAM J. Numer. Anal.*, 15:736–754, 1978.

[BR87] I. Babuška and E. Rank. An expert system like feedback approach in the h-p version of the finite element method. *Finite Elements and Design*, 3:127–147, 1987.

[Bra76] A. Brandt. Multi-level adaptive solutions to boundary value problems. *Math. Comp.*, 31:333–390, 1976.

[Bra92] D. Braess. *Finite Elemente.* Springer, Berlin, 1992.

[BS90] I. Babuška and M. Suri. The p- and h-p versions of the finite element method. An overview. *Comput. Methods Appl. Mech. Engrg.*, 80(1–3):5–26, 1990.

[BS94] I. Babuška and M. Suri. The p and h-p-versions of the finite element method, basic principles and properties. *SIAM Rev.*, 36(4):578–632, 1994.

[BSK81] I. Babuška, B. A. Szabó, and I. N. Katz. The p-version of the finite element method. *SIAM J. Numer. Anal.*, 18(3):515–545, 1981.

[BSS91] H. Berryman, J. Saltz, and J. Scroggs. Execution time support for adaptive scientific algorithms on distributed memory machines. *Concurrency: Practice and Experience*, 3:159–178, 1991.

[BSW83] R. E. Bank, A. H. Sherman, and H. Weiser. Refinement algorithms and data structures for regular local mesh refinement. In *Scientific Computing, R. Stepleman (ed.)*, pages 3–17, Amsterdam, 1983. IMACS North-Holland.

[BW60] B. A. Boley and J. H. Weiner. *Theory of Thermal Stresses*. J. Wiley & Sons, New York, 1960.

[BW85] R. E. Bank and A. Weiser. Some a-posteriori error estimators for elliptic partial differential equations. *Math. Comp.*, 44, 1985.

[BW93] P. Bastian and G. Wittum. *On Robust and Adaptive Multigrid Methods*. IWR, Universität Heidelberg, 1993. Preprint 93–59.

[BY87] I. Babuška and D. Yu. Asymptotically exact a posteriori error estimator for biquadratic elements. *Finite Elem. Anal. Des.*, 3:341–354, 1987.

[Cia80] P. G. Ciarlet. *The Finite Element Method for Elliptic Problems*. North–Holland, Amsterdam, 1980.

[Cia88] P. G. Ciarlet. *Mathematical Elasticity*, volume 1. North–Holland, Amsterdam, 1988.

[Cia91] P. G. Ciarlet. Basic error estimates for elliptic problems. In P. G. Ciarlet and J. L. Lions, editors, *Handbook of Numerical Analysis*, volume 2, pages 17–351, Amsterdam, 1991. North–Holland.

[CL90] C. K. Chui and Ming-Jun Lai. Multivariate vertex splines and finite elements. *J. Approx. Theory*, 60(3):245–343, 1990.

[CMTT93] P. Carnevali, R. B. Morris, Y. Tsuji, and G. Taylor. New basis functions and computational procedures for p-version finite element analysis. *Internat. J. Numer. Methods Engrg.*, 36:3759–3779, 1993.

[Coo68] S. Coons. Rational bicubic surface patches. Technical Report, MIT, Project MAC, 1968.

[Cri91] M. A. Crisfield. *Non-Linear Finite Element Analysis of Solids and Structures*, volume 1. J. Wiley & Sons, Chichester, 1991.

[CT65] R. Clough and J. Tocher. Finite element stiffness matrices for analysis of plates in bending. In *Proc. of Conference on Matrix Methods in Structural Mechanics*, 1965.

[CW90] R. R. Coifman and M. V. Wickerhauser. Best-adapted wave packet bases. via ftp, Yale Univ., New Haven, 1990.

[CZZ84] A. W. Craig, J. Z. Zhu, and O. C. Zienkiewicz. A posteriori error estimation, adaptive mesh refinement and multigrid methods using hierarchical finite element bases. In *MAFELAP84, The Mathematics of Finite Elements and Applications*, 1984.

[Dau88] M. Dauge. *Elliptic Boundary Value Problems on Corner Domains*. Springer, Berlin, 1988.

[Deu83] P. Deuflhard. Order and stepsize control in extrapolation methods. *Numer. Math.*, 41:399–422, 1983.

[Deu94] P. Deuflhard. Cascadic conjugate gradient methods for elliptic partial differential equations. Algorithm and numerical results. In D. Keyes and J. Xu, editors, *Proceedings of the 7th International Conference on Domain Decomposition Methods 1993*. AMS, Providence, 1994.

