Meshfree and Generalized Finite Element Methods

Habilitationsschrift

an der

Mathematisch–Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich–Wilhelms–Universität Bonn

vorgelegt von

Marc Alexander Schweitzer

aus

Frankfurt am Main

Bonn 2008

Contents

Part I

Generalized Finite Element and Meshfree Methods

Chapter 1
Introduction

The classical finite element method (FEM) is a well-established tool in scientific computing and widely used in many areas of application, see e.g. [160, 161]. The success of the FEM can be attributed in part to its improved geometry handling compared with the finite difference or finite volume methods, and to the fact that finite element (FE) shape functions ϕ_i^{FE} are piecewise polynomial.

These features render the FEM a very versatile numerical approach and it can be regarded as a general purpose solver e.g. for the discretization of partial differential equations (PDE). However, this flexibility comes at a prize — mesh-generation. The construction of good quality meshes is not an easy task and accounts for a large percentage of the total (computational and economical) cost of a FE simulation. Moreover, we must acknowledge the fact that (piecewise) polynomials are very much appropriate for the approximation of smooth functions but they are not tailored for the approximation of non-smooth functions. Here, local *geometric* mesh refinement must be employed. This may yield an optimal asymptotic convergence behavior but can involve a large number of refinement steps and degrees of freedom to reach the required accuracy.

This effect can only be avoided by abandoning the restriction to piecewise polynomial shape functions in the FEM; i.e. by the generalization of the FEM. Then, we can employ an *algebraic* refinement of the approximation space V^{FE} which can provide a much more efficient approximation than geometric mesh refinement. However, the incorporation of non-polynomial shape functions in V^{FE} must respect the global regularity constraints, i.e. the inter-element continuity conditions. To this end, the partition of unity (PU) property

$$\sum_{i=1}^{N} \phi_i^{\mathrm{FE}} \equiv 1$$

of the piecewise polynomial FE shape functions ϕ_i^{FE} is utilized. With the PU approach we attain an *enriched* approximation space by

$$V_E^{\mathrm{FE}} := V^{\mathrm{FE}} + \eta \sum_{\lambda \in \Lambda} \phi_\lambda^{\mathrm{FE}} \tag{1.1}$$

where η denotes a specific (solution- or problem-dependent) non-smooth enrichment function and $\Lambda \subset \{1, \ldots, N\}$ defines the subset of algebraically refined FE shape functions. This first generalization of the classical FEM lead to the introduction of the special finite elements of [7], the extended finite element method (XFEM) [16, 20, 100, 101], and the generalized finite element method [5, 42, 44, 44, 45, 134, 135].

Yet, the algebraic refinement (1.1) of the classical FE approximation space V^{FE} may compromise the stability of the basis $\langle \phi_i^{\mathrm{FE}}, \phi_\lambda^{\mathrm{FE}} \eta \rangle$ and can yield an ill-conditioned or even singular stiffness matrix. Thus, the improved approximation properties of V_E^{FE} may come at a high prize since the (iterative) solution of the arising linear system can be challenging and very expensive.

To overcome this drawback of the algebraic refinement approach we must impose an additional assumption on the employed PU — the flat top property [8, 65, 125]. This property, however, is not satisfied by the FE shape functions ϕ_i^{FE} so that the second generalization of the FEM which ensures the stability of the basis of an enriched approximation space of type (1.1) independently of the employed enrichment functions η cannot be carried out on a mesh-based PU.[1] Thus we abandon the mesh and construct a *meshfree* PU $\{\varphi_i\}$ which satisfies the flat top property to define the meshfree generalized finite element space

$$V^{\mathrm{PU}} := \sum_{i=1}^{N} \varphi_i V_i, \qquad \text{with} \quad V_i := \mathcal{P}^{p_i} + \mathcal{E}_i \tag{1.2}$$

where $\mathcal{P}^{p_i} = \operatorname{span}\langle \psi_i^s \rangle$ denotes the space of polynomials of degree $p \leq p_i$ and $\mathcal{E}_i = \operatorname{span}\langle \eta_i^t \rangle$ a problem-dependent (local) enrichment space.

Due to the flat top property we can easily identify a stable basis of V^{PU} so that we obtain a regular stiffness matrix which can be efficiently solved by a generalized multilevel solver. Furthermore, we have eliminated the need for the expensive mesh-generation due to the meshfree construction of the PU. Hence, a meshfree generalized FEM has by design the capabilities to outperform the classical FEM especially in the approximation of non-smooth functions. Though the conceptual advantages of a meshfree method also lead to some practical challenges e.g. the implementation of essential boundary conditions or the numerical integration of the meshfree shape functions.

In this manuscript we present a general framework for the construction of a meshfree generalized finite element space of type (1.2). Here, we rather focus on the methodology than on a specific type of enrichment, i.e. a particular class of applications. Thus, our construction yields an approximation space V^{PU} with the following properties.

1. The construction of the meshfree shape functions is simpler than full-blown mesh-generation.
2. The approximation properties of V^{PU} are determined by N, the number of the PU functions φ_i, and the choice of the local approximation spaces $V_i = \mathcal{P}^{p_i} + \mathcal{E}_i$, i.e. the polynomial degree p_i of $\mathcal{P}^{p_i}$ and the enrichment space $\mathcal{E}_i$. Thus, we can employ local h-refinement (acting on φ_i), local p-refinement (acting on $\mathcal{P}^{p_i}$), or local algebraic refinement (acting on $\mathcal{E}_i$) to optimize the approximation properties of V^{PU}.
3. The selection of a stable basis of V^{PU} requires at most $O(N)$ operations independently of the employed enrichment space $\mathcal{E}_i$.
4. The treatment of essential boundary conditions via a non-conforming discretization as well as the automatic construction of a conforming subspace $V_K^{\mathrm{PU}} \subset V^{\mathrm{PU}}$ is of optimal complexity.
5. The linear system arising from the Galerkin discretization of a PDE with V^{PU} as trial and test space can be solved efficiently by a generalized multigrid method independently of the local enrichment spaces $\mathcal{E}_i$.

Furthermore, we present a fast and (numerically) reliable scheme for the numerical integration of the constructed meshfree shape functions and the parallelization of the overall method.

[1] With special mesh-generation techniques a flat top PU can be constructed, see e.g. [117].

We not only summarize the background of meshfree and generalized finite element methods but present some recent developments and improvements which substantially advanced the maturity of meshfree methods. In particular the treatment of essential boundary conditions was somewhat cumbersome in most meshfree methods. With the popularization of Nitsche's method [108] this issue was significantly simplified. However the specific global formulation employed in [64] turned out to be less appropriate for the multilevel solution of linear systems arising from adaptive discretizations [125]. Here, we present an overlapping local formulation of Nitsche's method which not only yields optimal convergence properties but also maintains the optimal convergence behavior of our multilevel solver for adaptive discretizations.

Even though this non-conforming implementation of essential boundary conditions provides optimal convergence it has certain drawbacks. First and foremost, there is the need to construct an appropriate weak form of the PDE *analytically* which depend strongly on the configuration of the boundary conditions. Thus, it is not trivial to change boundary conditions in an interactive user-driven manner. Often a change in the boundary conditions requires some amount of implementation work and a re-assembly of the stiffness matrix on the boundary. Secondly, the essential boundary data is only weakly approximated and the error on the boundary is balanced with the error in the interior by Nitsche's approach. This can be inappropriate in situations where the boundary conditions need to be enforced strictly. Moreover, Nitsche's method is in some sense restricted to stationary problems and its application in explicit time-discretization schemes is not obvious. Thus, the realization of an efficient technique for the conforming treatment of essential boundary conditions that does not rely on very restrictive assumptions on the input data is an important research topic in meshfree methods.

In this manuscript we present an algebraic approach to the construction of a conforming subspace $V_K^{\mathrm{PU}} \subset V^{\mathrm{PU}}$ that does not involve any additional restrictions on the employed local approximation spaces or on the distribution of the degrees of freedom near the boundary. To our knowledge it is the only constructive approach to the automatic conforming treatment of essential boundary conditions and constitutes a significant improvement in the applicability and usefulness of meshfree methods outside of academia.

Moreover, we present the adaptive refinement of a meshfree approximation space V^{PU} based on a local error estimator. Here, we not only allow for a classical hp-refinement but also the algebraic refinement of V^{PU} via enrichment. The hierarchical enrichment scheme we present in this manuscript not only recovers the optimal convergence behavior of the uniform h-version independently of the regularity of the solution but in fact attains a kind of super-convergence *near* the singularities of the solution.

The remainder of this manuscript is structured as follows. In §2 we introduce the moving least squares method (MLSM) which is the foundation of many meshfree methods (MM). Here, we limit ourselves to the discussion of properties of this scattered data approach which are relevant to the construction of meshfree shape functions, i.e. a meshfree PU.

The abstract partition of unity method (PUM) [8, 9] is presented in §3. Here, we focus on its theoretical approximation properties and the implementation of essential boundary conditions with a PUM. The subject of §4 is the particle–partition of unity method (PPUM) a meshfree instance of the PUM. We present the specific construction of the meshfree PU employed in the PPUM and how a stable basis of the approximation space V^{PU} can be selected automatically and independently of the employed enrichment spaces $\mathcal{E}_i$. Furthermore, we discuss the extraction of a conforming subspace $V_K^{\mathrm{PU}} \subset V^{\mathrm{PU}}$ for the direct implementation of essential boundary conditions.

The extension of the PPUM to the multilevel setting is the topic of §5. Here, we present an automatic coarsening scheme for the approximation space $V^{\mathrm{PU}} = V_J^{\mathrm{PU}}$ to obtain an initial sequence of (non-nested) spaces V_k^{PU} with $k = 0, \dots, J$. Based on this sequence of spaces we then introduce a multilevel solver for the PPUM utilizing the flat top property of the constructed PU and specific geometric properties of the coarsening process.

Then, we turn to the question of refinement of a space V_J^{PU}. First, we consider the algebraic refinement of V_J^{PU} by hierarchical enrichment. Here, the focus is on the construction of a stable basis via a local preconditioning technique which attains the stability of the enriched basis and optimal approximation properties. The presented error analysis shows that a uniform h-refinement in conjunction with hierarchical enrichment yields the optimal convergence rate independently of the regularity of the solution. Moreover, a kind of super-convergence *near* the singularities of the solution can be observed.

Next, we are concerned with the adaptive hp-refinement of the PPUM space V_J^{PU}. To this end, we present an error estimator of subdomain type which is asymptotically reliable and efficient. With the help of this local error estimator and a simple extrapolation approach we define an hp-refinement scheme that preserves the essential properties of the resulting sequence of PPUM spaces V_k^{PU} with $k = 0, \ldots, J$ such that the multilevel construction of §5 is directly applicable.

The numerical integration of our PPUM shape functions is presented in §6. Here, we also discuss the visualization of the computed approximation and the parallelization of the overall method. The validation of the theoretical results of the PPUM is the subject of §7. Here, we focus on the approximation properties of the PPUM and the efficiency of the presented multilevel solver. The presented results clearly show that our meshfree generalized finite element method practically attains the theoretical properties of §3, §4, and §5. Finally, we conclude this manuscript with a discussion of some open questions and propose several extension of the PPUM for future research in §8.

At this point I would like to take the opportunity to express my gratitude to all my friends and colleagues who made this work possible and provided valuable input. First and foremost I thank Michael Griebel for his continuous encouragement and support over the years. He introduced me to the field of Scientific Computing and gave me the opportunity to work in this exciting field of meshfree methods. He and all of my colleagues (former and current) at the Institut für Numerische Simulation and the Institut für Angewandte Mathematik of the Rheinische Friedrich–Wilhelms–Universität Bonn contributed significantly to the very inspiring, open-minded and friendly atmosphere in the group. Thank you.

I owe special thanks to Lukas Jager and Jan Hamaekers for proof-reading the manuscript and the very helpful discussions on enrichment techniques. Furthermore, I would like to acknowledge the financial support of the Sonderforschungsbereich 611 funded by the Deutsche Forschungsgemeinschaft.

Finally, I apologize to my family for spending too much time away from them during the preparation of this manuscript. It took much longer than it should have. Undskyld.

Chapter 2
Meshfree Methods

There are (at least) two different classes of meshfree methods (MM) [48, 49, 66, 76, 90, 93, 121, 144]: classical particle schemes and techniques inspired by scattered data approximation. Traditional particle methods [2, 57, 104–107] originated from physics applications like the Boltzmann equation and are of great interest also in mathematical modeling. These schemes though are discrete and Lagrangian in nature, i.e., they can only be applied in a time-dependent setting and usually exhibit rather poor convergence properties in weak norms.[1] We, however, are interested in the approximation of a continuum model in a function space setting. Thus, we focus on the latter approach — MM stemming from scattered data techniques [17, 43, 49, 76, 77, 121, 125, 151]. Note that there exists a large variety of such methods. For instance, there is the smoothed particle hydrodynamics (SPH) technique of Lucy, Gingold and Monaghan [55, 56, 97, 102, 103, 139] which is closely related to Shepard's method [131] and was generalized by Dilts [37, 38] using the moving least squares method (MLSM) [87, 88]. The MLSM was furthermore used by Duarte and Oden [43, 46] in their hp-clouds approach and by Belytschko and coworkers [17, 18] in the element free Galerkin method (EFGM). The reproducing kernel methods of Liu et al. [89, 94, 95] are also closely connected to the MLSM; so is the generalized finite difference method (GFDM) of Liska and Orkisz [92]. Thus, many MM share the same mathematical foundation: the MLSM.[2] In the following we introduce the MLSM with the focus on the construction of shape functions for a meshfree Galerkin discretization of a PDE.

2.1 Scattered Data Approximation

The *reconstruction* of an unknown function $u : \Omega \subset \mathbb{R}^D \to \mathbb{R}$ from discrete data pairs $(x_i, f_i = u(x_i)) \in \mathbb{R}^D \times \mathbb{R}$ for $i = 1, \ldots, N$ is probably the most fundamental constructive approximation problem. The goal is the construction of an approximation u_N either such that the data are exactly matched, i.e, $u_N(x_i) = f_i$, then u_N is an *interpolant*, or in some sense approximated, i.e., $u_N(x_i) \approx f_i$. If the data are noisy the latter approach is more appropriate. In both cases the approximation

[1] The finite mass method (FMM) of Yserentant [81, 82, 155, 156] is somewhat different from the classical particle methods. The FMM is rather a discretization of the mass than of the domain which guarantees the conservation of mass. Moreover, the particles are not considered in the sense of statistical mechanics but they are viewed as relatively large mass-packets.

[2] Another scattered approach to the development of MM is the radial basis function method [49, 51, 52, 78, 79, 121, 148, 149].

can essentially be written as

$$u_N(x) := \sum_{j=1}^{N} c_j(\{f_i \,|\, i = 1, \dots, N\})\, \phi_j(\{x_i \,|\, i = 1, \dots, N\}, x)$$

where the coefficients c_j depend on the data values $\{f_i \,|\, i = 1, \dots, N\}$ and the basis functions ϕ_j on the data sites $\mathcal{X}_N := \{x_i \,|\, i = 1, \dots, N\}$. Note that we do not assume a specific regularity in the distribution of the data sites or sampling points $\mathcal{X}_N$, i.e. the data sites are *scattered* and there is no *given* connectivity pattern among the data sites.

The reconstruction problem arises in a wide arena of applications, see e.g. [121] and the references therein, and serves as a basis for our considerations concerning the construction of meshfree basis functions ϕ_j for the Galerkin discretization of a PDE.

Let us consider the vector space $\mathcal{P}^k(\Omega)$ with $d^{\mathcal{P}^k} := \dim(\mathcal{P}^k)$ of all polynomials p with total degree less than or equal to k. Furthermore, let a set of pairs $(x_i, f_i) \in \mathbb{R}^D \times \mathbb{R}$ with sampling points $x_i \in \mathbb{R}^D$ and data values $f_i \in \mathbb{R}$ for $i = 1, \dots, N$ be given. We obtain the classical *least squares fit* to the data f_i at positions x_i by the minimization of the quadratic energy

$$J_{\mathrm{LS}}(\pi) := \sum_{i=1}^{N} (f_i - \pi(x_i))^2 \tag{2.1}$$

over all polynomials $\pi \in \mathcal{P}_k(\Omega)$. Setting the first variation δJ_{LS} to zero and choosing a particular basis $\{p_q \,|\, q = 1, \dots, d^{\mathcal{P}^k}\}$ for $\mathcal{P}_k$ we find the solution to the minimization problem by the solution of the linear system

$$G_{\mathrm{LS}} \tilde{u} = \hat{f}_{\mathrm{LS}} \tag{2.2}$$

where the entries $G_{\mathrm{LS}_{q,r}}$ of the system matrix $G_{\mathrm{LS}} \in \mathbb{R}^{d^{\mathcal{P}^k} \times d^{\mathcal{P}^k}}$ are given by

$$G_{\mathrm{LS}_{q,r}} := \sum_{i=1}^{N} p_q(x_i) p_r(x_i) \quad \text{for all } q, r = 1, \dots, d^{\mathcal{P}^k}$$

and the vector $\hat{f}_{\mathrm{LS}} \in \mathbb{R}^{d^{\mathcal{P}^k}}$ on the right-hand side is defined as

$$\hat{f}_{\mathrm{LS}} := \Big(\sum_{i=1}^{N} f_i p_q(x_i) \Big)_{q=1}^{d^{\mathcal{P}^k}}.$$

The minimizing polynomial π_{LS} is then simply

$$\pi_{\mathrm{LS}}(x) = \sum_{q=1}^{d^{\mathcal{P}^k}} \tilde{u}_q p_q(x).$$

Note that the approximation π_{LS} does in general *not* align with the data, i.e. $\pi_{\mathrm{LS}}(x_i) \neq f_i$.

We define the least squares operator A_{LS} which maps a data vector $\tilde{f} = (f_i)_{i=1}^N$ to its associated polynomial $\pi_{\mathrm{LS}} \in \mathcal{P}_k$ by $A_{\mathrm{LS}} \tilde{f} = \pi_{\mathrm{LS}}$. Note that the operator A_{LS} maps elements from a vector space of dimension N to elements of a space of dimension $d^{\mathcal{P}^k}$. To obtain a uniquely solvable linear system (2.2) it is a necessary condition that $d^{\mathcal{P}^k} \leq N$. In fact we attain a unique solution to (2.2) if and only if the set of sampling points $\mathcal{X}_N$ is $\mathcal{P}_k(\Omega)$-unisolvent, see Definition 2.1 and Theorem 2.1.

Definition 2.1 ($\mathcal{V}$-unisolvent)**.** A set $Y \subset \mathbb{R}^D$ is called $\mathcal{V}$-unisolvent, if for all $\phi \in \mathcal{V}$ the implication

$$\phi|_Y = 0 \qquad \Rightarrow \qquad \phi \equiv 0$$

holds.

To assess the quality of the least squares approach let us consider the approximation of a smooth function $u \in \mathcal{C}^r(\Omega)$. Given a set of sampling points $\mathcal{X}_N$, we define the data values $f_i = u(x_i)$. Obviously, the error $u(x) - \pi_{\mathrm{LS}}(x)$ with $\pi_{\mathrm{LS}} = A_{\mathrm{LS}}\tilde{f} \in \mathcal{P}_k$ can be bounded with respect to the polynomial degree k. However, increasing the number of sampling points N will provide no further reduction of the error. If we choose the maximal polynomial degree $K = K(N)$ for which the set $\mathcal{X}_N$ is $\mathcal{P}_K$-unisolvent we have a unique solution of (2.2) and the error can be bounded with respect to $K = K(N)$, i.e. with respect to $\min(K, r)$. However, the condition number of G_{LS} will deteriorate rapidly with increasing K and the least squares approach will become instable just like interpolation using global polynomials. Keeping in mind that we approximate a smooth function u we can make use of all available information, i.e. an increasing number of sampling points, by a localization approach. Recall that the value of the least squares approximation π_{LS} at a particular point x involves *all* data pairs $(x_i, f_i = u(x_i))$. However, for a smooth function u it is clear that values $f_i = u(x_i)$ with x_i close to the point of evaluation x provide all relevant information already. Hence, it is very natural to extend the least squares approach in the following way, see also [49, 53, 98, 150, 151].

2.2 Moving Least Squares Method

Consider a locally supported non-negative function $\mathcal{W}$ often referred to as *window function* or *weight function* and the pointwise *moving least squares energy*

$$J_{\mathrm{MLS}}(\pi)(x) := \sum_{i=1}^{N} \mathcal{W}(x - x_i)(f_i - \pi(x_i))^2. \tag{2.3}$$

Note that (2.3) is defined for all $x \in \Omega$ and formally involves all data pairs (x_i, f_i) for each point of evaluation $x \in \Omega$. Utilizing the fact that the weight function is locally supported we can rewrite (2.3) to obtain

$$J_{\mathrm{MLS}}(\pi)(x) = \sum_{i=1}^{N} \mathcal{W}(x - x_i)(f_i - \pi(x_i))^2 = \sum_{\substack{x_i \in \mathcal{X}_N \\ \mathcal{W}(x-x_i)>0}} \mathcal{W}(x - x_i)(f_i - \pi(x_i))^2.$$

With the convention $\omega_i := \operatorname{supp}^\circ \mathcal{W}(\cdot - x_i)$ and Definition 2.2 of a neighborhood $\mathcal{N}(x) \subset \mathcal{X}_N$ of an arbitrary point $x \in \Omega$, compare (2.7), we attain

$$J_{\mathrm{MLS}}(p)(x) = \sum_{x_i \in \mathcal{N}(x)} \mathcal{W}(x - x_i)(f_i - \pi(x_i))^2. \tag{2.4}$$

Definition 2.2. Let a set of points $\mathcal{X}_N$ and associated patches $C_\Omega := \{\omega_i\}$ be given. Then we refer to the sets

$$\mathcal{N}_i := \{x_k \in \mathcal{X}_N \,|\, x_k \in \omega_i\} \tag{2.5}$$

and

$$C_i := \{\omega_k \in C_\Omega \,|\, \omega_k \cap \omega_i \neq \emptyset\}. \tag{2.6}$$

as local neighborhoods of a particular particle x_i or the respective patch ω_i. For an arbitrary point $x \in \overline{\Omega}$ we define its associated neighborhood as

$$\mathcal{N}(x) := \{x_k \in \mathcal{X}_N \,|\, x \in \omega_k\}. \tag{2.7}$$

The solution $\pi_{\mathrm{MLS}}(x)$ with $\pi_{\mathrm{MLS}} \in \mathcal{P}_k$ to the minimization of (2.4) is obtained by the linear system

$$G_{\mathrm{MLS}}(x)\tilde{u}_x = \hat{f}_{\mathrm{MLS}}(x) \quad \text{with} \quad (G_{\mathrm{MLS}}(x))_{q,r} := \sum_{x_i \in \mathcal{N}(x)} p_q(x_i)\mathcal{W}(x - x_i)p_r(x_i). \tag{2.8}$$

The vector $\hat{f}_{\mathrm{MLS}}(x) \in \mathbb{R}^{d^{\mathcal{P}^k}}$ on the right-hand side is defined as

$$\hat{f}_{\mathrm{MLS}}(x) := \Big(\sum_{x_i \in \mathcal{N}(x)} f_i \mathcal{W}(x - x_i) p_q(x_i) \Big)_{q=1}^{d^{\mathcal{P}^k}}.$$

The linear system (2.8) is uniquely solvable if the neighborhood $\mathcal{N}(x)$ is $\mathcal{P}_k$-unisolvent and we can define the moving least squares operator

$$\Big(A_{\mathrm{MLS}}\tilde{f}\Big)(x) = \pi_{\mathrm{MLS}}(x) = \sum_{q=1}^{d^{\mathcal{P}^k}} u_{x,q} p_q(x). \tag{2.9}$$

Note that in the classical least squares approximation we need to solve a *single* linear system (2.2) only to obtain the approximation π_{LS} on the complete domain Ω. The resulting polynomial π_{LS} can then be evaluated directly for all $x \in \Omega$, i.e., the coefficients $\tilde{u}_q$ of π_{LS} are independent of x. In the moving least squares approach, however, we need to solve a linear system for each point of evaluation $x \in \Omega$ to obtain the respective polynomial π_{MLS} which can then be evaluated at x only, i.e., the coefficients $\tilde{u}_x$ of π_{MLS} depend on x, compare (2.9). Let us summarize our findings so-far in the following theorem which generalizes the above setting slightly.

Theorem 2.1. *Let the set of points $\mathcal{X}_N = \{x_i \,|\, i = 1, \dots, N\}$, associated weight functions $W_i \in \mathcal{C}(\mathbb{R}^D, \mathbb{R}_0^+)$ such that $(\mathrm{supp}(W_i))^\circ = \omega_i$ and data $\tilde{f} = (f_i) \in \mathbb{R}^N$ be given. Assume that for $k \in \mathbb{N}_0$ the neighborhood associated with each $x \in \Omega$ defined in (2.7) is $\mathcal{P}_k$-unisolvent. Then the approximation*

$$\Big(A_{\mathrm{MLS}}\tilde{f}\Big)(x) = \pi_{\mathrm{MLS}}(x)$$

where $\pi_{\mathrm{MLS}} \in \mathcal{P}_k$ and $\pi_{\mathrm{MLS}}(x)$ is the solution of the minimization problem

$$\min_{\pi \in \mathcal{P}_k} J_{\mathrm{MLS}}(\pi)(x) = \min_{\pi \in \mathcal{P}_k} \sum_{x_i \in \mathcal{N}(x)} W_i(x)(f_i - \pi(x_i))^2 \tag{2.10}$$

is well-defined.

Proof. Consider an arbitrary but fixed point of evaluation $x^* \in \Omega$ and the respective minimization problem

$$\min_{\pi \in \mathcal{P}_k} J_{\mathrm{MLS}}(\pi)(x^*) = \min_{\pi \in \mathcal{P}_k} \sum_{x_i \in \mathcal{N}(x^*)} W_i(x^*)(f_i - \pi(x_i))^2.$$

The necessary condition $\delta J_{\mathrm{MLS}}(\pi_{\mathrm{MLS}}, \pi) = 0$ for all $\pi \in \mathcal{P}_k$ yields

$$\sum_{x_i \in \mathcal{N}(x^*)} \Big(f_i - \pi_{\mathrm{MLS}}(x_i)\Big) W_i(x^*)\pi(x_i) = 0 \quad \text{for all } \pi \in \mathcal{P}_k.$$

Choosing an arbitrary basis $\{p_q \,|\, q = 1, \dots, d^{\mathcal{P}^k}\}$ and setting

$$\pi_{\mathrm{MLS}}(x^*) = \sum_{q=1}^{d^{\mathcal{P}^k}} u_{x^*,q} p_q(x^*) \tag{2.11}$$

we obtain

$$\sum_{x_i \in \mathcal{N}(x^*)} \Big(f_i - \sum_{q=1}^{d^{\mathcal{P}^k}} u_{x^*,q} p_q(x_i) \Big) W_i(x^*) p_r(x_i) = 0 \quad \text{for all } r = 1, \dots, d^{\mathcal{P}^k} \tag{2.12}$$

which is equivalent to the matrix equation

$$G_{\mathrm{MLS}}(x^*) \tilde{u}_{x^*} = \hat{f}_{\mathrm{MLS}}(x^*) \tag{2.13}$$

with the system matrix $G_{\mathrm{MLS}}(x^*) = (G_{\mathrm{MLS}}(x^*)_{q,r}) \in \mathbb{R}^{d^{\mathcal{P}^k} \times d^{\mathcal{P}^k}}$ and the right-hand side vector $\hat{f}_{\mathrm{MLS}}(x^*) = \big(\hat{f}_{\mathrm{MLS}}(x^*)_q\big) \in \mathbb{R}^{d^{\mathcal{P}^k}}$ defined as

$$G_{\mathrm{MLS}}(x^*)_{q,r} := \sum_{x_i \in \mathcal{N}(x^*)} p_q(x_i) W_i(x^*) p_r(x_i), \quad \text{and} \quad \hat{f}_{\mathrm{MLS}}(x^*)_q := \sum_{x_i \in \mathcal{N}(x)} f_i W_i(x^*) p_q(x_i).$$

Recall that the unique solvability of (2.13) and the minimal property of $\pi_{\mathrm{MLS}}(x^*)$ follow from the positive definiteness of $G_{\mathrm{MLS}}(x^*)$. To this end, we consider for an arbitrary vector $\gamma \in \mathbb{R}^{d^{\mathcal{P}^k}}$ the scalar product

$$\gamma \cdot G_{\mathrm{MLS}}(x^*)\gamma = \sum_{x_i \in \mathcal{N}(x)} W_i(x^*) \Big(\sum_{q=1}^{d^{\mathcal{P}^k}} \gamma_q p_q(x_i) \Big)^2.$$

From the non-negativity of W_i follows $\gamma \cdot G_{\mathrm{MLS}}(x^*)\gamma \geq 0$ and hence G_{MLS} is positive semi-definite. Now let us assume that there is a particular $\gamma \neq 0$ such that $\gamma \cdot G_{\mathrm{MLS}}(x^*)\gamma = 0$. Since $x_i \in \mathcal{N}(x^*)$ we have $x^* \in \omega_i = (\mathrm{supp}(W_i))^\circ$ and due to the smoothness of the weight functions W_i we have $W_i(x^*) > 0$. Therefore, $\gamma_q p_q(x_i) = 0$ for all $q = 1, \dots, d^{\mathcal{P}^k}$ and $x_i \in \mathcal{N}(x^*)$. The unisolvence of $\mathcal{N}(x^*)$ implies that $\gamma = 0$ which contradicts the assumption $\gamma \neq 0$ and hence we conclude that $G_{\mathrm{MLS}}(x^*)$ is positive definite. □

Recall that in the classical least squares approach the minimizer $\pi_{\mathrm{LS}} \in \mathcal{P}^k$ is a global polynomial. It can be represented via the basis $\langle p_q \rangle$ of $\mathcal{P}^k$. In the moving least squares approach however only the value $\pi_{\mathrm{MLS}}(x)$ at the current point of evaluation $x \in \Omega$ can be expressed via $\langle p_q \rangle$. So far we have no representation of π_{MLS} as a function. Hence, the question is if we can construct appropriate basis functions ϕ_i such that $\pi_{\mathrm{MLS}} \in \mathrm{span}\langle \phi_i \rangle$.

Corollary 2.1. *Let the assumptions of Theorem 2.1 be satisfied. Then the representation formula*

$$\Big(A_{\mathrm{MLS}} \tilde{f} \Big)(x) = \sum_{i=1}^{N} f_i \phi_i(x) \tag{2.14}$$

holds and the basis functions ϕ_i are given by

$$\phi_i(x) := W_i(x) \sum_{q=1}^{d^{\mathcal{P}^k}} \alpha_{x,q} p_q(x_i) \tag{2.15}$$

where the coefficient vector $\alpha_x = (\alpha_{x,q})$ is the unique solution of the linear system

$$G_{\mathrm{MLS}}(x)\alpha_x = p(x) = (p_q(x))_{q=1}^{d_k^{\mathcal{P}}}. \tag{2.16}$$

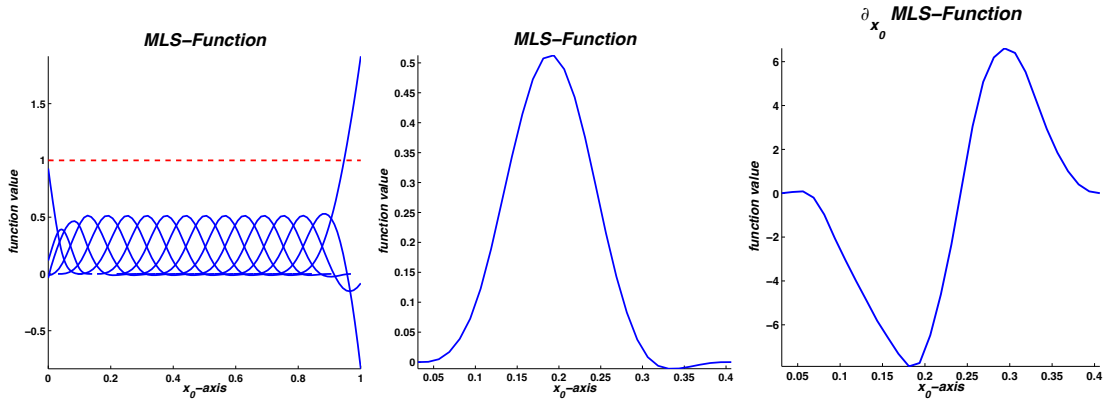


Figure 2.1. *Moving least squares basis functions in one dimension using a cubic spline weight function (left: all basis functions; center: single basis function; right: first derivative).*

Proof. Again, consider a fixed but arbitrary point of evaluation $x^* \in \Omega$. According to (2.9) and (2.13) we have

$$\left(A_{\mathrm{MLS}}\tilde{f}\right)(x^*) = \sum_{q=1}^{d^{\mathcal{P}^k}} \tilde{u}_{x^*,q} p_q(x^*).$$

With the equivalence (2.16) this yields

$$\left(A_{\mathrm{MLS}}\tilde{f}\right)(x^*) = \tilde{u}(x^*) \cdot G_{\mathrm{MLS}}(x^*)\alpha_{x^*} = \sum_{q=1}^{d^{\mathcal{P}^k}} u_{x^*,q} \sum_{x_i \in \mathcal{N}(x^*)} W_i(x^*) p_q(x_i) \sum_{r=1}^{d^{\mathcal{P}^k}} p_r(x_i)\alpha_{x^*,r}.$$

Rearranging the sums we obtain

$$\left(A_{\mathrm{MLS}}\tilde{f}\right)(x^*) = \sum_{r=1}^{d^{\mathcal{P}^k}} \alpha_{x^*,r} \sum_{x_i \in \mathcal{N}(x^*)} W_i(x^*) \sum_{q=1}^{d^{\mathcal{P}^k}} u_{x^*,q} p_q(x_i) p_r(x_i). \tag{2.17}$$

Plugging (2.12) into (2.17) gives

$$\left(A_{\mathrm{MLS}}\tilde{f}\right)(x^*) = \sum_{r=1}^{d^{\mathcal{P}^k}} \alpha_{x^*,r} \sum_{x_i \in \mathcal{N}(x^*)} W_i(x^*) f_i p_r(x_i) = \sum_{x_i \in \mathcal{N}(x^*)} f_i W_i(x^*) \sum_{r=1}^{d^{\mathcal{P}^k}} \alpha_{x^*,r} p_r(x_i)$$

and with definition (2.15) this yields the asserted representation (2.14), compare Figures 2.1 and 2.2. □

Remark 2.1. Note that the coefficient vector α_x of (2.16) is independent of the particle x_i and hence α_x is *identical* for all particles $x_i \in \mathcal{N}(x)$; i.e. for all respective basis functions ϕ_i. Thus the evaluation of a single basis function ϕ_i at a particular point $x \in \Omega$ is of similar complexity as the simultaneous evaluation of all non-vanishing basis functions ϕ_j in $x \in \Omega$.

Due to (2.15) we can view the moving least squares technique as a constructive approach to obtain compactly supported shape functions ϕ_i with $\mathrm{supp}(\phi_i) = \mathrm{supp}(W_i)$ from scattered independent points $x_i \in \mathcal{X}_N$ only; i.e., an approach for the construction of meshfree shape functions.

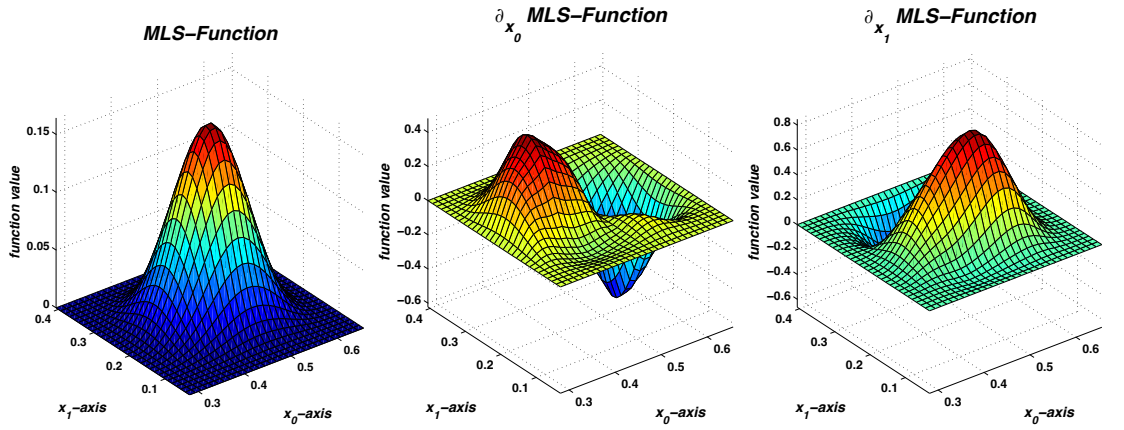


Figure 2.2. *Moving least squares basis function (left) in two dimensions and its partial derivatives (center and right) using a cubic spline weight function.*

Since we are ultimately interested in the development of a meshfree Galerkin method for the numerical treatment of partial differential equations we must be concerned with the regularity of the basis functions (2.15).

Lemma 2.1. *Let the assumptions of Theorem 2.1 be satisfied with $W_i \in \mathcal{C}^r(\mathbb{R}^D)$ with $r > 0$ for all $i = 1, \ldots, N$. Then, there holds $\phi_i \in \mathcal{C}^r(\mathbb{R}^D)$ for the basis functions ϕ_i of (2.15).*

The second important property we must consider is the consistency of our moving least squares functions (2.15).

Lemma 2.2. *Let the assumptions of Theorem 2.1 be satisfied. Then, the composed operator*

$$A_{\mathrm{MLS}} E_{\mathcal{X}_N} : \mathcal{C}(\Omega) \to \operatorname{span}\langle \phi_i \,|\, i = 1, \ldots, N \rangle \subset \mathcal{C}(\Omega)$$

with ϕ_i defined in (2.15), $E_{\mathcal{X}_N} : \mathcal{C}(\Omega) \to \mathbb{R}^N$ denotes the point evaluation, i.e. $E_{\mathcal{X}_N}(u) = (u(x_i))_{i=1}^N$, reproduces all polynomials $\pi \in \mathcal{P}_k(\Omega)$, i.e.

$$A_{\mathrm{MLS}} E_{\mathcal{X}_N}|_{\mathcal{P}_k(\Omega)} = \mathbb{I}, \quad A_{\mathrm{MLS}} E_{\mathcal{X}_N}(\pi) = \pi \quad \text{for all } \pi \in \mathcal{P}_k(\Omega). \tag{2.18}$$

Proof. Recall the representation (2.14)

$$\Big(A_{\mathrm{MLS}} E_{\mathcal{X}_N}(\pi)\Big)(x) = \sum_{i=1}^{N} \pi(x_i)\phi_i(x).$$

The polynomial π has a unique representation in the employed basis for $\mathcal{P}_k(\Omega)$, i.e.

$$\sum_{q=1}^{d^{\mathcal{P}^k}} \gamma_q p_q(x) = \pi(x)$$

for all x. With the representation of the basis (2.15) this yields

$$\Big(A_{\mathrm{MLS}} E_{\mathcal{X}_N}(\pi)\Big)(x) = \sum_{i=1}^{N} \sum_{q=1}^{d^{\mathcal{P}^k}} \gamma_q p_q(x_i)\phi(x) = \sum_{i=1}^{N} \sum_{q=1}^{d^{\mathcal{P}^k}} \gamma_q p_q(x_i) W_i(x) \sum_{r=1}^{d^{\mathcal{P}^k}} \alpha_{x,r} p_r(x_i).$$

Rearranging the sums, we obtain

$$\Big(A_{\mathrm{MLS}}E_{\mathcal{X}_N}(\pi)\Big)(x) = \sum_{q=1}^{d^{\mathcal{P}^k}}\sum_{r=1}^{d^{\mathcal{P}^k}}\sum_{i=1}^{N}\gamma_q p_q(x_i)W_i(x)p_r(x_i)\alpha_{x,r} = \gamma\cdot G_{\mathrm{MLS}}(x)\alpha_x.$$

Since (2.16) holds, we obtain the asserted equivalence

$$\Big(A_{\mathrm{MLS}}E_{\mathcal{X}_N}(\pi)\Big)(x) = \gamma\cdot p(x) = \sum_{q=1}^{d^{\mathcal{P}^k}}\gamma_q p_q(x) = \pi(x) \text{ for all } x\in\Omega.$$

□

An immediate consequence of this lemma is that the basis functions ϕ_i constructed by the moving least squares approach are a *partition of unity* independent of the employed polynomial degree $k\in\mathbb{N}_0$.

Corollary 2.2. *Let the assumptions of Theorem 2.1 be satisfied. Then, the basis functions defined in (2.15) satisfy*

$$\sum_{i=1}^{N}\phi_i(x) = 1 \quad \textit{for all } x\in\Omega. \tag{2.19}$$

Yet, the basis functions ϕ_i in general do *not* satisfy the Kronecker property, i.e.

$$\phi_i(x_j) \neq \delta_{ij} = \begin{cases} 1 & i=j,\\ 0 & i\neq j,\end{cases}$$

compare Figures 2.1 and 2.2. Note also that the assumption of the $\mathcal{P}_k$-unisolvence of $\mathcal{N}(x)$ for $k\geq 0$ and all $x\in\Omega$ is not trivial to verify for arbitrary point sets $\mathcal{X}_N$, i.e. a specific choice of W_i for $i=1,\dots,N$. Thus, the selection of appropriate supports ω_i is in general a somewhat challenging task. Observe though that for the important special case of $k=0$, i.e. an approximation with the constant function, there holds the equivalence

$$\mathcal{N}(x) \text{ is } \mathcal{P}^0\text{-unisolvent} \quad\Longleftrightarrow\quad \Omega\subset\bigcup_{i=1}^{N}\omega_i.$$

In this case (2.15) reduces to

$$\phi_i(x) = W_i(x)\alpha_{x,0}, \quad \text{with } G_{\mathrm{MLS}0,0}\alpha_{x,0} = 1 \tag{2.20}$$

if we choose $p_0\equiv 1$ as the basis of $\mathcal{P}^0$. Plugging the definition (2.8) into (2.20) we obtain the *explicit* representation

$$\phi_i(x) = \frac{W_i(x)}{S(x)}, \quad \text{with } S(x) := \sum_{j=1}^{N}W_j(x)$$

whereas in general with $k>0$ the basis functions ϕ_i of (2.15) are known *implicitly* only via (2.16). Due to the compact support of the weights W_i and the observation that $\mathcal{N}(x)\subset C_i$ for all $x\in\omega_i$, compare (2.6) and Definition 2.2, we can rewrite the moving least squares function for $k=0$ as

$$\phi_i(x) = \frac{W_i(x)}{S_i(x)}, \quad \text{with } S_i(x) := \sum_{\omega_j\in C_i}W_j(x). \tag{2.21}$$

Observe that these function, the so-called Shepard functions, satisfy Corollary 2.2. The Shepard functions are a partition of unity.

Finally, we need to consider the stability of the evaluation of the basis functions (2.15); i.e. the conditioning of the system matrix $G_{\mathrm{MLS}}(x^*)$ for all $x^* \in \Omega$. Note that the choice of the polynomial basis in the proof of Theorem 2.1 is arbitrary for each point of evaluation x^*. Hence, we can for instance consider linear transformations $T_{x^*} : \mathbb{R}^D \to \mathbb{R}^D$ depending on the point of evaluation x^* of a fixed basis $\{\hat{p}_q \,|\, q = 1, \dots, d^{\mathcal{P}^k}\}$, i.e.,

$$T_{x^*} : x \mapsto \frac{x^* - x}{\rho_{x^*}}, \quad p_{x^*,q}(x) = \hat{p}_q \circ T_{x^*}(x) = \hat{p}_q\Big(\frac{x^* - x}{\rho_{x^*}}\Big). \tag{2.22}$$

The approximation (2.9) and hence the representation (2.14) and the basis functions (2.15) are unchanged, but the respective linear systems G_{MLS} and right-hand side vectors need to be modified.