[DH93] P. Deuflhard and A. Hohmann. *Numerische Mathematik I. Eine algorithmisch orientierte Einführung*. de Gruyter, Berlin, 2. edition, 1993.

[DKO92] L. Demkowicz, A. Karafiat, and J.T. Oden. Solution of elastic scattering problems in linear acoustics using *h-p* boundary element method. *Comput. Methods Appl. Mech. Engrg.*, 101:251–282, 1992.

[DL76] G. Duvaut and J. L. Lions. *Inequalities in Mechanics and Physics*. Springer, Berlin, 1976.

[DLY89] P. Deuflhard, P. Leinen, and H. Yserentant. Concepts of an adaptive hierarchical finite element code. *IMPACT Comput. Sci. Engrg.*, 1:3–35, 1989.

[Dör94] W. Dörfler. A convergent adaptive algorithm for poisson's equation. *SIAM J. Numer. Anal.*, 1994. to appear.

[DORH89] L. Demkowicz, J. T. Oden, W. Rachowicz, and O. Hardy. Toward a universal *h-p* adaptive finite element strategy. i: Constrained approximation and data structure. *Comput. Methods Appl. Mech. Engrg.*, 77:79–112, 1989.

[Dub91] M. Dubiner. Spectral methods on triangles and other domains. *J. Sci. Comp.*, 6(4):345–390, 1991.

[ERFA91] B. Erdmann, R. Roitzsch, and Bornemann F. A. *KASKADE, Numerical Experiments*. Konrad-Zuse-Zentrum, Berlin, 1991. Technical Report TR 91-1.

[Far90] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design – A Practical Guide*. Academic Press, San Diego, 2nd edition, 1990.

[FLO76] W. N. Findley, J. S. Lai, and K. Onaran. *Creep and Relaxation of Nonlinear Viscoelastic Materials*. North–Holland, Amsterdam, 1976.

[FP93] D. A. Field and Y. Pressburger. An h-p-multigrid method for finite element analysis. *Internat. J. Numer. Methods Engrg.*, 36:893–908, 1993.

[GB86a] W. Gui and I. Babuška. The h, p and h-p versions of the finite element method in 1 dimension. i. The error analysis of the p-version. *Numer. Math.*, 49:577–612, 1986.

[GB86b] W. Gui and I. Babuška. The h, p and h-p versions of the finite element method in 1 dimension. ii. The error analysis of the h- and h-p versions. *Numer. Math.*, 49:613–657, 1986.

[GB86c] W. Gui and I. Babuška. The h, p and h-p versions of the finite element method in 1 dimension. iii. The adaptive h-p version. *Numer. Math.*, 49:659–683, 1986.

[GM78] A. Grundmann and H. M. Möller. Invariant integration formulas for the n-simplex by combinatorial methods. *SIAM J. Numer. Anal.*, 15(2):282–290, 1978.

[Gri89] P. Grisvard. Singularités en elasticité. *Arch. Rational Mech. Anal.*, 107:157–180, 1989.

[Gri92] P. Grisvard. *Singularities in Boundary Value Problems.* Masson, Paris, Springer, Berlin, 1992.

[Gri94] M. Griebel. *Punktblock-Multilevelmethoden zur Lösung elliptischer Differentialgleichungen.* Teubner, Stuttgart, 1994.

[Guo93] B. Q. Guo. The h-p version of the finite element method for solving boundary value problems in polyhedral domains. In *Proceedings of the International Conference on Boundary Value Problems and Integral Equations in Nonsmooth Domains*, CIRM Luminy, France, 1993.

[Gur81] M. E. Gurtin. *An Introduction to Continuum Mechanics.* Academic Press, Orlando, 1981.

[Hea93] A. C. Hearn. *Reduce User's Manual 3.5.* Rand, Santa Monica, CA 90407-2138, 1993.

[HHNL88] I. Hlaváček, J. Haslinger, J. Nečas, and J. Lovíšek. *Solution of Variational Inequalities in Mechanics.* Springer, New York, 1988.

[Hip95] R. Hiptmair. Object oriented concepts for an adaptive finite element code. via www, TU München, 1995.

[HK90] R. H. W. Hoppe and R. Kornhuber. Multi-grid solution of two coupled stefan equations arising in induction heating of large steel slabs. *Internat. J. Numer. Methods Engrg.*, 30:779–801, 1990.

[HK94] R. H. W. Hoppe and R. Kornhuber. Adaptive multilevel-methods for obstacle problems. *SIAM J. Numer. Anal.*, 31:301–323, 1994.