Corollary 2.3. *Consider the choice of basis (2.22) for the solution of the pointwise minimization problem (2.10) at a fixed but arbitrary point $x^* \in \Omega$. Then, the solution $\pi_{\mathrm{MLS}}(x^*)$ is obtained by the solution of the linear system (2.13) with the system matrix $G_{\mathrm{MLS}}(x^*) = (G_{\mathrm{MLS}}(x^*)_{q,r}) \in \mathbb{R}^{d^{\mathcal{P}^k} \times d^{\mathcal{P}^k}}$*

$$G_{\mathrm{MLS}}(x^*)_{q,r} := \sum_{x_i \in \mathcal{N}(x^*)} p_{x^*,q}(x_i) W_i(x^*) p_{x^*,r}(x_i) = \sum_{x_i \in \mathcal{N}(x^*)} \hat{p}_q\Big(\frac{x^* - x_i}{\rho_{x^*}}\Big) W_i(x^*) \hat{p}_r\Big(\frac{x^* - x_i}{\rho_{x^*}}\Big)$$

and the right-hand side

$$\hat{f}_{\mathrm{MLS}}(x^*)_q := \sum_{x_i \in \mathcal{N}(x^*)} f_i W_i(x^*) p_{x^*,q}(x_i) = \sum_{x_i \in \mathcal{N}(x^*)} f_i W_i(x^*) \hat{p}_q\Big(\frac{x^* - x_i}{\rho_{x^*}}\Big).$$

The basis functions ϕ_i are given by

$$\phi_i(x^*) := W_i(x^*) \sum_{q=1}^{d^{\mathcal{P}^k}} \alpha_{x^*,q} p_{x^*,q}(x_i) = W_i(x^*) \sum_{q=1}^{d^{\mathcal{P}^k}} \alpha_{x^*,q} \hat{p}_q\Big(\frac{x^* - x_i}{\rho_{x^*}}\Big) \tag{2.23}$$

where the coefficient vector $\alpha_{x^*} = (\alpha_{x^*,q})$ is the unique solution of the linear system

$$G_{\mathrm{MLS}}(x^*)\alpha(x^*) = p_{x^*}(x^*) = \hat{p}\Big(\frac{x^* - x^*}{\rho_{x^*}}\Big) = \hat{p}(0). \tag{2.24}$$

The advantage of (2.23) over (2.15) in computations is that the condition number of the system matrix $G_{\mathrm{MLS}}(x^*)$ can be controlled via the parameter ρ_{x^*} whereas (2.8) can become unstable when we use more and more points $x_i \in \mathcal{X}_N$. Observe that such a refinement of the point set $\mathcal{X}_N$ does not assume any connectivity among the points x_i. The insertion of new points x_i into $\mathcal{X}_N$ and thereby an *h-adaptive* refinement of the respective meshfree function space $V_{\mathrm{MLS}} := \mathrm{span}\langle\phi_i\rangle$ is straightforward. Unfortunately this is not the case for a local *p-adaptive* refinement.

Recall that the weight functions W_i can be chosen arbitrarily on each ω_i; i.e., they are independent of each other and more importantly independent of the point of evaluation x^*. Hence, we can consider the linear transformations of a window function

$$T_i(x) : x \mapsto \frac{x - x_i}{\rho_i}, \quad W_i(x) := \mathcal{W}\Big(\frac{x - x_i}{\rho_i}\Big)$$

as weight functions. Then, (2.23) becomes

$$\phi_i(x^*) := \mathcal{W}\Big(\frac{x^* - x_i}{\rho_i}\Big) \sum_{q=1}^{d^{\mathcal{P}^k}} \alpha_{x^*,q} \hat{p}_q\Big(\frac{x^* - x_i}{\rho_{x^*}}\Big). \tag{2.25}$$

Note the difference in the scaling of the window function (using $1/\rho_i$) and the employed polynomial (scaled by $1/\rho_{x^*}$). This is due to the fact that the polynomial basis can be chosen with respect to x^* and is evaluated at x_i, whereas the weight functions can be chosen with respect to x_i and are evaluated at x^*, compare (2.3) and (2.10). From this observation it is clear that the moving least squares approach does not support a p-adaptive approximation, i.e., the variation of the polynomial degree k on each ω_i. We can change the polynomial degree k only with respect to the point of evaluation x^*. Yet, such an approach suggests a disjoint partition of the domain Ω into $\Omega_l = \{x^* \in \Omega \,|\, k(x^*) = k_l\}$ with $k_l \in \mathbb{N}_0$ and it is clear that at the boundaries of these disjoint sub-regions, i.e., $x \in \partial\Omega_{l_1} \cap \partial\Omega_{l_2}$ with $l_1 \neq l_2$, the variation in the polynomial degree may lead to a jump in the resulting approximation $A_{\text{MLS}}(x)\tilde{f}$ and the associated basis functions ϕ_i.

Thus, we can use the moving least squares approach to construct meshfree functions spaces V_{MLS} that support h-adaptive refinement easily but do they not allow for p-adaptive refinement without compromising the regularity of the shape functions. The capability of p-adaptive refinement of our meshfree function space must be provided by an additional construction outside of the moving least squares approach.

2.3 Local Enrichment

The partition of unity property (2.19) of the MLS basis functions ϕ_i defined in (2.23) can be utilized to enhance the approximation properties of the associated MLS function space

$$V_{\text{MLS}} := \text{span}\langle \phi_i \rangle \tag{2.26}$$

and more importantly we can define an enriched version $V_{\text{MLS}}^{\text{LE}}$ of V_{MLS} that allows for a p-adaptive approximation approach, see e.g. [41, 128].

Let us consider the MLS basis functions ϕ_i and a collection of sufficiently regular local approximation spaces

$$V_i(\omega_i, \mathbb{R}) := \text{span}\langle \vartheta_i^n \rangle \tag{2.27}$$

with $1 \in V_i(\omega_i, \mathbb{R})$ for all $i = 1, \dots, N$. Then, the space

$$V_{\text{MLS}}^{\text{LE}} := \sum_{i=1}^{N} \phi_i V_i = \text{span}\langle \phi_i \vartheta_i^n \rangle \supset V_{\text{MLS}} \tag{2.28}$$

obviously contains V_{MLS} and hence enjoys at least the approximation properties of V_{MLS}. Moreover, the local approximation spaces are completely arbitrary and independent of each other since the global regularity of the product functions is inherited from the ϕ_i. Thus, if we choose V_i as local polynomials of degree p_i on ω_i the space $V_{\text{MLS}}^{\text{LE}}$ of (2.28) can reproduce higher order polynomials in some parts of the domain than in others. Therefore, $V_{\text{MLS}}^{\text{LE}}$ can be used in a p-adaptive setting — unlike the space V_{MLS}.

On the other hand, if we choose the local approximation spaces V_i such that their basis functions resolve particular singularities we can make use of this enrichment approach to avoid a more expensive approximation of singular functions by h-adaptive refinement.

In summary, the concept of local enrichment based on a partition of unity can substantially improve the approximation properties and improves the cost-efficiency of the respective numerical method dramatically. Not surprisingly, this technique is employed in many meshfree e.g. [19, 109, 128] and mesh-based methods e.g. [7, 16, 20, 42, 44, 45, 100, 143]. Rather than focusing on the particular differences of the realization of the presented enrichment concept in these methods we refer to the abstract technique as a partition of unity method [8, 9] (PUM) since the PU property of the ϕ_i is the only necessary assumption for the presented approach and regard all of the above methods as special instances of the PUM. The mathematical foundation of the PUM and its particularities are the subject of the next chapter.

Chapter 3

Partition of Unity Method

The notion of a partition of unity method (PUM) was coined in [8, 9] and is based on the special finite element methods developed in [7]. The abstract ingredients of a PUM are

- a partition of unity (PU) $\{\varphi_i \,|\, i = 1, \ldots, N\}$ with

$$\varphi_i \in \mathcal{C}^r(\mathbb{R}^D, \mathbb{R}) \quad \text{and patches} \quad \omega_i := \operatorname{supp}^\circ(\varphi_i),$$

- a collection of local approximation spaces

$$V_i(\omega_i, \mathbb{R}) := \operatorname{span}\langle \vartheta_i^n \rangle \tag{3.1}$$

  defined on the patches ω_i for $i = 1, \ldots, N$.

With these two ingredients we define the PUM space

$$V^{\mathrm{PU}} := \sum_{i=1}^{N} \varphi_i V_i = \operatorname{span}\langle \varphi_i \vartheta_i^n \rangle; \tag{3.2}$$

i.e., the shape functions of a PUM space are simply defined as the products of the PU functions φ_i and the local approximation functions ϑ_i^n. The PU functions provide the locality and global regularity of the product functions whereas the functions ϑ_i^n equip V^{PU} with its approximation power. Thus, we refer to the PU functions φ_i also as h-components and denote the local approximation functions ϑ_i^n also as p-components of the PUM space V^{PU}.

3.1 Properties and Approximation

To study the approximation properties of the PUM space V^{PU} we need to introduce some notation and specific assumptions on the PU and the local approximation spaces, see also [9, 98].

Definition 3.1 (Partition of Unity)**.** Let $\Omega \subset \mathbb{R}^D$ be an open set. Let $\{\varphi_i \,|\, i = 1, \ldots, N\}$ be a collection of N Lipschitz functions with

$$\begin{aligned} 0 \;\leq\; \varphi_i(x) \;&\leq\; 1, & \sum_{i=1}^{N} \varphi_i \;&\equiv\; 1 \text{ on } \overline{\Omega}, \\ \|\varphi_i\|_{L^\infty(\mathbb{R}^D)} \;&\leq\; C_\infty, & \|\nabla \varphi_i\|_{L^\infty(\mathbb{R}^D)} \;&\leq\; \frac{C_\nabla}{\operatorname{diam}(\omega_i)}, \end{aligned}$$

where $\omega_i := \mathrm{supp}(\varphi_i)^\circ$ is a Lipschitz domain, C_∞ and C_∇ are two positive constants. The collection of functions $\{\varphi_i \,|\, i = 1, \dots, N\}$ is referred to as a partition of unity (PU) and the PU is said to be of degree $k \in \mathbb{N}_0$ if $\varphi_i \in \mathcal{C}^k(\mathbb{R}^D)$ and $\|\nabla^k \varphi_i\|_{L^\infty(\mathbb{R})} \leq \frac{C_{\nabla^k}}{\mathrm{diam}(\omega_i)}$ for all $i = 1, \dots, N$. The sets ω_i are called patches and their collection is referred to as a cover $C_\Omega := \{\omega_i \,|\, i = 1, \dots, N\}$ of the domain Ω.

For PUM spaces (3.2) which employ a PU $\{\varphi_i\}$ satisfying Definition 3.1 there hold the following error estimates due to [9].

Theorem 3.1. *Let $\Omega \subset \mathbb{R}^D$ be a Lipschitz domain. Let $\{\varphi_i\}$ be a partition of unity according to Definition 3.1. Let us further introduce the covering index $\lambda_{C_\Omega} : \Omega \to \mathbb{N}$ such that*

$$\lambda_{C_\Omega}(x) := \mathrm{card}(\{\omega_i \in C_\Omega \,|\, x \in \omega_i\}) \tag{3.3}$$

and let us assume that $\lambda_{C_\Omega}(x) \leq M \in \mathbb{N}$ for all $x \in \Omega$. Let a collection of local approximation spaces $V_i = \mathrm{span}\langle \vartheta_i^n \rangle \subset H^1(\omega_i)$ be given. Let $u \in H^1(\Omega)$ be the function to be approximated. Assume that the local approximation spaces V_i have the following approximation properties: On each patch $\Omega \cap \omega_i$, the function u can be approximated by a function $u_i \in V_i$ such that

$$\|u - u_i\|_{L^2(\Omega\cap\omega_i)} \leq \hat{\epsilon}_i, \quad \textit{and} \quad \|\nabla(u - u_i)\|_{L^2(\Omega\cap\omega_i)} \leq \tilde{\epsilon}_i \tag{3.4}$$

hold for all $i = 1, \dots, N$. Then the function

$$u^{\mathrm{PU}} := \sum_{i=1}^N \varphi_i u_i \in V^{\mathrm{PU}} \subset H^1(\Omega)$$

satisfies the global estimates

$$\|u - u^{\mathrm{PU}}\|_{L^2(\Omega)} \leq \sqrt{M} C_\infty \Big(\sum_{i=1}^N \hat{\epsilon}_i^2 \Big)^{1/2}, \tag{3.5}$$

$$\|\nabla(u - u^{\mathrm{PU}})\|_{L^2(\Omega)} \leq \sqrt{2M} \Big(\sum_{i=1}^N \big(\frac{C_\nabla}{\mathrm{diam}(\omega_i)} \big)^2 \hat{\epsilon}_i^2 + C_\infty^2 \tilde{\epsilon}_i^2 \Big)^{1/2}. \tag{3.6}$$

Proof. There holds $V^{\mathrm{PU}} \subset H^1(\Omega)$ since the local approximation spaces $V_i \subset H^1(\omega_i)$ can be extended to the complete domain Ω due to the fact that the patches are Lipschitz domains.

With the PU property $\sum_{i=1}^N \varphi_i \equiv 1$ on Ω there holds

$$u(x) - \sum_{i=1}^N \varphi_i(x) u_i(x) = \sum_{i=1}^N \varphi_i(x)(u(x) - u_i(x)).$$

Since the covering index λ_{C_Ω} is uniformly bounded by M for all $x \in \Omega$ there are no more than M non-vanishing terms in these sums. With the help of the inequality

$$\big(\sum_{j=1}^M |a_j| \big)^2 \leq M \sum_{j=1}^M |a_j|^2 \quad \text{for all } M < \infty$$

we can bound

$$\big| \sum_{i=1}^N \varphi_i(x)(u(x) - u_i(x)) \big|^2 \leq M \sum_{i=1}^N |\varphi_i(x)(u(x) - u_i(x))|^2$$

for all $x \in \Omega$. With the bounds of Definition 3.1 this yields the estimate

$$\|u - u^{\mathrm{PU}}\|^2_{L^2(\Omega)} \leq M \sum_{i=1}^{N} \|\varphi_i(u - u_i)\|^2_{L^2(\omega_i \cap \Omega)} \leq M C_\infty^2 \sum_{i=1}^{N} \|u - u_i\|^2_{L^2(\omega_i \cap \Omega)} \leq M C_\infty^2 \sum_{i=1}^{N} \hat{\epsilon}_i^2.$$

Analogously, we obtain the bounds for the gradient of the error

$$\begin{aligned} \|\nabla(u - u^{\mathrm{PU}})\|^2_{L^2(\Omega)} &\leq 2M\Big(\sum_{i=1}^{N} \|(\nabla\varphi)(u - u_i)\|^2_{L^2(\omega_i \cap \Omega)} + \sum_{i=1}^{N} \|\varphi \nabla(u - u_i)\|^2_{L^2(\omega_i \cap \Omega)} \Big) \\ &\leq 2M\Big(C_\infty^2 \sum_{i=1}^{N} \tilde{\epsilon}_i^2 + \sum_{i=1}^{N} \Big(\frac{C_\nabla}{\operatorname{diam}(\omega_i)} \Big)^2 \hat{\epsilon}_i^2 \Big) \end{aligned}$$

□

The estimates (3.5) and (3.6) show that the global error is of the same order as the local errors provided that the covering index is bounded independent of N, i.e. $M = O(1)$.

If we for instance assume that $\operatorname{diam}(\omega_i) \approx h$ for all $i = 1, \dots, N$ and the local spaces V_i contain polynomials of degree p for all $i = 1, \dots, N$ then the spaces V_i satisfy the local error bounds

$$\begin{aligned} \|u - u_i\|_{L^2(\Omega \cap \omega_i)} &\leq C\, h^{p+1} \|u\|_{H^k(\Omega \cap \omega_i)} &=:\ \hat{\epsilon}_i, \\ \|\nabla(u - u_i)\|_{L^2(\Omega \cap \omega_i)} &\leq C\, h^{p} \|u\|_{H^k(\Omega \cap \omega_i)} &=:\ \tilde{\epsilon}_i \end{aligned}$$

for $u \in H^k(\Omega)$ with $k \geq 1$ and $p \leq k - 1$ due to the Bramble–Hilbert-Lemma. Then, the estimates (3.5) and (3.6) become

$$\|u - u^{\mathrm{PU}}\|^2_{L^2(\Omega)} \leq \sqrt{M} C_\infty\, C\, h^{p+1} \|u\|_{H^k(\Omega)}$$

and

$$\|\nabla(u - u^{\mathrm{PU}})\|^2_{L^2(\Omega)} \leq \sqrt{2M(C_\infty + C_\nabla)}\, C\, h^{p} \|u\|_{H^k(\Omega)}$$

which correspond to the classical finite element estimates of a uniform h-version.

Note however that Theorem 3.1 is an abstract approximation result only and involves a specific choice of the approximation u^{PU}. It does not state that this approximation is the best-approximation in V^{PU} nor the uniqueness of the representation $\sum_{i=1}^{N} \varphi_i u_i$. In fact the above assumption are *not* sufficient to ensure the uniqueness of the representation $u^{\mathrm{PU}} = \sum_{i=1}^{N} \varphi_i u_i$; i.e., the shape functions $\varphi_i \vartheta_i^n$ may be linearly dependent in the PUM with the assumptions of Theorem 3.1.

To this end, consider the following simple one-dimensional situation: Let $\Omega = (0, 1)$ be the unit interval and define

$$\varphi_0(x) := 1 - x, \quad \text{and} \quad \varphi_1(x) := x.$$

Furthermore, we define the local approximation spaces

$$V_0 := \operatorname{span}\langle 1, x \rangle, \quad \text{and} \quad V_1 := \operatorname{span}\langle 1, x \rangle.$$

Obviously, the choice of components satisfies the assumptions above, however, the resulting PUM shape functions are linearly dependent. Observe that the PU functions φ_0 and φ_1 are linear polynomials as are all local approximation functions ϑ_i^n. Hence, the four product functions $\varphi_i \vartheta_i^n$ are the quadratic polynomials

$$(1 - x), x(1 - x), x, x^2.$$

Yet, there exist only three linearly independent quadratic polynomials on the interval Ω so that the four shape functions of our PUM space $V^{\mathrm{PU}} = \varphi_0 V_0 + \varphi_1 V_1$ must be linearly dependent. On

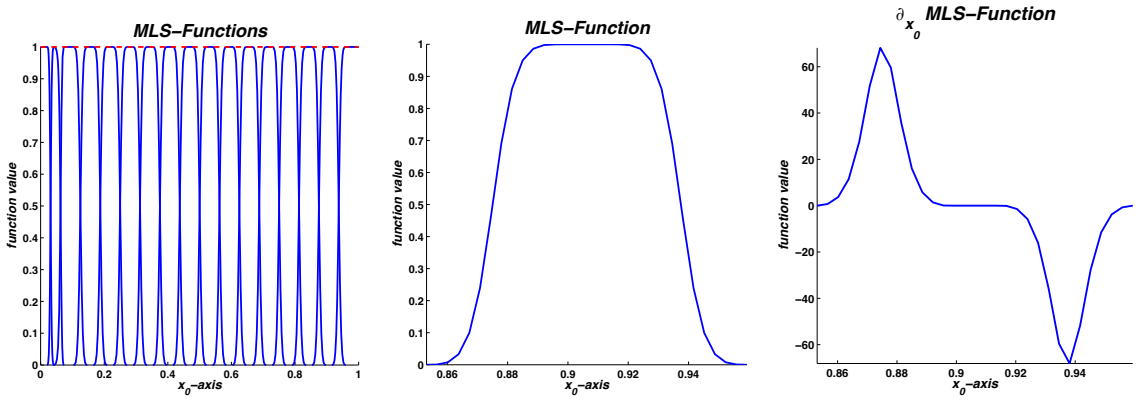


Figure 3.1. *Shepard functions satisfying the flat top property using a cubic weight function (left: all Shepard functions; center: single Shepard function; right: first derivative).*

the other hand, the space $V^{\mathrm{PU}} = \mathcal{P}^2$ obviously has better reproduction properties than the local spaces $V_i = \mathcal{P}^1$ so that better global error bounds than those of Theorem 3.1 can be attained.

To overcome this problem of linearly dependent shape functions which is sometimes referred to as the *nullity* of the PUM we introduce an additional assumption: the so-called flat top property, see also Figures 3.1 and 3.2.

Definition 3.2 (Flat top property)**.** Let $\{\varphi_i, | \, i = 1, \ldots, N\}$ be a partition of unity according to Definition 3.1. Let us define the sub-patches $\omega_{\mathrm{FT,i}} \subset \omega_i$ such that $\varphi_i|_{\omega_{\mathrm{FT,i}}} \equiv 1$. Then, the PU is said to have the flat top property, if there exists a constant C_{FT} such that for all patches $\omega_i = \mathrm{supp}^\circ(\varphi_i)$

$$\mu(\omega_i) \leq C_{\mathrm{FT}} \mu(\omega_{\mathrm{FT,i}}) \tag{3.7}$$

where $\mu(A)$ denotes the Lebesgue measure of $A \subset \mathbb{R}^D$. We have $C_\infty = 1$ for a PU satisfying (3.7).

Remark 3.1. The PU concept is employed in many meshfree methods. However, in most cases very smooth PU functions $\varphi_i \in C^k(\Omega)$ with $k \geq 2$ are used and the functions φ_i have rather large supports ω_i which overlap extensively. Hence in most meshfree methods $\mathrm{card}(C_i)$ and M are very large and the employed PU does *not* have the flat top property. This makes it easier to control $\|\nabla \varphi_i\|_{L^\infty}$, compare Definition 3.1, (3.6), and Figures 2.1 and 3.1, but it can lead to ill-conditioned and even singular stiffness matrices.

For a PU that satisfies the flat top property[1] we obtain the equivalence

$$\sum_{i=1}^{N} \varphi_i \sum_{n=1}^{\dim(V_i)} u_i^n \vartheta_i^n \equiv 0 \quad \Longleftrightarrow \quad \sum_{n=1}^{\dim(V_i)} u_i^n \vartheta_i^n \equiv 0 \text{ for all } i = 1, \ldots, N$$

which essentially states that the PUM space V^{PU} is not just a weighted sum of the local approximation spaces V_i but it is a *direct* sum; i.e.

$$V^{\mathrm{PU}} = \sum_{i=1}^{N} \varphi_i V_i = \bigoplus_{i=1}^{N} \varphi_i V_i.$$

Therefore, the product functions $\varphi_i \vartheta_i^n$ inherit the linear independence of the local approximation functions ϑ_i^n and we obtain the stability of the approximation.

[1] Note that the flat top property is a sufficient condition only for the linear independence of the shape functions in a PUM.

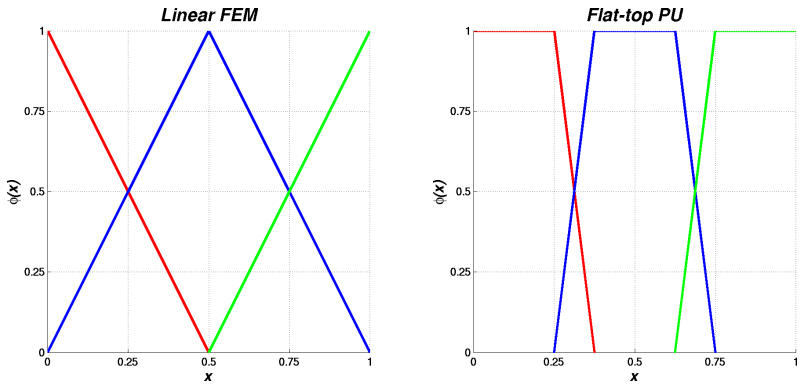


Figure 3.2. *A partition of unity obtained by classical linear finite elements (left) and a partition of unity which satisfies the flat top property (right).*

Remark 3.2. Note that for a PU that satisfies Definition 3.2 the estimates of Theorem 3.1 are of optimal order since the products $\varphi_i \vartheta_i^m$ of the local approximation functions ϑ_i^m with the PU functions φ_i agree with ϑ_i^m on $\omega_{\mathrm{FT},\mathrm{i}}$. Therefore, the approximation order of the global space V^{PU} is limited by the approximation properties of the local approximation spaces V_i. We do not encounter higher reproduction properties as in the above example but attain a linearly independent set of product functions.

The main challenge in a PUM is the construction of a PU that satisfies Definition 3.1 and Definition 3.2. To this end let us introduce the notion of an admissible cover.

Definition 3.3 (Admissible Cover)**.** Let $\Omega \subset \mathbb{R}^D$ be an open set. Let $\omega_i \subset \mathbb{R}^D$ be open sets with $\Omega \cap \omega_i \neq \emptyset$ for $i = 1, \ldots, N$. The collection $C_\Omega := \{\omega_i \,|\, i = 1, \ldots, N\}$ is called an admissible cover of Ω and the sets ω_i are denoted admissible cover patches if the following conditions are satisfied.

- Global covering:
$$\overline{\Omega} \subset \bigcup_{i=1}^{N} \omega_i.$$
- Minimal overlap: There exists a constant C_{FT} such that
$$\mu(\omega_i) \leq C_{\mathrm{FT}} \mu(\{x \in \omega_i \,|\, \lambda_{C_\Omega}(x) = 1\}). \tag{3.8}$$
- Bounded overlap: There exists a constant $M > 0$ such that for any $x \in \Omega$ there holds
$$\|\lambda_{C_\Omega}\|_{L^\infty(\Omega)} < M \ll N. \tag{3.9}$$
- Sufficient overlap: There exists a constant $C_S > 0$ such that for any $x \in \Omega$ there is at least one cover patch ω_i such that $x \in \omega_i$ and
$$\operatorname{dist}(x, \partial\omega_i) \geq C_S \operatorname{diam}(\omega_i). \tag{3.10}$$
- Comparability of neighboring patches: A subset
$$C_i := \{\omega_j \in C_\Omega \,|\, \omega_j \cap \omega_i \neq \emptyset\} \subset C_\Omega \tag{3.11}$$

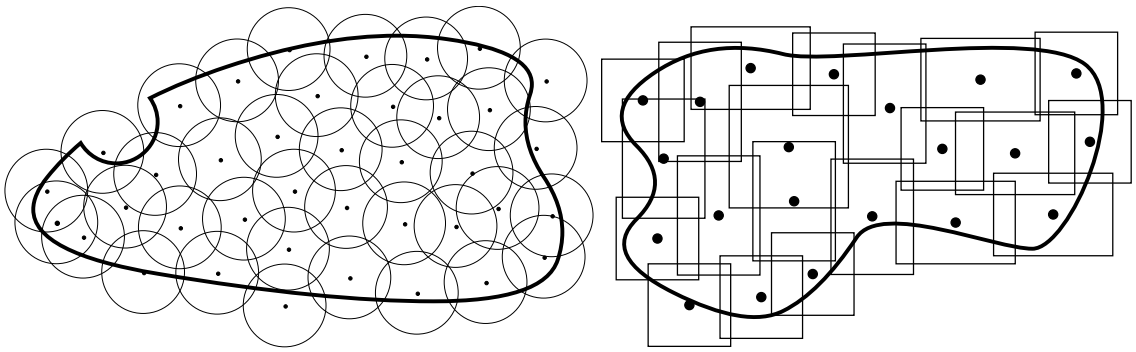


Figure 3.3. *Examples of an open cover C_Ω of a domain $\Omega \subset \mathbb{R}^2$ using spherical patches (left) and rectangular patches (right).*

is called a local neighborhood or local cover of a particular cover patch $\omega_i \in C_\Omega$. There exists a constant $C_\mathcal{N} \geq 1$ such that for all patches $\omega_j, \omega_i \in C_\Omega$ the implication

$$\omega_j \cap \omega_i \neq \emptyset, \quad \operatorname{diam}(\omega_i) \geq \operatorname{diam}(\omega_j) \qquad \Longrightarrow \qquad \frac{\operatorname{diam}(\omega_i)}{\operatorname{diam}(\omega_j)} \leq C_\mathcal{N}. \tag{3.12}$$

holds.

Based on an admissible cover, compare Figure 3.3, we can now employ the MLS construction, i.e. Shepard's method, of the previous chapter to obtain a PU that satisfies Definition 3.1 and Definition 3.2. To this end let us assume that non-negative weight functions W_k are associated with the cover patches ω_k, i.e. $W_k(x) > 0$ for all $x \in \omega_k \setminus \partial\omega_k$. Recall that the Shepard functions are defined as

$$\varphi_i(x) := \frac{W_i(x)}{S_i(x)} \quad \text{where } S_i(x) := \sum_{\omega_j \in C_i} W_j(x). \tag{3.13}$$

Obviously, the smoothness of the resulting PU functions φ_i is determined entirely by the smoothness of the employed weight functions. Hence, on a cover with tensor product patches ω_i we can easily construct partitions of unity of any regularity for instance by using tensor products of splines with the desired regularity as weight functions.[2] Hence, let us assume that the weight functions W_i are all given as affine transformations of a generating normalized spline weight function $\mathcal{W} : \mathbb{R}^D \to \mathbb{R}$ with $\operatorname{supp}(\mathcal{W}) = [-1,1]^D$, i.e.,

$$\begin{gathered} W_i(x) = \mathcal{W} \circ T_i(x), \quad T_i : \omega_i \to [-1,1]^D, \quad \|DT_i\|_\infty \leq \frac{C_T}{\operatorname{diam}(\omega_i)}, \quad \|\mathcal{W}\|_\infty = 1, \\ \|\nabla \mathcal{W}\|_\infty \leq C_\mathcal{W}, \quad \mathcal{W}(x) \geq C_{\mathcal{W},\partial} \operatorname{dist}(x, \partial([-1,1]^D)) \text{ for all } x \in (-1,1)^D \end{gathered} \tag{3.14}$$

Then there holds the following Lemma.

Lemma 3.1. *The PU defined by (3.13) with weights (3.14) defined on an admissible cover C_Ω is valid according to Definition 3.1 and satisfies Definition 3.2.*

[2] Other shapes of the cover patches $\omega_i \in C_\Omega$ are of course possible, e.g. balls or ellipsoids compare Figure 3.3, but the resulting partition of unity functions φ_i are more challenging to integrate numerically. For instance a subdivision scheme based on the piecewise constant covering index λ_{C_Ω} leads to integration cells with very complicated geometry, see §6.1.

Proof. Note that $S_i(x) \leq \lambda_{C_\Omega}(x) \leq M$ for all $x \in \Omega$ due to (3.14). Furthermore, for all $x \in \omega_i$ there holds

$$S_i(x) = \sum_{\omega_j \in C_i} W_j(x) = \sum_{j=1}^{N} W_j(x) \geq \mathcal{W} \circ T_k(x).$$

With (3.10) and (3.14) we obtain the lower bound

$$|S_i(x)| \geq 2C_S C_{\mathcal{W},\partial}$$

for all $x \in \Omega$. Together with (3.14) and (3.12) this yields the point-wise estimate

$$\begin{aligned} |\nabla \varphi_i(x)| &= \Big| \frac{W_i(x)\nabla S_i(x) - \nabla W_i(x) S_i(x)}{S_i^2(x)} \Big| \\ &\leq \frac{\Big(|\nabla \mathcal{W} \circ T_i(x) DT_i(x) S_i(x)| + |W_i(x) \sum_k \nabla \mathcal{W} \circ T_k(x) DT_k(x)| \Big)}{|S_i^2(x)|} \\ &\leq \frac{(C_\mathcal{N}+1) M C_T C_\mathcal{W}}{(C_S C_{\mathcal{W},\partial})^2 \operatorname{diam}(\omega_i)}, \end{aligned}$$

which gives the asserted bound $\|\nabla \varphi_i\|_{L^\infty(\mathbb{R}^D)} \leq \frac{C_\nabla}{\operatorname{diam}(\omega_i)}$ with $C_\nabla \geq (C_\mathcal{N}+1) C_\mathcal{W} M C_T (C_{\mathcal{W},\partial} C_S)^{-2}$. Property (3.7) follows directly from (3.8) and therefore $C_\infty = 1$. □

It remains to specify the local approximation spaces V_i employed in the PUM. From Theorem 3.1 it is clear that we are by no means limited to classical polynomial approximation spaces V_i. The ability to employ problem-dependent local approximation spaces in the PUM is one of its key advantages over classical numerical methods. On the other hand, polynomials are highly appropriate local spaces V_i if we are interested in the approximation of locally smooth solutions. Thus, we make the convention that the local approximation spaces V_i in our PUM are composed of two sub-spaces: A smooth part $\mathcal{P}^{p_i} := \operatorname{span}\langle \psi_i^s \rangle$ comprised of polynomials ψ_i^s of total degree $p \leq p_i$, and a problem-dependent enrichment part $\mathcal{E}_i := \operatorname{span}\langle \eta_i^t \rangle$; i.e.,

$$V_i := \mathcal{P}^{p_i} + \mathcal{E}_i = \operatorname{span}\langle \psi_i^s, \eta_i^t \rangle. \tag{3.15}$$

For the time being let us assume that the system of functions $\langle \psi_i^s, \eta_i^t \rangle$ provide a stable basis for V_i, see §4.2.1 and §5.3.2 on how this assumption can be satisfied in general. For the ease of notation we furthermore define $\langle \vartheta_i^n \rangle := \langle \psi_i^s, \eta_i^t \rangle$. Apart from this assumption we impose no further restriction on the choice of the local approximation spaces. The resulting PUM space then becomes

$$V^{\mathrm{PU}} := \sum_{i=1}^{N} \varphi_i V_i = \operatorname{span}\langle \varphi_i \vartheta_i^n \rangle = \sum_{i=1}^{N} \varphi_i \mathcal{P}^{p_i} + \sum_{i=1}^{N} \varphi_i \mathcal{E}_i = \operatorname{span}\langle \varphi_i \psi_i^s, \varphi_i \eta_i^t \rangle. \tag{3.16}$$

3.2 Boundary Conditions

Our PUM shape functions $\varphi_i \vartheta_i^n$ are not cardinal functions; i.e., they do not satisfy the Kronecker δ-condition. One of the reason for this is that the PU functions themselves do not satisfy the Kronecker condition since it is not guaranteed that $\varphi_i(x_i) = 1$ due to the fact that $x_i \notin \omega_{\mathrm{FT},i}$ is allowed by the construction presented above. Furthermore, the usage of multi-dimensional local approximation spaces V_i generates an approximation space V^{PU} with more degrees of freedom than sampling points $\mathcal{X}_N = \{x_i \,|\, i = 1, \ldots, N\}$. Thus, the treatment of boundary conditions is not straightforward in the PUM (as it is for most meshfree methods).

Consider the abstract model problem

$$\begin{array}{rcll} Lu & = & f & \text{in } \Omega \subset \mathbb{R}^D, \\ B_N u & = & g_N & \text{on } \Gamma_N, \\ B_D u & = & g_D & \text{on } \Gamma_D := \partial\Omega \setminus \Gamma_N, \end{array} \tag{3.17}$$

where L is a symmetric partial differential operator of second order and B_N and B_D express suitable boundary conditions. First consider (3.17) with pure Neumann boundary conditions $B_N u = u_n := \partial u/\partial n := \nabla u \cdot n = g$ on $\Gamma_N := \partial\Omega$, where n denotes the outer normal. Here, we learn from the variational formulation

$$F(v) := \frac{1}{2}a(v,v) - \langle f, v\rangle_{L^2} - \int_{\partial\Omega} gv \to \min\{v \in H^1(\Omega)\}, \tag{3.18}$$

that the trial functions v have to fulfill no additional constraint besides being from the definition space $H^1(\Omega)$ of the differential operator L in its weak form. The boundary conditions are not imposed explicitly on the function space; i.e., the employed basis functions do not need to satisfy the boundary conditions explicitly. Thus, the basis of a finite-dimensional subspace $V \subset H^1(\Omega)$ used to approximate the solution of (3.18) may be compiled of arbitrary functions $v \in H^1(\Omega)$. Hence, we may use our PUM shape functions $\varphi_i\vartheta_i^n$ as trial and test functions in a Galerkin procedure without any modification for the approximation of pure Neumann problems.

However, Dirichlet boundary conditions $B_D u = u = g$ on $\Gamma_D \neq \emptyset$ explicitly impose the values of the solution u on the boundary segment Γ_D. Thus, the trial space of the usual weak formulation

$$\text{Find } u \in H^1_D(\Omega): \quad a(u,v) = \langle f, v\rangle_{L^2} \text{ for all } v \in H^1_0(\Omega)$$

is not the complete space $H^1(\Omega)$ but $H^1_D(\Omega) := \{v \in H^1(\Omega) \mid B_D u = g \text{ on } \Gamma_D\}$, whereas the test space is $H^1_0(\Omega)$. The PU functions φ_i however do not vanish on the boundary so that we must realize the boundary conditions via the local approximation spaces V_i to obtain a conforming treatment of essential boundary conditions in the PUM.

3.2.1 Conforming Local Approximation Spaces

To this end, we need to assume that the local basis system $\langle\vartheta_i^n\rangle$ of V_i for all $\omega_i \in C_\Omega$ with $\omega_i \cap \Gamma_D \neq \emptyset$ can be split into the sub-systems $\langle\vartheta_{i,D}^n\rangle$ and $\langle\vartheta_{i,0}^n\rangle$ where $\langle\vartheta_{i,0}^n\rangle$ is used for the respective PUM test space and $\langle\vartheta_{i,D}^n\rangle$ is employed to approximate the boundary data. A seemingly simple solution to this assumption is the use of classical finite elements as local approximation spaces V_i. A similar approach was also proposed in [84]. This however destroys the meshfree character of the PUM and explicitly requires mesh-generation near the Dirichlet boundary. Other a priori approaches to the construction of conforming local approximation introduce severe restrictions on the cover C_Ω and essentially lead to the same mesh-generation issues as the previous approach. Hence, we will not make the assumption that the local approximation spaces V_i employed in our PUM are given a priori via a conforming splitting. Note however that there is a simple a posteriori technique [129] that automatically constructs the required splitting of the local approximation spaces, see §4.2.2.

3.2.2 Non-conforming Approaches

There are many different non-conforming techniques for the treatment of essential boundary conditions, see e.g. [5, 46, 68, 124] and the references therein. Within the meshfree context one of the

1. penalty or perturbation methods [74, 96],

2. the Lagrange multiplier method [60, 124],
3. or Nitsche's method [5, 64, 76]

is usually employed.

The penalty or perturbation approaches are very general concepts for the implementation of constraints in a variational problem. In our setting we would introduce an additional surface term in the variational formulation to enforce the boundary conditions. This penalty term may change the properties of the functional and we need to be concerned with the issues of existence and uniqueness of a solution. Furthermore, we usually do not achieve the maximal rate of convergence [3, 5, 108]; i.e., thus we would experience a reduction in the approximation quality of the overall method just because of the inappropriate treatment of boundary conditions.

The Lagrange multiplier method is a general approach toward the solution of constrained minimization problems which is also used in the finite element [4, 24] and wavelet [86] context to implement essential boundary conditions. It is well-known that the method converges with the optimal rates if the function spaces involved, in our setting the interior PUM approximation space and the multiplier space on the boundary, fulfill a (discrete) Ladyzhenskaya–Babuška–Brezzi (or inf-sup) condition. Here, the main problem is the design of an appropriate multiplier space on the boundary. Within the finite element context Pitkäranta [113–115] showed that there is not much freedom in the design of the multiplier space if the optimal convergence of the method is desired. Furthermore, the use of Lagrange multipliers (in general) leads to a saddle-point problem and the arising linear system is indefinite and the design of an optimal solver is not an easy task.

A different variational approach to Dirichlet problems due to Nitsche [108], however, allows for the use of subspaces $V_N \subset H^1(\Omega)$ which do not have to satisfy the boundary conditions explicitly, yet it gives the optimal rate of convergence. Let us shortly review this approach in the following referred to as Nitsche's method, see [1, 14, 98, 125, 133] for details and the connection to Mortar techniques. To this end, we first consider the Poisson problem

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega \subset \mathbb{R}^D, \\ u &= g \quad \text{on } \partial\Omega, \end{aligned} \tag{3.19}$$

for reasons of simplicity. Let $\partial_n u := \frac{\partial u}{\partial n}$ denote the normal derivative. Testing equation (3.19) with any sufficiently smooth test function v yields

$$\int_\Omega \nabla u \cdot \nabla v \, dx - \int_{\partial\Omega} (\partial_n u) v \, ds = \int_\Omega f v \, dx. \tag{3.20}$$

Since $u = g$ on $\partial\Omega$ is given as an essential boundary condition, we can introduce the terms

$$\int_{\partial\Omega} u(\partial_n v) \, ds = \int_{\partial\Omega} g(\partial_n v) \, ds$$

in (3.20) and obtain the symmetric formulation

$$\int_\Omega \nabla u \cdot \nabla v \, dx - \int_{\partial\Omega} (\partial_n u) v \, ds - \int_{\partial\Omega} u(\partial_n v) \, ds = \int_\Omega f v \, dx - \int_{\partial\Omega} g(\partial_n v) \, ds. \tag{3.21}$$

Yet, the problem (3.21) is not uniquely solvable since the associated bilinear form is not definite. To overcome this issue, we define additional regularization terms again exploiting the given boundary data $u = g$. For instance, we can introduce the terms

$$\int_{\partial\Omega} uv \, ds = \int_{\partial\Omega} gv \, ds.$$

Adding a β multiple of these regularization terms to (3.21) gives

$$\int_\Omega \nabla u \cdot \nabla v \, dx - \int_{\partial\Omega} \big((\partial_n u)v + u(\partial_n v)\big) \, ds + \beta \int_{\partial\Omega} uv \, ds = \int_\Omega fv \, dx - \int_{\partial\Omega} g(\partial_n v) \, ds + \beta \int_{\partial\Omega} gv \, ds. \quad (3.22)$$

Hence, we arrive at the bilinear form

$$a_\beta(u,v) := \int_\Omega \nabla u \cdot \nabla v \, dx - \int_{\partial\Omega} \big((\partial_n u)v + u(\partial_n v)\big) \, ds + \beta \int_{\partial\Omega} uv \, ds \quad (3.23)$$

and the associated linear form

$$\langle l_\beta, v\rangle := \int_\Omega fv \, dx - \int_{\partial\Omega} g(\partial_n v) \, ds + \beta \int_{\partial\Omega} gv \, ds. \quad (3.24)$$

The regularization parameter β can now be used to enforce the definiteness of the bilinear form (3.23) and the stability of the above scheme on a particular *subspace* of $V \subset H^1(\Omega)$ under the assumption that there holds the inverse estimate

$$\|\partial_n u\|_{L^2(\partial\Omega)} \leq C_{\text{inv}} \|\nabla u\|_{L^2(\Omega)}$$

for all $u \in V$ with some constant $C_{\text{inv}} > 0$. This is readily seen from the inequality

$$\begin{aligned} a_\beta(u,u) &\geq \|\nabla u\|^2_{L^2(\Omega)} - 2\|\partial_n u\|_{L^2(\partial\Omega)}\|u\|_{L^2(\partial\Omega)} + \beta\|u\|^2_{L^2(\partial\Omega)} \\ &\geq \|\nabla u\|^2_{L^2(\Omega)} - 2C_{\text{inv}}\|\nabla u\|_{L^2(\Omega)}\|u\|_{L^2(\partial\Omega)} + \beta\|u\|^2_{L^2(\partial\Omega)} \\ &\geq \frac{1}{2}\|\nabla u\|^2_{L^2(\Omega)} + (\beta - 2C^2_{\text{inv}})\|u\|^2_{L^2(\partial\Omega)}. \end{aligned}$$

Nitsche proved that the discrete solution of this formulation converges to the exact solution with optimal order in $H^1(\Omega)$ and $L^2(\Omega)$ if $C_{\text{inv}} \asymp C \operatorname{diam}(\operatorname{supp}(\phi_i))^{-1}$ for a basis $\langle \phi_i \rangle$ of V. This additional assumption essentially introduces some geometric constraints on the intersections $\operatorname{supp}(\phi_i) \cap \Omega$ and $\operatorname{supp}(\phi_i) \cap \partial\Omega$; i.e., in our meshfree context on the cover C_Ω or in the finite element context on the regularity of the mesh.

In the following we present the application of Nitsche's method in the PUM context. Here, a slightly different choice of the regularization terms is more appropriate due to the overlapping supports of our shape functions. Before we can define these special regularization terms let us introduce the cover of the boundary $\partial\Omega$

$$C_{\partial\Omega} := \{\omega_i \in C_\Omega \,|\, \gamma_i \neq \emptyset\}, \quad \text{where} \quad \gamma_i := \omega_i \cap \partial\Omega. \quad (3.25)$$

Now we define the discrete *cover-dependent* regularization terms

$$\sum_{\omega_i \in C_{\partial\Omega}} \operatorname{diam}(\gamma_i)^{-1} \int_{\gamma_i} uv \, ds := \sum_{\omega_i \in C_{\partial\Omega}} \operatorname{diam}(\gamma_i)^{-1} \int_{\gamma_i} gv \, ds.$$

Note that these regularization terms are *weighted* sums of *overlapping* boundary integrals. Adding a τ-multiple of these terms to (3.21) yields the bilinear form

$$a_{C_\Omega,\tau}(u,v) := \int_\Omega \nabla u \cdot \nabla v \, dx - \int_{\partial\Omega} \big((\partial_n u)v + u(\partial_n v)\big) \, ds + \tau \sum_{\omega_i \in C_{\partial\Omega}} \operatorname{diam}(\gamma_i)^{-1} \int_{\gamma_i} uv \, ds \quad (3.26)$$

and the associated linear form

$$\langle l_{C_\Omega,\tau}, v\rangle := \int_\Omega fv\,dx - \int_{\partial\Omega} g(\partial_n v)\,ds + \tau \sum_{\omega_i\in C_{\partial\Omega}} \operatorname{diam}(\gamma_i)^{-1}\int_{\gamma_i} gv\,ds. \tag{3.27}$$

What is the benefit of using this formulation over (3.23) and (3.24)? To answer this question let us introduce the following *discrete* cover-dependent norms on the space $H^{\frac{3}{2}+\epsilon}(\Omega)$[3]

$$\begin{aligned}
\|u\|^2_{\frac{1}{2},C_\Omega} &:= \sum_{\omega_i\in C_{\partial\Omega}} \operatorname{diam}(\gamma_i)^{-1}\|u\|^2_{L^2(\gamma_i)},\\
\|\partial_n u\|^2_{-\frac{1}{2},C_\Omega} &:= \sum_{\omega_i\in C_{\partial\Omega}} \operatorname{diam}(\gamma_i)\|\partial_n u\|^2_{L^2(\gamma_i)},\\
\|u\|^2_{1,C_\Omega} &:= \|\nabla u\|^2_{L^2(\Omega)} + \|u\|^2_{\frac{1}{2},C_\Omega} + \|\partial_n u\|^2_{-\frac{1}{2},C_\Omega}.
\end{aligned} \tag{3.28}$$

The respective inverse assumption now reads as follows.