[Hoh94] A. Hohmann. *Inexact Gauss Newton Methods for Parameter Dependent Nonlinear Problems.* PhD thesis, FU Berlin, 1994.

[Hop90] R. H. W. Hoppe. Une méthode multigrille pour la solution des problèmes d' obstacle. *RAIRO Modél. Math. Anal. Numér.*, 24:711–736, 1990.

[IR72] B. M. Irons and A. Razzaque. Experience with the patch test for convergence of finite element method. In *Mathematical foundations of the finite element method (ed. A.K. Aziz)*, pages 557–587. Academic Press, 1972.

[Jac36] D. Jackson. Formal properties of orthogonal polynomials in two variables. *Duke Math. J.*, 2:423–434, 1936.

[Jam76] P. Jamet. Estimations d'erreur pour des elements finis droits presque degeneres. *RAIRO Anal. Numér.*, 10:43–61, 1976.

[Jar86] H. Jarausch. On an adaptive grid refining technique for finite element approximations. *SIAM J. Sci. Statist. Comput.*, 7:1105–1120, 1986.

[JH92] C. Johnson and P. Hansbo. Adaptive finite element methods in computational mechanics. *Comput. Methods Appl. Mech. Engrg.*, 101:143–181, 1992.

[JP92] M.T. Jones and P.E. Plassmann. *Parallel Algorithms for the Adaptive Refinement and Partitioning of Unstructured Meshes.* Argonne National Laboratory, 1992.

[KO88] N. Kikuchi and J. T. Oden, editors. *Contact Problems in Elasticity: A Study of Variational Inequalities and Finite Element Methods.* SIAM, Philadelphia, 1988.

[KOR95] M. Kaddoura, C.-W. Ou, and S. Ranka. Mapping unstructured computational graphs for adaptive and nonuniform computational environments. *IEEE Parallel and Distributed Technology*, page submitted, 1995.

[Kos82] A. I. Kostrikin. *Intorduction to Algebra.* Springer, New York, 1982.

[KR90] R. Kornhuber and R. Roitzsch. On adaptive grid refinement in the presence of internal or boundary layers. *IMPACT Comput. Sci. Engrg.*, 2:40–72, 1990.

[Kri92] M. Křížek. On the maximum angle condition for linear tetrahedral elements. *SIAM J. Numer. Anal.*, 29(2):513–520, 1992.

[Le 90] P. Le Tallec. *Numerical Analysis of Viscoelastic Problems.* Masson, Paris, Springer, Berlin, 1990.

[Le 94] P. Le Tallec. Numerical methods for nonlinear three-dimensional elasticity. In P. G. Ciarlet and J. L. Lions, editors, *Handbook of Numerical Analysis*, volume 3, pages 465–622, Amsterdam, 1994. North–Holland.

[Lei90] P. Leinen. *Ein schneller adaptiver Löser für elliptische Randwertprobleme auf Seriell– und Parallelrechnern.* PhD thesis, Universität Dortmund, 1990.

[Lem94] M. Lemke. *Multilevelverfahren mit selbstadaptiven Gitterverfeinerungen für Parallelrechner mit verteiltem Speicher.* Oldenburg, München, 1994.

[Man90a] J. Mandel. Hierarchical preconditioning and partial orthogonalization for the p-version finite element method. In T. Chan, R. Glowinski, J. Periaux, and O. Widlund, editors, *Proceedings Third Internat. Symp. on Domain Decomposition Methods for Partial Differential Equations*, pages 141–156. SIAM, 1990.

[Man90b] J. Mandel. Two-level domain decomposition preconditioning for the p-version finite element method in three dimensions. *Internat. J. Numer. Methods Engrg.*, 29:1095–1108, 1990.

[Mat92] The Math Works, Inc., Natick, Mass. 01760. *Matlab, High-Performance Numeric Computation and Visualization Software*, 1992.

[McC89] S. McCormick. *Multilevel Adaptive Methods for Partial Differential Equations.* SIAM, Philadelphia, 1989.

[Mit89] W. F. Mitchell. A comparison of adaptive refinement techniques for elliptic problems. *ACM Trans. Math. Software*, 15:326–347, 1989.

[MMS82] C. Mesztenyi, A. Miller, and W. Szymczak. FEARS: Details of mathematical formulation UNIVAC 1100. Technical Report BN-994, Univ. of Maryland, College Park, 1982.

[MP72] A. R. Mitchell and G. M. Phillips. Construction of basis functions in the finite element method. *BIT*, 12:81–89, 1972.