Assumption 3.1 (Inverse Assumption)**.** Consider the discrete function space $V^{\mathrm{PU}} \subset H^{3/2+\epsilon}(\Omega)$. There exists a constant $C_{\mathrm{inv}} > 0$ such that

$$\|\partial_n u\|_{-\frac{1}{2},C_\Omega} \le C_{\mathrm{inv}}\|\nabla u\|_{L^2(\Omega)} \tag{3.29}$$

holds for all $u \in V^{\mathrm{PU}}$.

Lemma 3.2. *If Assumption 3.1 is satisfied. Then, the bilinear form (3.26) is coercive on V^{PU} provided that $\tau > 2C^2_{\mathrm{inv}}$. There hold the estimates*

$$\begin{aligned}
a_{C_\Omega,\tau}(u,u) &\ge \min\{\frac{1}{4},\frac{1}{4}C^{-1}_{\mathrm{inv}},\tau - 2C^2_{\mathrm{inv}}\}\|u\|_{1,C_\Omega} && \text{for all } u\in V^{\mathrm{PU}} \text{ and}\\
|a_{C_\Omega,\tau}(u,v)| &\le (1+\tau)\|u\|_{1,C_\Omega}\|v\|_{1,C_\Omega} && \text{for all } u,v\in H^{3/2+\epsilon}(\Omega).
\end{aligned}$$

Proof. Following the presentation of [98] there hold the inequalities

$$\Big|\int_{\partial\Omega}(\partial_n u)u\,dx\Big| \le \sum_{\omega_i\in C_{\partial\Omega}}\int_{\gamma_i}|(\partial_n u)u|\,dx$$

and

$$\int_{\gamma_i}|(\partial_n u)u|\,dx \le \operatorname{diam}(\gamma_i)^{1/2}\|\partial_n u\|_{L^2(\gamma_i)}\operatorname{diam}(\gamma_i)^{-1/2}\|u\|_{L^2(\gamma_i)},$$

so that with the Cauchy–Schwarz inequality we obtain

$$\begin{aligned}
\Big|\int_{\partial\Omega}(\partial_n u)u\,dx\Big| &\le \sum_{\omega_i\in C_{\partial\Omega}} \operatorname{diam}(\gamma_i)^{1/2}\|\partial_n u\|_{L^2(\gamma_i)}\operatorname{diam}(\gamma_i)^{-1/2}\|u\|_{L^2(\gamma_i)}\\
&\le \Big(\sum_{\omega_i\in C_{\partial\Omega}}\operatorname{diam}(\gamma_i)\|\partial_n u\|^2_{L^2(\gamma_i)}\Big)^{1/2}\Big(\sum_{\omega_i\in C_{\partial\Omega}}\operatorname{diam}(\gamma_i)^{-1}\|u\|^2_{L^2(\gamma_i)}\Big)^{1/2}\\
&= \|\partial_n u\|_{-1/2,C_\Omega}\|u\|_{1/2,C_\Omega}.
\end{aligned}$$

For any $\epsilon > 0$ we infer

$$\begin{aligned}
|a_{C_\Omega,\tau}(u,u)| &\ge \|\nabla u\|^2_{L^2(\Omega)} - 2\|\partial_n u\|_{-1/2,C_\Omega}\|u\|_{1/2,C_\Omega} + \tau\|u\|^2_{1/2,C_\Omega}\\
&\ge \|\nabla u\|^2_{L^2(\Omega)} - \epsilon\|\partial_n u\|^2_{-1/2,C_\Omega} - \epsilon^{-1}\|u\|^2_{1/2,C_\Omega} + \tau\|u\|^2_{1/2,C_\Omega}\\
&\ge \|\nabla u\|^2_{L^2(\Omega)} - \epsilon C^2_{\mathrm{inv}}\|\nabla u\|^2_{L^2(\Omega)} + (\tau - \epsilon^{-1})\|u\|^2_{1/2,C_\Omega}
\end{aligned}$$

[3]The underlying assumption is $\partial_n u \in H^{-1/2}(\partial\Omega)$ is meaningful, i.e. $\partial_n u \in L^2(\partial\Omega)$.

due to the inverse assumption (3.29). Since $(\tau - \epsilon^{-1}) > 0$ and $\tau > 2C_{\text{inv}}^2$, we choose $\epsilon^{-1} = 2C_{\text{inv}}^2$ to establish

$$|a_{C_\Omega,\tau}(u,u)| \geq \frac{1}{2}\|\nabla u\|^2_{L^2(\Omega)} + (\tau - 2C_{\text{inv}}^2)\|u\|^2_{1/2,C_\Omega}.$$

Finally, we apply the inverse estimate (3.29) again to obtain

$$|a_{C_\Omega,\tau}(u,u)| \geq \frac{1}{4}\|\nabla u\|^2_{L^2(\Omega)} + \frac{1}{4}C_{\text{inv}}^{-1}\|\partial_n u\|_{-1,2,C_\Omega} + (\tau - 2C_{\text{inv}}^2)\|u\|^2_{1/2,C_\Omega}.$$

The second estimate is a direct consequence of (3.28) and the trace theorem. □

Hence, the problem

$$a_{C_\Omega,\tau}(u,v) = \langle l_{C_\Omega,\tau}, v\rangle \quad \text{for all } v \in V^{\text{PU}} \tag{3.30}$$

is well-defined and leads to a symmetric positive definite stiffness matrix — if Assumption 3.1 is satisfied and the regularization parameter τ is chosen sufficiently large, i.e. $\tau > 2C_{\text{inv}}^2$. This however requires a good estimate of C_{inv}^2 from Assumption 3.1. Thus we need to be concerned with the automatic computation of a reliable estimate of C_N for our PUM space V^{PU} [125]. To this end, we consider (3.29) as a sequence of generalized eigenvalue problem

$$\mathcal{A}_{\omega_i} x = \lambda_i \mathcal{B}_{\omega_i} x \tag{3.31}$$

where

$$(\mathcal{A}_{\omega_i})_{k,m} := \sum_{\omega_j \in C_{\partial\Omega}\cap C_i} \text{diam}(\gamma_i)^{-1} \int_{\partial\Omega\cap\omega_i} (\varphi_i\vartheta^m)_n(\varphi_i\vartheta_i^k)_n \, ds$$

and

$$(\mathcal{B}_{\omega_i})_{k,m} := \int_{\Omega\cap\omega_i} \nabla(\varphi_i\vartheta_i^m)\nabla(\varphi_i\vartheta_i^k)\, dx$$

for patches ω_i with $\partial\Omega \cap \omega_i \neq \emptyset$. Solving (3.31) for the maximal eigenvalues $\lambda_{i,\max}$ we get a good estimate for $C_N^2 := \max_{i=1}^N \lambda_{i,\max}$.

The advantage of this formulation thus is that the parameter τ is invariant under uniform h-refinement (provided that the inverse assumption holds) and τ is only weakly effected by local h-refinement whereas β strongly depends on any h-refinement. Hence, (3.26) in some sense accounts for local variations in the support sizes via the local weighting factors $\sum_{\omega_i\in C_{\partial\Omega}} \text{diam}(\gamma_i)^{-1}$ (e.g. in adaptive discretizations).

Note that in the case of a PDE with coefficients this automatic computation of the appropriate regularization parameter must be generalized. To this end let us consider the model problem

$$\begin{aligned} -\,\mathbf{div}\,\sigma(u) &= f && \text{in } \Omega \subset \mathbb{R}^D, \\ \sigma(u)\cdot n &= g_N && \text{on } \Gamma_N \subset \partial\Omega, \\ u\cdot n &= g_{D,n} && \text{on } \Gamma_D = \partial\Omega \setminus \Gamma_N, \\ (\sigma(u)\cdot n)\cdot t &= 0 && \text{on } \Gamma_D = \partial\Omega \setminus \Gamma_N, \end{aligned} \tag{3.32}$$

of linear elasticity with

$$\sigma(u) := \mathcal{C}\cdot\epsilon(u), \quad \text{and} \quad \epsilon(u) := \frac{1}{2}\Big(\nabla u + (\nabla u)^T\Big),$$

where $\sigma(u)$ denotes the symmetric stress tensor and $\epsilon(u)$ the symmetric strain tensor. According to the construction steps of Nitsche's method we first test the PDE with a sufficiently smooth function v and integrate by parts. Next we eliminate natural boundary conditions and symmetrize the

bilinear form exploiting the information about essential boundary conditions. Finally we consistently add a regularization term to the bilinear form to ensure its definiteness.

Integrating by parts and utilizing the symmetry of $\sigma(u)$, we obtain

$$-\int_\Omega \mathbf{div}\,\sigma(u)\cdot v\,dx = \int_\Omega \sigma(u):\epsilon(v) - \int_{\partial\Omega}\sigma(u)n\cdot v\,ds = \int_\Omega fv\,dx.$$

Decomposing the boundary $\partial\Omega$ into the Neumann Γ_N and Dirichlet parts $\Gamma_D = \partial\Omega\setminus\Gamma_N$ we can bring the Neumann boundary conditions to the right-hand side. On the left hand side only the *consistency* term on the Dirichlet boundary Γ_D remains.

$$\int_\Omega \sigma(u):\epsilon(v)\,dx - \int_{\Gamma_D}\sigma(u)n\cdot v\,ds = \int_\Omega fv\,dx + \int_{\Gamma_N} g_N v\,ds$$

On Γ_D we have essential boundary conditions in normal direction but vanishing natural boundary conditions in tangential direction. Therefore, we split the integrand of the consistency term on the left hand side into normal and tangential parts.

$$\int_{\Gamma_D}\sigma(u)n\cdot v\,ds = \int_{\Gamma_D}\big((n\cdot\sigma(u)n)n + (t\cdot\sigma(u)n)t\big)\cdot v\,ds$$

With (3.32) we obtain

$$\int_\Omega \sigma(u):\epsilon(v)\,dx - \int_{\Gamma_D}(n\cdot\sigma(u)n)n\cdot v\,ds = \int_\Omega fv\,dx + \int_{\Gamma_N} g_N v\,ds$$

Now we are in a position to symmetrize the bilinear form consistently using available boundary values only.

$$\int_\Omega \sigma(u):\epsilon(v)\,dx - \int_{\Gamma_D}\big((n\cdot\sigma(u)n)n\cdot v + (n\cdot\sigma(v)n)n\cdot u\big)\,ds = \int_\Omega fv\,dx + \int_{\Gamma_N} g_N v\,ds - \int_{\Gamma_D} g_{D,n}(n\cdot\sigma(v)n)\,ds$$

Finally, we introduce the regularization term which again may only involve available boundary information. Hence, the regularization for this model problem employs information in normal direction only, e.g.

$$a_\beta(u,v) := \int_\Omega \sigma(u):\epsilon(v)\,dx - \int_{\Gamma_D}\big((n\cdot\sigma(u)n)n\cdot v + (n\cdot\sigma(v)n)n\cdot u\big)\,ds + \beta\int_{\Gamma_D} u\cdot nv\cdot n\,ds$$

and

$$\langle l_\beta, v\rangle := \int_\Omega fv\,dx + \int_{\Gamma_N} g_N v\,ds - \int_{\Gamma_D} g_{D,n}(n\cdot\sigma(v)n) + \beta\int_{\Gamma_D} g_{D,n}v\cdot n\,ds.$$

The respective discrete cover-dependent forms are given by

$$\begin{aligned} a_{C_\Omega,\tau}(u,v) := \int_\Omega \sigma(u):\epsilon(v)\,dx - \int_{\Gamma_D}\big((n\cdot\sigma(u)n)n\cdot v + (n\cdot\sigma(v)n)n\cdot u\big)\,ds + \\ \tau\sum_{\omega_i\in C_{\Gamma_D}} \operatorname{diam}(\gamma_i)^{-1}\int_{\gamma_i} u\cdot nv\cdot n\,ds \end{aligned} \tag{3.33}$$

and

$$\begin{aligned}\langle l_{C_\Omega,\tau}, v\rangle := \int_\Omega fv\,dx + \int_{\Gamma_N} g_N v\,ds - \int_{\Gamma_D} g_{D,n}(n\cdot\boldsymbol{\sigma}(v)n)\,ds + \\ \tau \sum_{\omega_i\in C_{\Gamma_D}} \operatorname{diam}(\gamma_i)^{-1}\int_{\gamma_i} u\cdot nv\cdot n\,ds.\end{aligned} \tag{3.34}$$

The inverse assumption (3.29) then becomes

$$\begin{aligned}\|(n\cdot\boldsymbol{\sigma}(u)n)\|^2_{-\frac{1}{2},C_\Omega} &\leq C^2_{\text{inv}}\int_\Omega \boldsymbol{\sigma}(u):\boldsymbol{\epsilon}(v)\,dx \\ &= C^2_{\text{inv}}a(u,u) \\ &\leq C^2_{\text{inv}}C_{\text{cont}}\|\boldsymbol{\epsilon}(u)\|^2_{L^2(\Omega,\mathbb{R}^D)}\end{aligned} \tag{3.35}$$

and we can estimate the regularization parameter that automatically accounts for the material parameters by

$$\mathcal{A}_{\omega_i}x = \lambda\mathcal{B}_{\omega_i}x$$

where

$$\left(\mathcal{A}_{\omega_i}\right)_{k,m} := \sum_{\omega_j\in C_{\partial\Omega}\cap C_i} \operatorname{diam}(\gamma_i)^{-1}\int_{\partial\Omega\cap\omega_i}(n\cdot\boldsymbol{\sigma}(u)n)n\cdot(n\cdot\boldsymbol{\sigma}(u)n)n\,ds$$

and

$$\left(\mathcal{B}_{\omega_i}\right)_{k,m} := \int_{\Omega\cap\omega_i}\boldsymbol{\sigma}(u):\boldsymbol{\epsilon}(u)\,dx.$$

Observe that the left-hand side essentially involves the square of the coefficients of the PDE whereas on the right-hand side the coefficients enter linearly only. Thus, the eigenvalues (and thereby also our regularization parameter) are implicitly weighted by the coefficients of the PDE.

Part II

Particle–Partition of Unity Method

Chapter 4

Particle–Partition of Unity Method

The PUM presented in the previous chapter is a general procedure for the construction of an appropriate approximation space V^{PU} for the Galerkin discretization of a partial differential equation (PDE). The fundamental assumption for the PUM introduced above is the availability of an admissible cover C_Ω, see Definition 3.3. The cover C_Ω of the PUM is in some sense the meshfree analogue of a computational mesh in the FEM or other mesh-based methods. The generation of an appropriate computational mesh in the FEM is a rather challenging problem and one of the main reasons which lead to the advent of meshfree methods. Hence, the construction of an admissible cover must be simpler than mesh-generation. So how can we construct an admissible cover C_Ω for an arbitrary input of sampling points $\mathcal{X}_N = \{x_i \in \overline{\Omega} \,|\, i = 1, \ldots, N\}$ efficiently?

In this chapter we now focus on this very critical issue of constructing an admissible cover C_Ω for a general given point set $\mathcal{X}_N$. Hence, we construct a specific PUM from particle positions only. The resulting numerical method is hence denoted particle-partition of unity method (PPUM). Here, we need to consider not only the construction of patches ω_i which cover the domain Ω but also the fast computation of the respective neighborhoods C_i. Both are not trivially computed for a general point set $\mathcal{X}_N = \{x_i \in \overline{\Omega}\}$. The computation of the neighborhoods C_i is essentially a geometric search problem [120]. Hence, tree-based techniques which have been used successfully for searching and sorting problems in many areas [83, 119] can be used to tackle this problem. For instance, there are several tree-based implementations of particle methods for astrophysics applications [35, 91, 146, 147]. In the context of meshfree Galerkin methods a tree-algorithm for finding the patches ω_i covering a single point x; i.e. for the computation of $\mathcal{N}(x)$ of Definition 2.2, was also proposed in [75], however, the cover C_Ω was still assumed to be given.

In the PPUM we employ a hierarchical cover construction algorithm for general domains $\Omega \subset \mathbb{R}^D$ based only on a given set of irregularly spaced points $\mathcal{X}_N = \{x_i \in \overline{\Omega} \,|\, i = 1, \ldots, N\}$ which was first proposed in [61, 125].[1] Here, we partition the domain into overlapping D-rectangular axis-aligned patches ω_i which we assign to the points $x_i \in \mathcal{X}_N$ to cover the complete domain. We use D-binary trees (binary trees, quadtrees, octrees) for the construction of these patches ω_i. Other patch geometries like spheres [36] are of course possible but non-rectangular patches pose additional challenges in the Galerkin discretization, i.e. in numerical integration.

A cover C_Ω generated by the presented hierarchical algorithm is minimal in the sense that $\mathrm{card}(\mathcal{N}(x)) \ll \mathrm{card}(C_\Omega)$ is small for all $x \in \Omega$ and that $\mathrm{card}(C_\Omega) = O(N)$ [21]. In essence, we construct covers for general domains that stay close to k-irregular grids. A k-irregular grid is completely sufficient for a PUM, since the PU functions φ_i will smoothen the jump within the spatial resolution and ensure the global regularity conditions imposed on the approximation u^{PU}.

[1] A similar approach was also proposed by Klaas and Shepard [80].

Unlike in the FEM, there is no need for a special treatment of so-called hanging nodes in the PUM.

4.1 Cover Construction

The cover construction algorithm we employ in the PPUM is essentially a sub-division technique. The starting point of our approach is the construction of a bounding box $R_\Omega \supset \overline{\Omega}$ of the domain Ω. Obviously, the bounding box provides an initial (coarse) cover of the domain. Hence, any sub-division of the bounding box will also yield a cover of the domain. To this end, we recursively sub-divide the bounding box based on the particle positions $x_i \in \mathcal{X}_N$ and scale the resulting cells to obtain an overlapping covering of the domain Ω, see Figure 4.1. The complete algorithm reads as follows, compare [125].

Algorithm 4.1 (Hierarchical Cover Construction)**.**

1. Given the domain $\Omega \subset \mathbb{R}^D$ and a bounding box $R_\Omega = \prod_{d=1}^{D} [l_\Omega^d, u_\Omega^d] \supset \overline{\Omega}$.
2. Given the sampling points $\mathcal{X}_N = \{x_i \,|\, x_i \in \overline{\Omega}, i = 1, \ldots, N\}$ and a scaling parameter $\alpha > 1$.
3. Build a D-binary tree over R_Ω such that per leaf L at most one $x_i \in \mathcal{X}_N$ lies within the associated cell $\mathcal{C}_L := \prod_{d=1}^{D} [l_L^d, u_L^d]$; see Figure 4.1.
4. Set $C_\Omega := \emptyset$.
5. For the root cell $\mathcal{C}_L = \prod_{d=1}^{D} [l_L^d, u_L^d] = R_\Omega$:
   (a) If current tree cell $\mathcal{C}_L$ is an INNER tree node and $\mathcal{C}_L \cap \Omega \neq \emptyset$:
      i. Descend tree for all successors $\mathcal{C}_S$ of $\mathcal{C}_L$. ($\rightarrow$ 5(a))
   (b) Else if $\mathcal{C}_L \cap \Omega \neq \emptyset$:
      i. Set patch
$$\omega_L := \prod_{d=1}^{D} [x_L^d - h_L^d, x_L^d + h_L^d] \supset \mathcal{C}_L \tag{4.1}$$
      where
$$x_L^d := l_L^d + \frac{1}{2}(u_L^d - l_L^d), \quad \text{and} \quad h_L^d := \frac{\alpha}{2}(u_L^d - l_L^d), \tag{4.2}$$
      with $\alpha > 1$, i.e. $\omega_L = \alpha \mathcal{C}_L$.
      ii. Set $C_\Omega = C_\Omega \cup \{\omega_L\}$.

The following lemma shows that the attained cover $C_\Omega = \{\omega_i\}$ is admissible for a PUM.

Lemma 4.1. *The cover $C_\Omega = \{\omega_i\}$ constructed by Algorithm 4.1 is an admissible cover according to Definition 3.3 for a particular choice of $\alpha \in (1, 2)$ (explicitly given in the proof below) which depends on the distribution of the sampling points $\mathcal{X}_N$ only.*

Proof. For the ease of notation let us assume $h_i = h_i^d$ for $d = 1, \ldots, D$. Then, we have $h_i \asymp 2^{-l_i} \operatorname{diam}(\Omega)$ where l_i refers to the tree-level of the cell $\mathcal{C}_i$ associated with patch $\omega_i = \alpha \mathcal{C}_i$ due to (4.1). Furthermore, we define the maximal difference of the tree-levels of two overlapping cover patches by
$$L := \max_{\omega_i \in C_\Omega} \max_{\omega_j \in C_i} |l_i - l_j|, \tag{4.3}$$

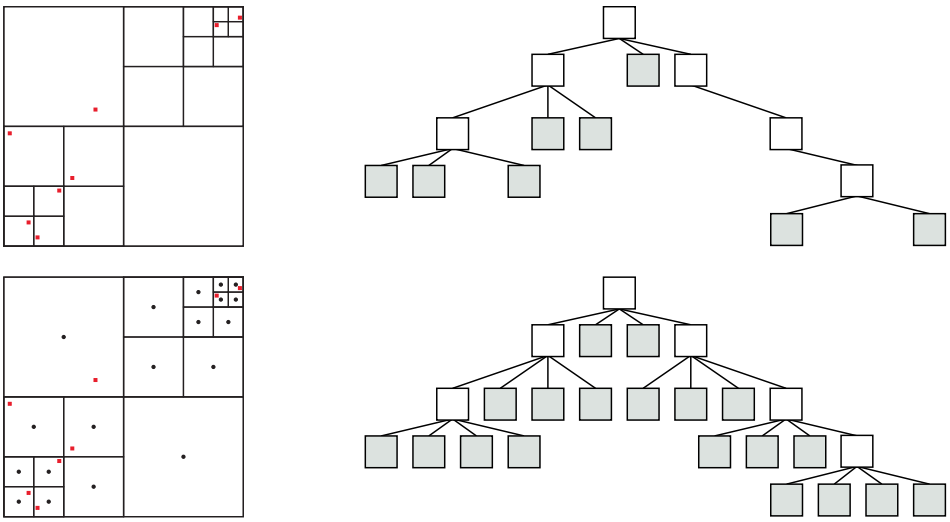


Figure 4.1. *Hierarchical cover construction in two dimensions. The cell decomposition induced by $\mathcal{X}_N$ (upper left, red squares: $x_i \in \mathcal{X}_N$) and its corresponding tree representation (upper right, white:* INNER *tree nodes, gray shaded:* LEAF *tree nodes) after step 3 of Algorithm 4.1. Here, the leaves of the tree correspond to the points $x_i \in \mathcal{X}_N$. The final cell decomposition (lower left) with the cell-centers x_L (black circles) and its tree representation (lower right) after the completion of Algorithm 4.1. Now, the leaves of the tree correspond to the cell-centers.*

compare Figure 4.2, and we obtain the comparability (3.12) of neighboring patches for $C_{\mathcal{N}} \geq 2^L$. Observe that for a uniform tree subdivision the covering index λ_{C_Ω} is easily bounded by

$$\lambda_{C_\Omega}(x) \leq \operatorname{card}(C_i) \leq 3^D$$

for all $x \in \Omega$ with $x \in \omega_i$ since $\alpha < 2$. For an irregular tree subdivision we sum these uniform bounds over all levels l_j of $\omega_j \in C_i$ where $x \in \omega_i$ to attain the bound

$$\lambda_{C_\Omega}(x) \leq \lambda_{C_\Omega}(x) \leq 3^D L$$

and $M \geq 3^D L$ yields (3.9). Yet, this bound for M is rather crude. For an arbitrary $x \in \Omega$ let us define

$$\omega_i := \operatorname{argmax}_{\substack{\omega_j \in C_\Omega \\ x \in \omega_j}} \operatorname{dist}(x, \partial\omega_j).$$

Then, there hold the bounds

$$\operatorname{dist}(x, \partial\omega_i) \leq \frac{1}{2} \operatorname{diam}(\omega_i), \qquad \operatorname{dist}(x, \partial\omega_i) \geq \frac{\alpha - 1}{2} \operatorname{diam}(\omega_i)$$

which yield $\alpha \in [1, 2]$. The inequality (3.8) is equivalent in our setting to

$$\operatorname{diam}(\omega_i) \leq C_{\mathrm{FT}} \operatorname{diam}(\{x \in \omega_i \,|\, \lambda_{C_\Omega}(x) = 1\}). \tag{4.4}$$

Observe that

$$(\alpha - 1) \operatorname{diam}(\omega_i) < \frac{1}{2} \operatorname{diam}(\omega_j)$$

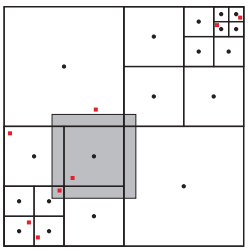


Figure 4.2. *The final cell decomposition obtained by Algorithm 4.1 for the points $x_i \in \mathcal{X}_N$ (red squares) with the respective cell-centers x_L (black circles) and a single patch (gray shaded) $\omega_L = \alpha\, \mathcal{C}_L$ obtained by scaling the respective tree-cell $\mathcal{C}_L$ with $\alpha = 1.2$.*

must hold for all $\omega_i \cap \omega_j \neq \emptyset$ with $\operatorname{diam}(\omega_i) \geq \operatorname{diam}(\omega_j)$ to attain $\lambda_{C_\Omega} = 1$ for some point $x \in \omega_j$. Thus, we obtain the restriction

$$(\alpha - 1) < \frac{1}{2} C_{\mathcal{N}}^{-1} \leq 2^{-L-1}, \quad \text{such that } \alpha < 1 + 2^{-L-1} \tag{4.5}$$

from (3.12) and (4.4) follows with

$$C_{\mathrm{FT}} = \frac{\alpha}{\alpha - 1} \geq 2^{L+1}.$$

□

Remark 4.1. The underlying tree construction of Algorithm 4.1 also enables us to compute the connectivity between the degrees of freedom of our PUM space V^{PU} efficiently. To this end, we only need to compute the neighborhoods C_i via a descent of the tree structure.

Recall from Lemma 3.1 that the Shepard PU (3.13) defined on an admissible cover C_Ω satisfies Definition 3.1 and has the flat top property (3.7). There we obtained the bound $C_\nabla \geq (C_{\mathcal{N}} + 1) C_{\mathcal{W}} M C_T (C_{\mathcal{W},\partial} C_S)^{-2}$ for the respective PU, with the results of Lemma 4.1 this yields

$$C_\nabla \geq (2^L + 1)\, 3^D L\, C_T\, C_{\mathcal{W}} C_{\mathcal{W},\partial}^{-2}\, 2^{2(L+3)} \tag{4.6}$$

for the Shepard PU (3.13) with weights (3.14) defined on the admissible cover C_Ω obtained from Algorithm 4.1 with a scaling parameter of $\alpha = 1 + 2^{-L-2}$.

Note that we may construct more cover patches $\omega_i \in C_\Omega$ than there are sampling points $x_i \in \mathcal{X}_N$, i.e. $\operatorname{card}(C_\Omega) \geq \operatorname{card}(\mathcal{X}_N) = N$, since we define a cover patch ω_i not only for tree-cells $\mathcal{C}$ that contain a particular point $x_i \in \mathcal{X}_N$ but also for tree-cells that do *not* contain a point from the sampling set $\mathcal{X}_N$. Even though this approach increases the number of degrees of freedom of the resulting PPUM space V^{PU} it actually reduces the computational cost of a respective Galerkin discretization with V^{PU} as trial and test space [125]. Furthermore, the resulting number of cover patches $\omega_i \in C_\Omega$ is of the same complexity $O(N)$ if the input data $\mathcal{X}_N$ are (close to) uniformly distributed. For the ease of notation we therefore denote the number of cover patches $N := \operatorname{card}(C_\Omega)$ and refer to the input data as $\mathcal{X}_{\hat{N}}$ with $\hat{N} \leq N$.

Recall from §2.2 that the weight functions W_i (3.14) of the Shepard construction (3.13) can be centered in an arbitrary point $x \in \omega_i$; i.e., there is no benefit of using the sampling points $x_i \in \mathcal{X}_N$ as centers for the weights W_i. Hence, we use the input data $\mathcal{X}_N$ only for the construction of the cover C_Ω and assign weight functions W_i to these cover patches ω_i that are centered in the centers of ω_i, compare (4.2). Due to this decision we obtain PU functions with simpler algebraic structure which substantially reduces the computational effort in numerical integration [125].

4.2 Selection of Local Approximation Spaces

Recall that the local approximation spaces V_i of a general PUM space $V^{\mathrm{PU}} = \sum_{i=1}^{N} \varphi_i V_i$ provide the approximation power to V^{PU}. Since the PU functions φ_i yield a smooth transition between the local approximation spaces there are no explicit compatibility conditions on the V_i as in the FEM. The V_i are completely independent of each other and the use of problem-dependent local approximation spaces V_i is one of the key benefits of the PUM approach. In general we assume that the local approximation spaces $V_i := \mathrm{span}\langle \vartheta_i^n \rangle$ are comprised of a smooth polynomial part $\mathcal{P}^{p_i} = \mathrm{span}\langle \psi_i^s \rangle$ and a problem-dependent enrichment part $\mathcal{E}_i = \mathrm{span}\langle \eta_i^t \rangle$; i.e., we assume that

$$V_i := \mathrm{span}\langle \vartheta_i^n \rangle = \mathcal{P}^{p_i} + \mathcal{E}_i = \mathrm{span}\langle \psi_i^s, \eta_i^t \rangle. \tag{4.7}$$

As a basis $\langle \psi_i^s \rangle$ for the polynomial space $\mathcal{P}^{p_i}$ we usually employ linear transformations of tensor products Ψ^s of univariate Legendre polynomials, i.e.

$$\psi_i^s := \Psi^s \circ T_i, \quad \text{with } T_i : \overline{B(\omega_i \cap \Omega)} \to [-1,1]^D$$

where $B(\omega_i \cap \Omega)$ denotes a minimal bounding box of $\omega_i \cap \Omega$.

Note that we do not need to assume that the system $\langle \vartheta_i^n \rangle$ is a stable basis of V_i in the PPUM. Here, we rather construct an appropriate stable basis from the generating system $\langle \vartheta_i^n \rangle$ automatically which allows us to select problem-dependent enrichment spaces $\mathcal{E}_i$ by focusing on the approximation properties only. There is no need to be concerned with the stability of the resulting system $\langle \vartheta_i^n \rangle = \langle \psi_i^s, \eta_i^t \rangle$ a priori. The stability of the PPUM will be attained automatically by our approach while maintaining the approximation properties of V_i of (4.7).

4.2.1 Stability

We ensure the stability of the PPUM by a special local preconditioning technique [127, 130], see also §5.3.2. Recall that we employ a flat top PU in the PPUM and hence must only be concerned with the stability of the local systems $\langle \vartheta_i^n \rangle$ for all patches $\omega_i \in C_\Omega$. Thus we need to transform each ill-conditioned system (or generating system) $\langle \vartheta_i^n \rangle$ with $V_i = \mathrm{span}\langle \vartheta_i^n \rangle$ to a stable basis $\langle \tilde{\vartheta}_i^m \rangle$ of the respective local approximation space V_i. To this end, we setup the local mass matrix with the entries

$$(M_i)_{m,n} := \int_{\omega_i \cap \Omega} \vartheta_i^n \vartheta_i^m \, dx \quad \text{for all } m,n = 1,\dots,d_i^V \tag{4.8}$$

using all functions ϑ_i^n of the generating system $\langle \vartheta_i^n \rangle$. From the eigenvalue decomposition

$$O_i^T M_i O_i = D_i \quad \text{with } O_i, D_i \in \mathbb{R}^{d_i^V \times d_i^V}$$

of the matrix M_i where

$$O_i^T O_i = \mathbb{I}_{d_i^V}, \qquad (D_i)_{m,n} = 0 \quad \text{for all } m,n = 1,\dots,d_i^V \text{ with } m \neq n$$

we can extract a stable basis $\langle \tilde{\vartheta}_i^m \rangle$ by a simple cut-off of small eigenvalues. To this end let us assume that the eigenvalues $(D_i)_{m,m}$ are given in decreasing order, i.e. $(D_i)_{m,m} \geq (D_i)_{m+1,m+1}$. Then we can easily partition the matrices O_i^T and D_i as

$$O_i^T = \begin{pmatrix} \tilde{O}_i^T \\ K_i^T \end{pmatrix}, \quad D_i = \begin{pmatrix} \tilde{D}_i & 0 \\ 0 & \kappa_i \end{pmatrix}$$

where the mth row of the rectangular matrix $\tilde{O}_i^T$ is an eigenvector of M_i that is associated with an eigenvalue $(D_i)_{m,m} = (\tilde{D}_i)_{m,m} \geq \epsilon\ (D_i)_{0,0}$ and K_i^T involves all eigenvectors that are associated with small eigenvalues. Since $(\tilde{D}_i)_{m,m} \geq \epsilon\ (D_i)_{0,0}$ the operator

$$S_i := \tilde{D}_i^{-1/2} \tilde{O}_i^T$$

is well-defined and can be evaluated stably. Furthermore, the projection S_i removes the near-null space of M_i due to the cut-off parameter ϵ and we have

$$S_i M_i S_i^T = \tilde{D}_i^{-1/2} \tilde{O}_i^T M_i O_i \tilde{D}_i^{-1/2} = \mathbb{I}_{d_i^\Pi}$$

where $d_i^S := \text{card}\{(D_i)_{m,m} \geq \epsilon\ (D_i)_{0,0}\}$ denotes the row-dimension of $\tilde{O}_i^T$ and S_i. Hence, the operator S_i maps the ill-conditioned generating system $\langle \vartheta_i^n \rangle = \langle \psi_i^s, \eta_i^t \rangle$ to a basis $\langle \tilde{\vartheta}_i^m \rangle$ that is optimally conditioned — it is an optimal preconditioner.[2]

Remark 4.2. Note that we do not need to apply the local preconditioner S_i for the evaluation of the basis $\langle \varphi_i \tilde{\vartheta}_i^m \rangle$ in each quadrature point during the assembly of the stiffness matrix. It is sufficient to transform the stiffness matrix A_{GS} which was assembled using the generating system $\langle \varphi_i \vartheta_i^n \rangle = \langle \varphi_i \psi_i^s, \varphi_i \eta_i^t \rangle$ by the block-diagonal operator S with the block-entries

$$(S)_{i,j} := \begin{cases} S_i & i = j \\ 0 & \text{else,} \end{cases} \tag{4.9}$$

for all $i, j = 1, \ldots, N$; i.e., we obtain the stiffness matrix A with respect to the stable basis $\langle \varphi_i \tilde{\vartheta}_i^m \rangle$ as the triple-product[3]

$$A = S A_{\text{GS}} S^T.$$

Remark 4.3. Note that in the presentation above the equivalencies

$$V_i = \text{span}\langle \vartheta_i^n \rangle = \text{span}\langle \tilde{\vartheta}_i^m \rangle$$

hold only up to the employed numerical cut-off parameter $\epsilon > 0$; i.e. we have

$$\text{span}\langle \tilde{\vartheta}_i^m \rangle \approx V_i.$$

For the ease of notation however we keep the notation $V_i = \text{span}\langle \tilde{\vartheta}_i^m \rangle$.

Remark 4.4. Note that in the discussion above we have considered the identity operator $\mathbb{I}$ on the local patch ω_i, i.e. the mass matrix M_i. However, we can construct the respective preconditioner also for different operators e.g. the operator $-\Delta + \mathbb{I}$ which corresponds to the H^1-norm. In exact arithmetic and with a cut-off parameter $\epsilon = 0$ changing the operator in the above construction has an impact on the constants only. However, due to our cut-off parameter ϵ we may obtain a different subspace $\text{span}\langle \tilde{\vartheta}_i^m \rangle$ for different operators with the same ϵ.

Now that we have a stable basis $\langle \varphi_i \tilde{\vartheta}_i^m \rangle$ of our PPUM space $V^{\text{PU}} = \sum_{i=1}^N \varphi_i V_i$ let us come back to the important issue of essential boundary conditions. Recall from §3.2 that the use of (a priori) conforming local approximation spaces V_i is in general not trivial in the PUM so that the non-conforming approach due to Nitsche is usually employed in the PUM. Yet, we can generalize the above approach for the construction of a stable PPUM basis to construct a direct splitting of the employed local approximation spaces V_i into a sub-space $V_{i,K}$ with vanishing traces on the Dirichlet boundary and a sub-space $V_{i,I}$ with non-vanishing traces automatically.

[2] To account for the original scaling of the local mass matrix M, i.e. of the smooth function space $\mathcal{P}^{p_i}$, we employ an additional diagonal scaling based on the matrix block corresponding to the basis functions ψ_i^s. Note that we formally need to construct the stable basis on the flat top region ω_{FT}.

[3] Note however that this operation is easily parallelizable since S and S^T are block-diagonal.

4.2.2 Conformity

Even though Nitsche's approach to the implementation of essential boundary conditions is of optimal complexity and provides an optimally convergent numerical scheme there are some drawbacks. First and foremost, there is the need to construct the respective bilinear form (3.23) and the associated linear form (3.24) *analytically*. Since these forms depend strongly on the configuration of the boundary conditions it is not trivial to change boundary conditions in an interactive user-driven manner. Often a change in the boundary conditions requires some amount of implementation work and a re-assembly of the stiffness matrix on the boundary. Secondly, the essential boundary data is only weakly approximated and the error on the boundary is balanced with the error in the interior by Nitsche's approach. This can be inappropriate in situations where the boundary conditions need to be enforced strictly.

Let us focus on the latter issue first. The essential boundary conditions can of course be enforced (more) strictly in Nitsche's approach by increasing the regularization parameter β. In the limit $\beta \to \infty$ the essential boundary data are strictly enforced in $L^2(\Gamma_D)$ and the convergence of the scheme is still of optimal order. However the constant, i.e. the absolute value of the error, can increase. Moreover a large regularization parameter β has a severely adverse effect on the condition number of the resulting stiffness matrix rendering the solution of the linear system rather challenging.

Nevertheless let us consider the limit case $\beta \to \infty$ in some more detail. To this end, let us consider the model problem

$$\begin{array}{rcrcll} \mathcal{L}(u) & := & -\mathbf{div}\,\boldsymbol{\sigma}(u) & = & f & \text{in } \Omega \subset \mathbb{R}^D, \\ \mathcal{B}_N(u) & := & \boldsymbol{\sigma}(u)\cdot n & = & g_N & \text{on } \Gamma_N \subset \partial\Omega, \\ \mathcal{B}_{D,t}(u) & := & (\boldsymbol{\sigma}(u)\cdot n)\cdot t & = & 0 & \text{on } \Gamma_D = \partial\Omega \setminus \Gamma_N, \\ \mathcal{B}_{D,n}(u) & := & u\cdot n & = & g_{D,n} & \text{on } \Gamma_D = \partial\Omega \setminus \Gamma_N. \end{array} \tag{4.10}$$

For the ease of notation let us introduce the following short-hand notation

$$\begin{array}{rcl} V_\Omega & := & \{v \in V^{\mathrm{PU}} \mid \mathrm{supp}(v) \cap \Gamma_D = \emptyset\}, \\ V_{D,n,K} & := & \{v \in V^{\mathrm{PU}} \mid \mathrm{supp}(v) \cap \Gamma_D \neq \emptyset \text{ and } \mathcal{B}_{D,n}(v) = (v\cdot n)|_{\Gamma_D} = 0\}, \\ V_{D,n,I} & := & \{v \in V^{\mathrm{PU}} \mid \mathrm{supp}(v) \cap \Gamma_D \neq \emptyset \text{ and } \mathcal{B}_{D,n}(v) = (v\cdot n)|_{\Gamma_D} \neq 0\}. \end{array} \tag{4.11}$$

With $\beta \to \infty$ the bilinear form obtained by Nitsche's method for the weak formulation of (4.10), compare (3.23) and (3.33), becomes

$$a_\infty(u,v) = \begin{cases} \displaystyle\int_\Omega \boldsymbol{\sigma}(u) : \boldsymbol{\epsilon}(v)\, dx & v \in V_\Omega, \\ \displaystyle\int_\Omega \boldsymbol{\sigma}(u) : \boldsymbol{\epsilon}(v)\, dx - \int_{\Gamma_D} (n\cdot\boldsymbol{\sigma}(u)n)n\cdot v\, ds & v \in V_{D,n,K}, \\ \displaystyle\int_{\Gamma_D} (u\cdot n)(v\cdot n)\, ds & v \in V_{D,n,I}, \end{cases} \tag{4.12}$$

and we attain the respective linear form

$$\langle l_\infty, v\rangle = \begin{cases} \displaystyle\int_\Omega fv\, dx + \int_{\Gamma_N} g_N v\, ds & v \in V_\Omega, \\ \displaystyle\int_\Omega fv\, dx + \int_{\Gamma_N} g_N v\, ds - \int_{\Gamma_D} g_{D,n}(n\cdot\boldsymbol{\sigma}(v)n)\, ds & v \in V_{D,n,K}, \\ \displaystyle\int_{\Gamma_D} g_{D,n}(v\cdot n)\, ds & v \in V_{D,n,I}. \end{cases} \tag{4.13}$$

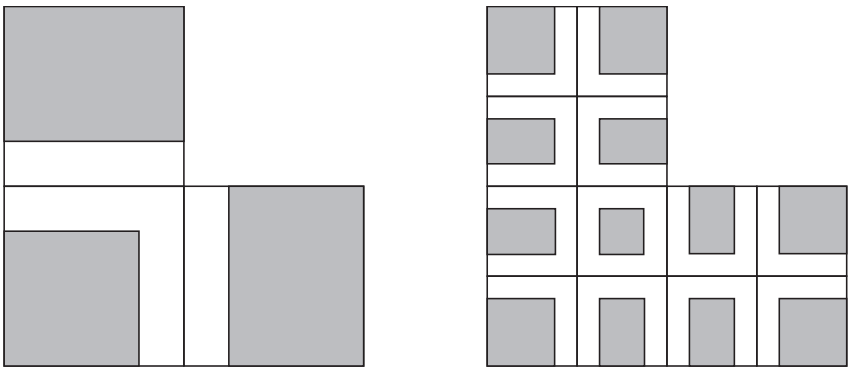


Figure 4.3. *Schematic of a uniform tree decomposition of an L-shaped domain Ω on level $k = 1$ (left) and level $k = 2$ (right). The shaded areas indicate the flat top areas $\omega_{\mathrm{FT},i}$ of the respective PU functions arising from the scaling (4.1) with $\alpha = 1.25$. On level $k = 2$ we find one patch ω_i which overlaps the re-entrant corner and satisfies $\omega_i \cap \partial\Omega \neq \emptyset$ and $\omega_{\mathrm{FT},i} \cap \partial\Omega = \emptyset$. Thus, the respective PU function φ_i does* not *satisfy the flat top condition on the boundary $\partial\Omega$.*

Obviously, in the case $v \in V_{D,n,K}$ we can simplify the weak form back to the classical weak formulation since $\mathcal{B}_{D,n}(u) = g_{D,n}$ and we obtain

$$\int_\Omega \boldsymbol{\sigma}(u) : \boldsymbol{\epsilon}(v)\, dx = \int_\Omega fv\, dx + \int_{\Gamma_N} g_N v\, ds \tag{4.14}$$

for all $v \in V_\Omega + V_{D,n,K}$ which involves no information about the Dirichlet boundary Γ_D. On the other hand, for $v \in V_{D,n,I}$ we need to consider the weak formulation

$$\int_{\Gamma_D} (u \cdot n)(v \cdot n)\, ds = \int_{\Gamma_D} g_{D,n}(v \cdot n)\, ds \tag{4.15}$$

which involves no information about the interior Ω° or the Neumann boundary Γ_N. Note that the above consideration involves a splitting of the test space only, not the trial space. Hence, we obtain the non-symmetric stiffness matrix A and the respective load vector $\hat{f}$ in block-form

$$A = \begin{pmatrix} A_{K,K} & A_{K,I} \\ 0 & B_{I,I} \end{pmatrix}, \qquad \hat{f} = \begin{pmatrix} \hat{f}_K \\ \hat{g}_I \end{pmatrix}. \tag{4.16}$$

where $A_{\cdot,\cdot}$ denotes the use of (4.14) and $B_{\cdot,\cdot}$ denotes the use of (4.15). Hence, the associated linear system $A\tilde{u} = \hat{f}$ with $\tilde{u} = (\tilde{u}_K, \tilde{u}_i)^T$ can (formally) be solved by block-elimination

$$\tilde{u}_I = B_{I,I}^{-1}\hat{g}_I, \quad \tilde{u}_K = A_{K,K}^{-1}(\hat{f}_K - A_{K,I}\tilde{u}_I). \tag{4.17}$$

This is a standard technique in the FEM since the kernel of the trace operator $\mathcal{B}_{D,n}$ (applied to the FEM space) is known a priori so that a respective partitioning (4.11) can be obtained easily. In meshfree methods as the PPUM, however, the kernel of the trace operator $\mathcal{B}_{D,n}$ (applied to the meshfree function space) is *not* known a priori. Furthermore, the computation of the kernel of $\mathcal{B}_{D,n}$ is in general a *global* operation and hence prohibitively expensive. Yet, for the PPUM we can compute the (essential) kernel of the *global* operator $\mathcal{B}_{D,n}$ by *local* operations only. To be more precise we can compute a sub-space $V_I^{\mathrm{PU}} \subset V^{\mathrm{PU}}$ and a localized approximation $\hat{B}_{I,I}$ to the trace operator $\mathcal{B}_{D,n}$ that is invertible on $V_I^{\mathrm{PU}} \subset V^{\mathrm{PU}}$.