[MTC92] R. B. Morris, Y. Tsuji, and P. Carnevali. Adaptive solution strategy for solving large systems of p-type finite element equations. *Internat. J. Numer. Methods Engrg.*, 33:2059–2071, 1992.

[Mys81] I. P. Mysowskich. *Interpoljazionnye Kubaturnye Formuly.* Nauka, Moskau, 1981.

[Nic72] R. A. Nicolaides. On a class of finite elements generated by Lagrange interpolation. *SIAM J. Numer. Anal.*, 9(3):435–445, 1972.

[NP94] S. A. Nazarov and B. A. Plamenevsky. *Elliptic Problems in Domains with Piecewise Smooth Boundaries.* de Gruyter, Berlin, 1994.

[NVL93] R. V. Nambiar, R. S. Valera, and K. L. Lawrence. An algorithm for adaptive refinement of triangular element meshes. *Internat. J. Numer. Methods Engrg.*, 36:499–509, 1993.

[Ode91] J. T. Oden. Finite elements: An introduction. In P. G. Ciarlet and J. L. Lions, editors, *Handbook of Numerical Analysis*, volume 2, pages 3–15, Amsterdam, 1991. North–Holland.

[Ode94] J. T. Oden. Optimal *h-p* finite element methods. *Comput. Methods Appl. Mech. Engrg.*, 112, 1994.

[ODRW89] J. T. Oden, L. Demkowicz, W. Rachowicz, and T. A. Westermann. Toward a universal *h-p* adaptive finite element strategy. ii: A posteriori error estimation. *Comput. Methods Appl. Mech. Engrg.*, 77:113–180, 1989.

[OPF93] J. T. Oden, A. Patra, and Y. Feng. Domain decomposition for adaptive *hp* finite element methods. Technical report, TICOM, Univ. of Texas at Austin, Tx., 1993.

[OR70] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables.* Academic Press, Orlando, San Diego, New York, 1970.

[Osw90] P. Oswald. On function spaces related to finite element approximation theory. *Z. Anal. Anwendungen*, 9(2):43–64, 1990.

[Pav92] L. F. Pavarino. *Domain Decomposition Algorithms for the p-Version Finite Element Method for Elliptic Problems.* PhD thesis, Report TR616, Courant Institute, New York, 1992.

[Pav94a] L. F. Pavarino. Additive Schwarz methods for the p-version finite element method. *Numer. Math.*, 66(4):493–515, 1994.

[Pav94b] L. F. Pavarino. Schwarz methods with local refinement for the p-version finite element method. *Numer. Math.*, 69(2):185–211, 1994.

[Pea76] A. G. Peano. Hierarchies of conforming finite elements for plane elasticity and plate bending. *Comput. Math. Appl.*, 2:211–224, 1976.

[PO93] A. Patra and J. T. Oden. Problem decomposition for adaptive *hp* finite element methods. Technical report, TICOM, Univ. of Texas at Austin, Tx., 1993.

[PSL90] A. Pothen, H.D. Simon, and K.-P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM J. Matrix Anal. Appl.*, 11:430–452, 1990.

[Ran78] R. Rannacher. *Probleme der Elastizitätstheorie und ihre numerische Behandlung.* Univ. Bonn, 1978. Vorlesungsskriptum.

[Rhe80] W. C. Rheinboldt. On a theory of mesh-refinement processes. *SIAM J. Numer. Anal.*, 17:766–778, 1980.

[Riv84a] M. C. Rivara. Algorithms for refining triangular grids suitable for adaptive and multigrid techniques. *Internat. J. Numer. Methods Engrg.*, 20:745–756, 1984.

[Riv84b] M. C. Rivara. EXPDES Users' guide. Technical report, Catholic Univ. Leuven, Belgium, 1984.

[ROD89] W. Rachowicz, J. T. Oden, and L. Demkowicz. Toward a universal h-p adaptive finite element strategy. iii: Design of h-p meshes. *Comput. Methods Appl. Mech. Engrg.*, 77:181–212, 1989.

[Roi89a] R. Roitzsch. *KASKADE Programmer's Manual.* Konrad-Zuse-Zentrum, Berlin, 1989. Technical Report TR 89-5.

[Roi89b] R. Roitzsch. *KASKADE User's Manual.* Konrad-Zuse-Zentrum, Berlin, 1989. Technical Report TR 89-4.

[RR93a] L. Rose and D. Ramey. *Advanced Visualizer User's Guide, Version 4.0.* Wavefront Technologies, Inc., Santa Barbara, CA 93103, 3rd edition, 1993.