For the ease of notation let us assume that the system $\langle \varphi_i \vartheta_i^n \rangle$ denotes a stable basis of V^{PU}, compare §4.2.1. Then, the trace $\mathcal{B}_{D,n}(u^{\mathrm{PU}})$ of an arbitrary PPUM function

$$u^{\mathrm{PU}} = \sum_{i=1}^{N} \varphi_i \sum_{m=1}^{d_i} u_i^m \vartheta_i^m$$

obviously vanishes if the traces $\mathcal{B}_{D,n}(\vartheta_i^m)$ of *all* local approximation functions vanish, i.e.,

$$\mathcal{B}_{D,n}(u^{\mathrm{PU}}) = 0 \quad \Longleftarrow \quad \mathcal{B}_{D,n}(\vartheta_i^m) = 0 \quad \text{for all } (i,m) \tag{4.18}$$

with $i = 1, \ldots, N$, and $m = 1, \ldots, d_i := \dim(V_i)$. We obtain the equivalence

$$\mathcal{B}_{D,n}(u^{\mathrm{PU}}) = 0 \quad \Longleftrightarrow \quad \mathcal{B}_{D,n}(\vartheta_i^m) = 0 \quad \text{for all } (i,m) \tag{4.19}$$

if we assume that the employed PU satisfies a flat top condition also for the Dirichlet boundary. For convex domains Ω this is automatically satisfied by our construction. However at re-entrant corners this boundary flat top property is not ensured by the uniform isotropic scaling of (4.1), see Figure 4.3. Here, the introduction of a more general anisotropic scaling is necessary. The equivalence (4.19) yet is needed only to compute the (global) inverse of $B_{I,I}$ in (4.17). Fortunately, we can avoid the computation of the inverse of $B_{I,I}$ with respect to the global basis $\langle \varphi_i \vartheta_i^m \rangle$ in our PPUM. Thus our construction requires the implication (4.18) only and we can stick with the uniform isotropic scaling (4.1) in our cover construction also for non-convex domains Ω.

Let us consider a patch $\omega_i \cap \Gamma_{D,n} \neq \emptyset$ and its associated local approximation space $V_i(\omega_i) = \mathrm{span}\langle \vartheta_i^m \rangle$ with $d_i = \dim(V_i)$. First, we discretize the trace operator $\mathcal{B}^{D,n}$ locally using the basis $\langle \vartheta_i^m \rangle$, i.e., we compute the (normal part of the) mass matrix $M_i^{D,n}$ on the Dirichlet boundary with the entries

$$(M_i^{D,n})_{k,l} = \int_{\Gamma_{D,n}} (\vartheta_i^k \cdot n)(\vartheta_i^l \cdot n)\, ds \quad \text{for all } k,l = 1, \ldots, d_i. \tag{4.20}$$

Then we compute the eigenvalue decomposition

$$O_i^T M_i^{D,n} O_i = D_i \quad \text{with } O_i, D_i \in \mathbb{R}^{d_i \times d_i}$$

of the matrix $M_i^{D,n}$ where

$$O_i^T O_i = \mathbb{I}_{d_i}, \quad (D_i)_{k,l} = 0 \quad \text{for all } k,l = 1, \ldots, d_i \text{ and } k \neq l,$$

the transformation O_i^T is normal and D_i is diagonal. Let us assume that the eigenvalues $(D_i)_{k,k}$ are given in decreasing order, i.e. $(D_i)_{k,k} \geq (D_i)_{k+1,k+1}$. Then the matrices O_i^T and D_i are block-partitioned as

$$O_i^T = \begin{pmatrix} \tilde{O}_i^T \\ K_i^T \end{pmatrix}, \quad D_i = \begin{pmatrix} \tilde{D}_i & 0 \\ 0 & \kappa_i \end{pmatrix}$$

where the rows of the rectangular matrix K_i^T denote those eigenvectors of the discrete local trace operator $M_i^{D,n}$ that span the (numerical) kernel of $M_i^{D,n}$, i.e. the near-null space. The diagonal matrix κ collects the respective d_i^K vanishing (or small) eigenvalues of $M^{D,n}$, i.e. $(\kappa_i)_{k,k} < \epsilon\, (D_i)_{0,0}$. Hence the product operators

$$\Pi_{i,I}^{D,n} := \tilde{O}_i \tilde{O}_i^T = \begin{pmatrix} \mathbb{I}_{d_i^I} & 0 \\ 0 & 0 \end{pmatrix}, \quad \text{and} \quad \Pi_{i,K}^{D,n} := K_i K_i^T = \begin{pmatrix} 0 & 0 \\ 0 & \mathbb{I}_{d_i^K} \end{pmatrix}$$

with $d_i^I = d_i - d_i^K$ are the projections on the image of the discrete local trace operator $M_i^{D,n}$ and the kernel respectively. These projections operate on the new basis $\langle \tilde{\vartheta}_i^m \rangle$ given by the normal transformation

$$O_i^T : V_i = \text{span}\langle \vartheta_i^m \rangle \to V_i = \text{span}\langle \tilde{\vartheta}_i^m \rangle.$$

Furthermore, we obtain the local sub-spaces

$$V_{i,I} := \tilde{O}_i^T(V_i), \quad \text{and} \quad V_{i,K} := K_i(V_i).$$

Thus, the new basis $\langle \tilde{\vartheta}_i^m \rangle$ (i.e. the respective eigenfunctions of $\mathcal{B}^{D,n}$) provides a direct splitting

$$V_i = V_{i,K} \oplus V_{i,I}$$

of the local space V_i into a sub-space $V_{i,I}$ which is suitable for the approximation of the Dirichlet boundary conditions *locally* on $\omega_i \cap \Gamma_D$ and a sub-space $V_{i,K}$ appropriate for the approximation of the PDE in $\omega_i \cap \Omega$. Considering these local splittings for all $\omega_i \cap \Gamma_D \neq \emptyset$ (for the patches $\omega_i \cap \Gamma_D = \emptyset$ we set $V_{i,K} := V_i$) we obtain the corresponding direct splitting of the global PPUM space V^{PU}, i.e.

$$\sum_{i=1}^N \varphi_i V_i = V^{\text{PU}} = V_K^{\text{PU}} \oplus V_I^{\text{PU}} := \sum_{i=1}^N \varphi_i V_{i,K} \oplus \sum_{i=1}^N \varphi_i V_{i,I} \tag{4.21}$$

and we obtain a partitioned global stiffness matrix A in the form (4.16) as the discretization of (4.10). Yet, the global discrete trace operator $B_{I,I}$ (which may not be invertible) of (4.16) was replaced by the block-diagonal operator $\hat{B}_{I,I} = (M_i^{D,n})$ that is by construction always invertible on the local sub-spaces $V_{i,I}$.

Note that we do not need to assemble the stiffness matrix A_C associated with the classical bilinear form (4.14) directly with respect to the computed basis $\langle \tilde{\vartheta}_i^m \rangle$. We can carry out the assembly of the stiffness matrix using the original basis functions $\langle \vartheta_i^m \rangle$ of V_i and apply the normal block-diagonal transformation C with the entries

$$(C)_{i,j} := \begin{cases} \mathbb{I}_{d_i} & j = i \text{ and } \omega_i \cap \Gamma_D = \emptyset, \\ O_i^T & j = i \text{ and } \omega_i \cap \Gamma_D \neq \emptyset, \\ 0 & j \neq i. \end{cases} \tag{4.22}$$

That is we attain the stiffness matrix A_C in block-form (4.16) with respect to the new basis $\langle \tilde{\vartheta}_i^m \rangle$ as the triple-product

$$A_C := CAC^T$$

via a simple post-processing operation, see also Remark 4.2. Furthermore, the blocks $A_{K,K}^C$ and $A_{K,I}^C$ corresponding to (4.16) of A_C can directly be computed with the help of the projections C_K and C_I where we just replace O_i^T in the definition (4.22) of C by K_i^T and $\tilde{O}_i^T$ respectively; i.e., we have

$$A_{K,K}^C := C_K A C_K^T, \quad \text{and} \quad A_{K,I}^C := C_K A C_I^T. \tag{4.23}$$

The matrix $B_{I,I}$ of (4.16) is replaced by the block-diagonal matrix $\hat{B}_{I,I}^C$ with the entries

$$(\hat{B}_{I,I}^C)_{i,j} := \begin{cases} \tilde{O}_i^T M_i^{D,n} \tilde{O}_i = \tilde{D}_i & j = i \text{ and } \omega_i \cap \Gamma_D \neq \emptyset, \\ 0 & \text{else} \end{cases}$$

This matrix however is never explicitly formed. We rather implement the action of the inverse of $\hat{B}_{I,I}$ directly by local operations on the respective patches ω_i, see Step 7 in Algorithm 4.2.

Note that this purely algebraic approach which yields a conforming local treatment of essential boundary conditions also eliminates the first drawback of Nitsche's approach. The user may now interactively change the boundary conditions of (4.10). A change of the boundary configuration only affects the transformation C and thereby requires only local operations. There is no need to change the employed bilinear form; i.e., we do not need to derive a new weak form analytically nor do we need to compute a new regularization parameter β. There is also no need for a direct re-assembly of the stiffness matrix. We only need to update the respective block-entries of C in (4.22). The entries of C are computed from the local matrices $M^{D,n}$ which involve only the local approximation functions ϑ_i^m *not* the PU functions φ_i. Moreover, $M^{D,n}$ is an operator of order zero and defined on the Dirichlet boundary only. Hence, the computation of the respective integrals is much less involved than the direct assembly of the stiffness matrix for the product functions $\varphi_i \vartheta_i^m$ for patches ω_i overlapping the Dirichlet boundary Γ_D.

Thus, a PPUM discretization of our model problem (4.10) using this conforming formulation of essential boundary conditions is summarized by the following algorithm.

Algorithm 4.2 (PPUM with automatic conforming boundary treatment)**.**

1. Discretize the classical bilinear form (4.14) using the global basis $\langle \varphi_i \vartheta_i^m \rangle$ of the global space V^{PU} ignoring all boundary conditions. Denote the obtained global matrix A_ϑ.

2. Discretize the linear form
$$\langle l_\Omega, v \rangle := \int_\Omega f v \, dx$$
using the global basis $\langle \varphi_i \vartheta_i^m \rangle$ of the global space V^{PU} ignoring all boundary conditions. Denote the obtained global vector $\hat{f}_V$.

3. Discretize the linear form
$$\langle l_N, v \rangle := \int_{\Gamma_N} g_N v \, ds$$
associated with the Neumann boundary conditions using the global basis $\langle \varphi_i \vartheta_i^m \rangle$ of the global space V^{PU} for all patches $\omega_i \cap \Gamma_N \neq \emptyset$. Denote the obtained global vector $\hat{g}_N$.

4. Discretize the linear forms
$$\langle l_{i,D,n}, v \rangle := \int_{\omega_i \cap \Gamma_D} g_D (v \cdot n) \, ds$$
associated with the Dirichlet boundary conditions *locally* on each patch $\omega_i \cap \Gamma_D \neq \emptyset$ using the respective basis $\langle \vartheta_i^m \rangle$. Denote the obtained local vectors $\hat{g}_D^i$.

5. Discretize the bilinear forms
$$b_i(u, v) := \int_{\omega_i \cap \Gamma_D} (u \cdot n)(v \cdot n) \, ds$$
associated with the (restricted) trace operator *locally* on each patch $\omega_i \cap \Gamma_D \neq \emptyset$ using the respective basis $\langle \vartheta_i^m \rangle$ of V_i with $d_i := \dim(V_i)$. Denote the obtained local matrices $M_i^{D,n}$.

6. Compute the eigenvalue decompositions
$$O_i^T M_i^{D,n} O_i = D_i \quad \text{with } O_i, D_i \in \mathbb{R}^{d_i \times d_i}$$

of the local matrices $M_i^{D,n}$ on the respective patches $\omega_i \cap \Gamma_D \neq \emptyset$. Define the sub-matrices corresponding to the block-partitioning

$$O_i^T = \begin{pmatrix} \tilde{O}_i^T \\ K_i^T \end{pmatrix}, \quad D_i = \begin{pmatrix} \tilde{D}_i & 0 \\ 0 & \kappa_i \end{pmatrix}$$

by ordering the eigenvalues $(D_i)_{k,k}$ decreasingly such that $\tilde{D}_i$ is invertible.

7. Solve *locally* on each patch $\omega_i \cap \Gamma_D \neq \emptyset$ for the essential boundary conditions in $V_{i,I} := \tilde{O}_i^T(V_i)$ via
$$\tilde{u}_{i,D} := \tilde{O}_i \tilde{D}_i^{-1} \tilde{O}_i^T \hat{g}_D^i.$$
Define the vector $\tilde{u}_I := (\tilde{u}_{i,D})$ which corresponds to a function $u_D \in V_I^{\mathrm{PU}} := \sum_{i=1}^N \varphi_i V_{i,I}$ with $\mathcal{B}_{D,n}(u_D) = g_D$ on the Dirichlet boundary Γ_D.

8. Define the transformation C according to (4.22) and the respective projections $C_K : V^{\mathrm{PU}} \to V_K^{\mathrm{PU}}$ and $C_I : V^{\mathrm{PU}} \to V_I^{\mathrm{PU}}$. Define the blocks $A_{K,K}^C$ and $A_{K,I}^C$ according to (4.23).

9. Solve globally for the remaining degrees of freedom, i.e. solve in $V_K^{\mathrm{PU}} := C_K(V^{\mathrm{PU}}) = \sum_{i=1}^N \varphi_i K_i^T(V_i)$, via
$$\tilde{u}_K := (A_{K,K}^C)^{-1}(C_k(\hat{f}_V + \hat{g}_N) - A_{K,I}^C \tilde{u}_I). \tag{4.24}$$

10. Apply the transformation C to obtain the solution $\tilde{u}^{\mathrm{PU}}$ with respect to the original global basis $\langle \varphi_i \vartheta_i^m \rangle$ of the global space V^{PU}, i.e. set
$$\tilde{u}^{\mathrm{PU}} := C^T \begin{pmatrix} \tilde{u}_K \\ \tilde{u}_I \end{pmatrix}.$$

Observe that step 7 of Algorithm 4.2 corresponds to the solution of $B_{I,I}\tilde{u}_I = \hat{g}_D$ of (4.17). However the discrete global trace operator $B_{I,I}$ is replaced by the block-diagonal matrix $\hat{B}_{I,I}$ of the discrete local trace operators $M_i^{D,n}$ with respect to the new basis, i.e. $\hat{B}_{I,I}$ is block-diagonal with diagonal blocks $\tilde{D}_i$. Hence, the boundary value g_D is approximated locally on each patch ω_i with $\omega_i \cap \Gamma_D \neq \emptyset$. Recall that the matrix $A_{K,K}^C$ is always invertible on V_K^{PU} due to the use of a flat top PU.

The proposed discretization scheme employs the classical weak formulation of the considered PDE only. Thus, there is no need for the analytical derivation of an appropriate weak form for the particular PDE as with Nitsche's method. The numerical treatment of a general PDE via our PPUM with automatic conforming boundary treatment is straightforward and substantially simplified compared with Nitsche's method. Most importantly, the configuration of boundary conditions can be changed efficiently by local operations only. There is no feedback into the weak formulation of the problem, no computation of a regularization parameter, and the resulting global linear system that needs to be solved is of smaller dimension than with Nitsche's method.

Observe however that the Dirichlet boundary data is approximated *locally* only, i.e. $B_{I,I}$ is replaced by the block-diagonal matrix $\hat{B}_{I,I}$ of the discrete local trace operators $M_i^{D,n}$. The blocks $M^{D,n}$ can be computed very efficiently since they involve only the local basis functions ϑ_i^m *not* the partition of unity functions φ_i; i.e., we ignore the overlap of the patches in the approximation of the Dirichlet data.

Remark 4.5. If the geometry of a particular boundary segment $\omega_i \cap \Gamma_D$ is rather complicated or the employed local approximation space V_i on the respective patch ω_i is not rich enough to resolve the geometry of $\omega_i \cap \Gamma_D$, then the kernel of the discrete local trace operator will be empty. Thus all degrees of freedom of V_i are used for the approximation of the Dirichlet data and the PDE is considered on the patch ω_i only as a correction of the right-hand side via $A_{K,I}$.

4.3 Galerkin Discretization

Let us summarize the complete algorithmic procedure involved with the Galerkin discretization of our model problem (4.10) via the PPUM. Here, besides the classical discretization steps we must also compute the stability transformation/preconditioner S of §4.2.1 and the transformation C which provides a conforming splitting of V^{PU}.

Algorithm 4.3 (PPUM stable and conforming)**.**

1. Compute an admissible cover $C_\Omega = \{\omega_i \,|\, i = 1, \dots, N\}$ via Algorithm 4.1.
2. Define the Shepard functions φ_i according to (3.13) on the admissible cover patches $\omega_i \in C_\Omega$.
3. Select a local space V_i with associated generating system $\langle \vartheta_i^n \rangle$; i.e. $V_i = \operatorname{span}\langle \vartheta_i^n \rangle$ for each patch $\omega_i \in C_\Omega$.
4. Define the global PPUM space V^{PU} with global generating system $\langle \varphi_i \vartheta_i^n \rangle$, i.e.
$$V^{\mathrm{PU}} := \sum_{i=1}^{N} \varphi_i V_i = \operatorname{span}\langle \varphi_i \vartheta_i^n \rangle.$$
5. For each patch $\omega_i \in C_\Omega$ compute local transformation/preconditioner S_i which maps the local generating system $\langle \vartheta_i^n \rangle$ to a stable basis $\langle \vartheta_{i,\mathrm{SB}}^n \rangle$ of V_i.
6. Define the global transformation/preconditioner S (4.9) which maps the global generating system $\langle \varphi_i \vartheta_i^n \rangle$ to a global stable basis $\langle \varphi_i \vartheta_{i\mathrm{SB}}^m \rangle$, i.e.,
$$V^{\mathrm{PU}} = \operatorname{span}\langle \varphi_i \vartheta_{i,\mathrm{SB}}^m \rangle.$$
7. For each patch $\omega_i \in C_\Omega$ with $\omega_i \cap \Gamma_D \neq \emptyset$ compute the local transformation C_i which maps the stable basis $\langle \vartheta_{i,\mathrm{SB}}^m \rangle$ to the stable basis $\langle \vartheta_{i,\mathrm{CSB}}^m \rangle$ and define the respective projections $C_{i,K}$, and $C_{i,I}$ which provide the local direct splitting
$$V_i = V_{i,K} \oplus V_{i,I} := C_{i,K}(V_i) \oplus C_{i,I}(V_i) = \operatorname{span}\langle \vartheta_{i,K,\mathrm{CSB}}^m \rangle \oplus \operatorname{span}\langle \vartheta_{i,I,\mathrm{CSB}}^m \rangle.$$
To this end discretize the trace operator locally using the generating system $\langle \vartheta_i^n \rangle$ of V_i. Denote the obtained local matrix $M_i^{D,n}$. Apply the local preconditioner S_i, i.e. define
$$M_{i,\mathrm{SB}}^{D,n} := S_i M_i^{D,n} S_i^T,$$
and compute C_i, $C_{i,K}$ and $C_{i,I}$ from the eigenvalue decomposition of $M_{i,\mathrm{SB}}^{D,n}$.
8. Define global transformation C according to (4.22) and the respective global projections C_K and C_I to obtain the global direct splitting
$$V^{\mathrm{PU}} = V_K^{\mathrm{PU}} \oplus V_I^{\mathrm{PU}} := C_K(V^{\mathrm{PU}}) \oplus C_I(V^{\mathrm{PU}}) = \sum_{i=1}^{N} \varphi_i V_{i,K} \oplus \sum_{i=1}^{N} \varphi_i V_{i,I}.$$
9. For each patch ω_i with $\omega_i \cap \Gamma_D \neq \emptyset$ discretize the Dirichlet data g_D locally using the generating system $\langle \vartheta_i^n \rangle$. Denote the local vector $\hat{g}_{i,D}$. Apply local preconditioner S_i, the local projection $C_{i,I}$ and solve
$$C_{i,I} M_{i,\mathrm{SB}}^{D,n} C_{i,I}^T \tilde{u}_i^{\mathrm{CSB}} = C_{i,I} S_i g_{i,D},$$
note that $C_{i,I} M_{i,\mathrm{SB}}^{D,n} C_{i,I}^T$ is diagonal and invertible.

10. Define the global representation $\tilde{u}_I^{\mathrm{CSB}} \in V_I^{\mathrm{PU}}$ of the discrete Dirichlet data by

$$(\tilde{u}_I^{\mathrm{CSB}})_i := \tilde{u}_i^{\mathrm{CSB}} \quad \text{for all } \omega_i \cap \Gamma_D \neq \emptyset.$$

11. Discretize the classical bilinear form (4.14) and the respective linear form using the global generating system $\langle \varphi_i \vartheta_i^m \rangle$ of the global space V^{PU}. Denote the obtained global matrix A and the global load vector $\hat{f}$. Apply the global preconditioner S and the global projections C_k and C_I to obtain

$$A_{K,K}^{\mathrm{CSB}} := C_K S A S^T C_K^T, \quad A_{K,I}^{\mathrm{CSB}} := C_K S A S^T C_I^T, \quad \hat{f}_K^{\mathrm{CSB}} := (C_K S \hat{f} - A_{K,I}^{\mathrm{CSB}} \tilde{u}_I^{\mathrm{CSB}}).$$

12. Solve for $\tilde{u}_K^{\mathrm{CSB}} \in V_K^{\mathrm{PU}}$ via

$$A_{K,K}^{\mathrm{CSB}} \tilde{u}_K^{\mathrm{CSB}} = \hat{f}_K^{\mathrm{CSB}}.$$

13. Apply the transformation C and the preconditioner S to obtain the global solution with respect to the generating system $\langle \varphi_i \vartheta_i^n \rangle$ of V^{PU}, i.e. define

$$\tilde{u}^{\mathrm{PU}} := S^T C^T \begin{pmatrix} \tilde{u}_K^{\mathrm{CSB}} \\ \tilde{u}_I^{\mathrm{CSB}} \end{pmatrix}.$$

Thus, the PPUM discretization of a PDE can be split into three major steps: First, an initial setup step, where all geometric operations, i.e. the cover construction and the computation of the neighborhoods C_i for the definition of the PU, are completed. This step is of the computational complexity $O(N \log N)$ if we assume that the input data $\mathcal{X}_N$ is (close to) uniformly distributed. The actual discretization is the second step. If we denote the number of local approximation functions on a particular patch ω_i by $c_i := \mathrm{card}(\langle \vartheta_i^m \rangle)$ we can bound the operation count of this step, i.e. the assembly of the stiffness matrix A, by

$$O\Big(\sum_{i=1}^{N} c_i \sum_{\omega_j \in C_i} c_j\Big) < O\Big(N(\max_{i=1,\dots,N} c_i)^2\Big).$$

The discretization of the load vector $\hat{f}$ is of

$$O\Big(\sum_{i=1}^{N} c_i\Big) < O\Big(N \max_{i=1,\dots,N} c_i\Big)$$

complexity. The computation of the preconditioner S and the transformation C requires

$$O\Big(\sum_{i=1}^{N} c_i^3\Big) < O\Big(N(\max_{i=1,\dots,N} c_i)^3\Big)$$

operations. Finally, we must consider the computational complexity of the solution step. For the degrees of freedom of V_I^{PU} we obtain the bound

$$O\Big(\sum_{\omega_i \cap \Gamma_D \neq \emptyset} c_i\Big) < O\Big(N^{(D-1)/D} \max_{i=1,\dots,N} c_i\Big)$$

due to the fact that the respective linear system is diagonal. The matrix $A_{K,K}^{\mathrm{CBS}}$ however is a general large sparse matrix so that it is not trivial to obtain an optimal operation count for the solution of the respective linear system.