[RR93b] L. Rose and D. Ramey. *Wavefront File Formats, Version 4.0.* Wavefront Technologies, Inc., Santa Barbara, CA 93103, 1st edition, 1993.

[RT91] J. E. Roberts and J.-M. Thomas. Mixed and hybrid methods. In P. G. Ciarlet and J. L. Lions, editors, *Handbook of Numerical Analysis*, volume 2, pages 523–639, Amsterdam, 1991. North–Holland.

[SB91] B. A. Szabó and I. Babuška. *Finite Element Analysis.* J. Wiley & Sons, New York, 1991.

[Sha94] V. V. Shaidurov. *Some Estimates of the Rate of Convergence for the Cascadic Conjugate-Gradient Method.* Otto-von-Guericke-Universität, Magdeburg, 1994. Preprint.

[Sil] Power series. Technical report, Silicon Graphics, Inc., Mountain View, CA 94039-7311.

[SK95] S. J. Sherwin and G. E. Karniadakis. Triangular and tetrahedral spectral elements. In *ICOSAHOM'95*, 1995.

[SW92] C. Schütte and M. Wulkow. Quantum theory with discrete spectra and countable systems of differential equations – a numerical treatment of infrared spectroscopy. Technical Report SC 92-7, Konrad-Zuse-Zentrum, 1992.

[Sza85] B. A. Szabó. *PROBE: Theoretical Manual, Release 1.0.* Noetic Technologies Corporation, 7980 Clayton Road, Suite 205, St. Louis, MO 63117, 1985.

[Sza86] B. A. Szabó. Implementation of a finite element software system with h and p extension capabilities. *Finite Elements in Analysis and Design*, 2:177–194, 1986.

[Vog83] M. Vogelius. An analysis of the p-version of the finite element method for nearly incompressible materials. *Numer. Math.*, 41:39–53, 1983.

[Wac75] E. L. Wachspress. *A Rational Finite Element Basis*. Academic Press, New York, 1975.

[WCE95] C. Walshaw, M. Cross, and M. G. Everett. A parallelisable algorithm for optimising unstructured mesh partitions. Technical report, School of Math., Stat. & Scientific Comp., Univ. of Greenwich, London, 1995.

[Web88] J. P. Webb. Finite element analysis of dispersion in waveguides with sharp metal edges. *IEEE Trans. Microwave Theory Tech.*, 36(12):1819–1825, 1988.

[Wil92] R. Williams. *A Dynamic Solution-Adaptive Unstructured Parallel Solver*. Caltech CCSF, Pasadena, 1992. Report 21–92.

[WR92] A. Wierse and M. Rumpf. GRAPE - Eine objektorientierte Visualisierungs- und Numerikplattform. *Informatik Forsch. Entw.*, 7, 1992.

[Xu89] J. Xu. *Theory of Multilevel Methods*. PhD thesis, Report No. AM 48, Pennsylvania State University, Department of Mathematics, 1989.

[Xu92] J. Xu. Iterative methods by space decomposition and subspace correction. *SIAM Rev.*, 34:581–613, 1992.

[Yse86] H. Yserentant. On the multilevel splitting of finite element spaces. *Numer. Math.*, 49:379–412, 1986.

[Zla68] M. Zlamal. On the finite element method. *Numer. Math.*, 12:394–409, 1968.

[ZT89] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method*, volume 1. McGraw-Hill, Maidenhead, 4th edition, 1989.

[ZT91] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method*, volume 2. McGraw-Hill, Maidenhead, 4th edition, 1991.

[Zum91] G. W. Zumbusch. Adaptive parallele Multilevel-Methoden zur Lösung elliptischer Randwertprobleme. SFB-Report 342/19/91A, TUM-I9127, TU München, 1991.

[Zum93] G. W. Zumbusch. Symmetric hierarchical polynomials for the h-p-version of finite elements. Preprint SC 93-32 ZIB, 1993.

[Zum94] G. W. Zumbusch. Visualizing functions of the h-p-version of finite elements. Technical Report TR 94-5, Konrad-Zuse-Zentrum, Berlin, 1994.

[ZW92] L. F. Zeng and N.-E. Wiberg. Adaptive h-p procedures for high accuracy finite element analysis of two-dimensional linear elastic problems. *Comput. & Structures*, 42(6):869–886, 1992.

[ZZCA89] O. C. Zienkiewicz, J. Z. Zhu, A. W. Craig, and M. Ainsworth. Simple and practical error estimation and adaptivity: h and h-p version procedures. In *Adaptive methods for partial differential equations*, pages 100–114, Troy / New York, 1989. Workshop 1988, SIAM.