4.4 Solution of Resulting Linear System

In the following we focus on the solution of the large sparse linear system $A\tilde{u} = \hat{f}$ where $A \in \mathbb{R}^{\mathsf{dof}\times\mathsf{dof}}$ and $\tilde{u} \in \mathbb{R}^{\mathsf{dof}}$ denotes a coefficient vector and $\hat{f} \in \mathbb{R}^{\mathsf{dof}}$ denotes a moment vector. The use of an inappropriate linear solver can drive up the compute time as well as the storage demand dramatically.

Classical direct solvers (for dense matrices) like Gaußian elimination or LU-decomposition have a storage requirement of $O(\mathsf{dof}^2)$ and the number of operations even scales with $O(\mathsf{dof}^3)$. For our PUM space we have $\mathsf{dof} = \sum_{i=1}^{N} c_i \leq N c_{\max}$ where $N = \mathrm{card}(C_\Omega)$ denotes the number of patches ω_i and $c_{\max} = \max_{i=1,\ldots,N} c_i$. Hence, the number of operations necessary to solve the stiffness matrix with a classical direct solver is of the order $O((Nc_{\max})^3)$. Since the stiffness matrix A is a sparse block-matrix with dense blocks its storage demand is of the order $O(Nc_{\max}^2)$, yet the storage requirement of the method would increase to $O((Nc_{\max})^2)$ if we apply a direct solver. The use of a more advanced direct solver for sparse matrices can cure this dramatic increase in compute time and storage requirements to some extent only. The minimal complexity of $O(Nc_{\max}^2)$ would still be lost and we experience a significant rise in the demand for computing resources due to the use of a direct solver (for dense or sparse matrices).

Another class of linear solvers are the classical iterative schemes like the Jacobi- or Gauß–Seidel method. Here, we do not have a significant increase in the storage requirements, but the number of operations necessary to obtain the solution of the linear system does also not scale with the optimal complexity. A very sophisticated class of iterative methods which not only show an optimal scaling in the storage demand but also in the operation count are so-called multilevel iterative solvers or multigrid methods [26, 27, 50, 69, 70, 152]. These solvers, however, are not general algebraic methods but involve a substantial amount of information about the discretization and possibly the PDE.[4] Hence, we cannot expect an existing multilevel solver which was designed for a completely different type of discretization to solve our linear system from a meshfree PPUM discretization. We rather need to translate the key ideas and ingredients to our meshfree setting [32, 34, 62, 125, 153].

For the efficient solution of linear systems derived from grid-based discretizations multigrid [70] and multilevel methods [152] have been developed since the late 1970s. Here, we usually deal with nested grids $\Omega_0 \subset \Omega_1 \subset \ldots \subset \Omega_J$ where J denotes the finest level of discretization. In a finite element setting we have the associated *nested* function spaces $\mathcal{V}_k$

$$\mathcal{V}_0 \subset \mathcal{V}_1 \subset \mathcal{V}_2 \subset \cdots \subset \mathcal{V}_{J-1} \subset \mathcal{V}_J,$$

with nodal basis functions $\phi_{i,k} \in \mathcal{V}_k$ that satisfy the Kronecker property $\phi_{i,k}(x_{j,k}) = \delta_{i,j}$. These two properties contribute significantly to the optimal convergence of multigrid methods and they are also the standard prerequisites in the respective convergence proofs.

The fundamental observation which led to the development of multigrid methods was that classical iterative schemes like the Jacobi- or the Gauß–Seidel method reduce oscillatory error components very efficiently but their convergence behavior breaks down for smooth errors. Such smooth errors, however, can be approximated very well on a coarser mesh. Furthermore, these formerly smooth functions (with respect to the original mesh-width) are now again more oscillatory (with respect to the coarser mesh-width). Hence, a classical iterative scheme on the coarser mesh will again start to converge very efficiently. Now, we can either apply this idea recursively or we can use a direct solver on the coarser mesh since the number of degrees of freedom is smaller than on the original mesh. Finally, we need to correct only the current iterate on the orig-

[4] There are algebraic multigrid (AMG) methods [142] but their construction is (in general) based on the assumption of a nodal linear basis. These methods are very involved and a generalization of AMG to meshfree discretizations is not an easy task. Another class of iterative algebraic solvers based on so-called $\mathcal{H}$ matrices [72, 73] may also be applied to meshfree methods to construct (close to) optimal solvers.

inal mesh by the computed solution on the coarse mesh to obtain a better approximation to the solution of the linear system on the fine level. Hence, a multigrid method essentially consist of two operations: the application of a classical iterative method (the so-called *smoother*) on the current mesh and the transfer of information between two successive meshes (the so-called *interlevel transfer*). Let us shortly look at how multilevel methods work. To this end we introduce the standard prerequisites and basic assumptions for a multilevel algorithm.

1. Let $\mathcal{V}_0, \ldots, \mathcal{V}_J$ with $\mathsf{dof}_k := \dim(\mathcal{V}_k)$ be a sequence of (nonnested) finite dimensional vector spaces with $\mathcal{V}_k \subset \mathcal{V}$ for all $k = 0, \ldots, J$ and let $\mathcal{V}_J$ denote the finest discretization space.
2. Assume that we have a symmetric positive definite bilinear form $a(\cdot, \cdot)$ on the function space $\mathcal{V}$ and its respective representation $A_k \in \mathbb{R}^{\mathsf{dof}_k \times \mathsf{dof}_k}$ on the discretization spaces $\mathcal{V}_k$ for $k = 0, \ldots, J$ with $\mathrm{nnz}(A_k) = O(\mathsf{dof}_k)$ non-zero entries.
3. Assume that we have a linear prolongation operator $I_{k-1}^k : \mathcal{V}_{k-1} \to \mathcal{V}_k$ for $k = 1, \ldots, J$ with $\mathrm{nnz}(I_{k-1}^k) = O(\mathsf{dof}_k)$ non-zero entries.
4. Assume that we have a linear restriction operator $I_k^{k-1} : \mathcal{V}_k \to \mathcal{V}_{k-1}$ for $k = 1, \ldots, J$ with $\mathrm{nnz}(I_k^{k-1}) = O(\mathsf{dof}_k)$ non-zero entries.
5. Assume that we have linear smoothing operators $S_k^{\mathrm{pre}} : \mathcal{V}_k \times \mathcal{V}_k \to \mathcal{V}_k$ and $S_k^{\mathrm{post}} : \mathcal{V}_k \times \mathcal{V}_k \to \mathcal{V}_k$ on the spaces $\mathcal{V}_k$ for $k = 1, \ldots, J$ which require no more than $O(\mathsf{dof}_k)$ operations.

With these spaces and operators we can define an abstract multiplicative multilevel algorithm:

Algorithm 4.4 (Multilevel Algorithm $M_\gamma^{\nu_1,\nu_2}(k, x_k, b_k)$)**.**

If $k > 0$:

1. For $l = 1, \ldots, \nu_1$: Set $x_k = S_k^{\mathrm{pre}}(x_k, b_k)$.
2. Set $d_{k-1} := I_k^{k-1}(b_k - A_k x_k)$.
3. Set $e_{k-1} := 0$.
4. For $i = 1, \ldots, \gamma$: $e_{k-1} = M_\gamma^{\nu_1,\nu_2}(k-1, e_{k-1}, d_{k-1})$.
5. Set $x_k = C_k(x_k, e_{k-1}) := x_k + I_{k-1}^k e_{k-1}$.
6. For $l = 1, \ldots, \nu_2$: Set $x_k = S_k^{\mathrm{post}}(x_k, b_k)$.

Else:

Set $x_k = A_k^{-1} b_k$.

Note that a single iteration of the multilevel algorithm $M_\gamma^{\nu_1,\nu 2}$ is of linear complexity $O(\mathsf{dof}_J)$ if the series

$$\sum_{k=0}^{J} \gamma^k \frac{\mathsf{dof}_{J-k}}{\mathsf{dof}_J} < \infty \quad \text{for} \quad J \to \infty \tag{4.25}$$

converges [71, 125]. Let us now assume that we want to solve the linear system $A_J x_J = b_J$ up to machine precision. Then, the multilevel iteration $M_\gamma^{\nu_1,\nu 2}$ gives an optimal solver, i.e. returns the solution x_J after $O(\mathsf{dof}_J)$ operations, if (4.25) holds and if the asymptotic error reduction rate

$$\rho_a := \frac{\|x_J^* - x_J^i\|}{\|x_J^* - x_J^{i-1}\|} \quad \text{with } i \to \infty, \tag{4.26}$$

where x_J^* denotes the exact solution to $A_J x_J = b_J$ and x_J^i is the ith iterate, is bounded away from one independent of the number of degrees of freedom dof_J; i.e., $\rho_a \neq \rho_a(\mathsf{dof}_J) < 1$.

However, it is usually not necessary to solve the linear system up to machine precision. Recall that $A_J x_J = b_J$ is a representation of a discretized PDE. Hence, there is already an error involved, the discretization error. Thus, it is sufficient to solve the discrete linear system up to the discretization error only. A more accurate solution of the linear $A_J x_J = b_J$ will not improve the overall error of the approximation. Thus, the stopping criterion is now dependent on the number of degrees of freedom dof. Hence, even though the asymptotic error reduction rate of our multilevel iteration is bounded independent of dof we do not have an optimal solver if we compute the solution x_J of the linear system up to discretization error only. Here, we need an increasing number of iterations for finer discretization spaces since their finer resolution requires a more accurate solution of the resulting system $A_J x_J = b_J$. Here, so-called nested iteration techniques [85] can help to overcome the resulting logarithmic complexity.

Algorithm 4.5 (Nested Multilevel Iteration)**.**

1. If $k > 0$, then set the initial guess
$$x_k^0 := I_{k-1}^k x_{k-1}^{n_{l-1}}.$$
Else, set the initial guess
$$x_k^0 := 0.$$
2. Apply n_k iterations of the multilevel iteration $M_\gamma^{\nu_1,\nu_2}$
$$x_k^{n_k} \leftarrow M_\gamma^{\nu_1,\nu_2}(k, x_k, b_k)$$
to compute the approximate solution $x_k^{n_k}$ on level k.

Again, we exploit the multilevel construction by using a coarser approximation to compute a more suitable initial guess for the iteration on a finer level. Hence, we reduce the number of operations on the finer and more expensive level by shifting as many operations as possible to the coarser level. Then, the solution of the linear system on a coarser level is transferred to a finer level where we now need to apply less iterations of our multilevel iteration (Algorithm 4.4) to reduce the (algebraic iteration) error to the finer discretization error. One instance of such a nested iteration multilevel scheme is the so-called full multigrid V-cycle [28, 29].

Assume that the employed discretization spaces $\mathcal{V}_k$ admit error estimates of the form

$$\|u - u_k\|_{\mathcal{V}} \leq C 2^{-kp} \|u\|_{\mathcal{V}'} \tag{4.27}$$

for all levels $k = 0, \ldots, J$. Then, it is sufficient to employ a single iteration ($n_k = 1$) of the multilevel iteration $M_\gamma^{\nu_1,\nu_2}$ in step 2 of Algorithm 4.5 for all $k = 1, \ldots, J$ if the asymptotic convergence rate (4.26) is bounded by 2^{-p}. The obtained discrete solutions u_k obviously satisfy the bounds (4.27) if we solve the respective linear system on level $k = 0$ up to machine precision. In general we can estimate the number of inner iterations n_k on level k by enforcing $\rho_a^{n_k} < \epsilon_k$ where ϵ_k denotes the respective error reduction factor from level $k-1$ to level k obtained from the respective error bounds for the discretization error.

Chapter 5

Multilevel Particle–Partition of Unity Method

To setup a fast iterative multilevel solver according to Algorithm 4.4 or Algorithm 4.5 for our PPUM discretization we need to generalize our PPUM construction to the multilevel setting. To this end, we need to construct a sequence of PPUM spaces V_k^{PU} with $k = 0, \ldots, J$ and with appropriate transfer operators

$$I_{k-1}^k : V_{k-1}^{\mathrm{PU}} \to V_k^{\mathrm{PU}}, \quad I_k^{k-1} : V_k^{\mathrm{PU}} \to V_{k-1}^{\mathrm{PU}}$$

which connect the spaces V_k^{PU}. Moreover, we need to specify suitable smoothing operators

$$S_k^{\mathrm{pre/post}} : V_k^{\mathrm{PU}} \times V_k^{\mathrm{PU}} \to V_k^{\mathrm{PU}}$$

on levels $k = 1, \ldots, J$. For the ease of notation we limit ourselves at the beginning to the use of polynomial local approximation spaces $V_i = \mathcal{P}^{p_i}$ only. The incorporation of problem-dependent local enrichment spaces $\mathcal{E}_i$ in the multilevel context is the subject of §5.3. Finally, we consider the adaptive refinement of a sequence of general PPUM spaces V_k^{PU} in §5.4.

5.1 Cover Coarsening

The hierarchical construction of a (fine level) cover $C_\Omega = C_\Omega^J$ with Algorithm 4.1 enables us to define a sequence of admissible covers $C_\Omega^k := \{\omega_{i,k}\}$ for $k = 0, \ldots, J$ with similar properties at no significant extra costs. This sequence of covers C_Ω^k can then be used to define a sequence of PPUM spaces V_k^{PU} which is the foundation for the construction of a fast multilevel solver like Algorithm 4.4 for the PPUM. To this end, we extend Algorithm 4.1 to obtain a sequence of covers C_Ω^k rather than just a single fine level cover. In this extended algorithm we allow that a patch ω_L is part of multiple covers of the sequence C_Ω^k. Thus, we introduce the notion of active levels $[k_L^{\min}, k_L^{\max}]$ of a particular patch ω_L where $\omega_L \in C_\Omega^k$ for all $k = k_L^{\min}, \ldots, k_L^{\max}$. On each active level $k = k_L^{\min}, \ldots, k_L^{\max}$ we can employ a different local approximation space, i.e. a different polynomial degree $p_{L,k}$. The respective multilevel cover construction algorithm then reads as follows, see also [125].

Algorithm 5.1 (Multilevel Cover Construction)**.**

1. Given the domain $\Omega \subset \mathbb{R}^D$ and a bounding box $R_\Omega = \prod_{d=1}^D [l_\Omega^d, u_\Omega^d] \supset \overline{\Omega}$.
2. Given the sampling points $\mathcal{X}_N = \{x_i \,|\, x_i \in \overline{\Omega}, i = 1, \ldots, N\}$ and a scaling parameter $\alpha > 1$.

3. Build a D-binary tree over R_Ω such that per leaf L at most one $x_i \in \mathcal{X}_N$ lies within the associated cell $\mathcal{C}_L := \prod_{d=1}^{D} [l_L^d, u_L^d]$; see Figure 4.1.
4. Set J to the finest refinement level of the tree.
5. For all $k = 0, \dots, J$ set $C_\Omega^k := \emptyset$.
6. For the root cell $\mathcal{C}_L = \prod_{d=1}^{D} [l_L^d, u_L^d] = R_\Omega$:
   (a) If current tree cell $\mathcal{C}_L$ is an INNER tree node and $\mathcal{C}_L \cap \Omega \neq \emptyset$:
      i. Descend tree for all successors $\mathcal{C}_S$ of $\mathcal{C}_L$. ($\rightarrow$ 6(a))
      ii. Set patch
      $$\omega_L := \prod_{d=1}^{D} [x_L^d - h_L^d, x_L^d + h_L^d] \supset \mathcal{C}_L$$
      where $x_L := \frac{1}{2^D} \sum x_S$ is the center of its successor points x_S and $h_L^i := 2 \max_S h_S^i$ is twice the maximum radius of its successors h_S^i.
      iii. Set active levels to $[k_L^{\min}, k_L^{\max}]$ with
      $$k_L^{\min} = k_L^{\max} = \min_{\mathcal{C}_S \subset \mathcal{C}_L} k_S^{\min} - 1$$
      and update for all successors $k_{S'}^{\min} = \min_{\mathcal{C}_S \subset \mathcal{C}_L} k_S^{\min}$.
      iv. Set polynomial degrees $p_{L,k} = p_{\min}$ for all $k \in [k_L^{\min}, k_L^{\max}]$ where
      $$p_{\min} := \min_{\mathcal{C}_S \subset \mathcal{C}_L} \min_{q \in [k_S^{\min}, k_S^{\max}]} p_{q,S}.$$
   (b) Else if $\mathcal{C}_L \cap \Omega \neq \emptyset$:
      i. Set patch
      $$\omega_L := \prod_{d=1}^{D} [x_L^d - h_L^d, x_L^d + h_L^d] \supset \mathcal{C}_L, \quad \text{and} \quad h_L^d := \frac{\alpha}{2}(u_L^d - l_L^d) \tag{5.1}$$
      where
      $$x_L^d := l_L^d + \frac{1}{2}(u_L^d - l_L^d) \quad \text{with} \quad \alpha > 1. \tag{5.2}$$
      ii. Set active levels to $[k_L^{\min}, k_L^{\max}]$ with $k_L^{\min} = k_L^{\max} = J$.
      iii. Set polynomial degrees $p_{L,k} = p$ to some given value p for all $k \in [k_L^{\min}, k_L^{\max}]$.
      iv. Set $C_\Omega = C_\Omega \cup \{\omega_L\}$.
7. For $k = 0, \dots, J-1$:
   Set $C_\Omega^k = \{\omega_L \,|\, k_L^{\min} \leq k \leq k_L^{\max}\}$.

In the above algorithm we define a coarser cover $C_\Omega^{k-1} = \{\omega_{i,k-1} \,|\, i = 1, \dots, N_{k-1}\}$ essentially from a finer cover $C_\Omega^k = \{\omega_{i,k} \,|\, i = 1, \dots, N_k\}$ via an agglomeration strategy applied to the respective tree-cells. To this end we collapse those leaves of the tree (with respect to level k) into the associated parent tree-cell if all siblings are also leaf cells (with respect to level k), compare Figure

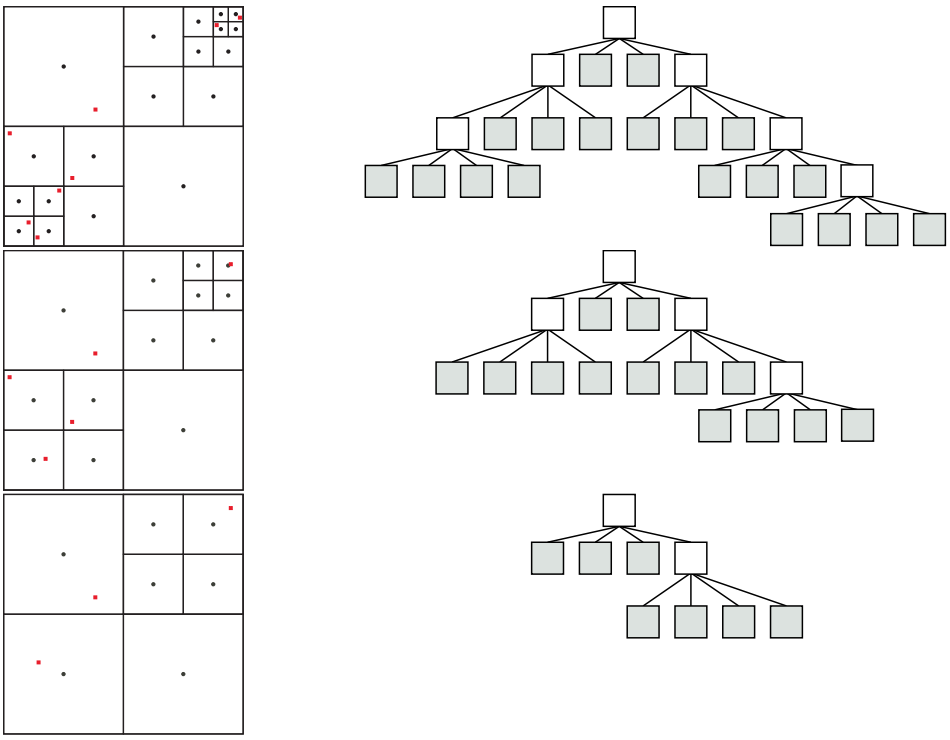


Figure 5.1. *Multilevel cover construction with Algorithm 5.1 in two dimensions. The cell decompositions (left, red squares: $x_i \in \mathcal{X}_N$, black circles: cell-centers x_L) and its respective tree representation (right, white:* INNER *tree nodes, gray shaded:* LEAF *tree nodes) for the given fine level point set $\mathcal{X}_{N_J} = \mathcal{X}_N$ (upper row), and two coarser point sets $\mathcal{X}_{N_{J-1}}$ (center row) and $\mathcal{X}_{N_{J-2}}$ (lower row) attained by a local center-of-mass approach.*

5.1. Note that this coarsening scheme is quite different from the usual level-oriented coarsening employed in the FEM, see Figure 5.2.

The coarse cover patches $\omega_{i,k-1}$ on level $k-1$ are then obtained by scaling of the respective tree-cells, i.e. $\omega_{i,k-1} = \alpha\mathcal{C}_L$. Note that the scaling parameter α can in principle be changed from level to level (or even from patch to patch), yet we have to keep in mind that α has a major impact on the constant C_∇ of Definition 3.1 which is involved in the error bounds of Theorem 3.1. Thus, we keep the scaling parameter constant on all levels $k = 0, \dots, J$ to obtain a uniform bound of the gradients $\nabla\varphi_{i,k}$ on all levels.

Due to this decision however the agglomeration of the tree-cells does not translate into an agglomeration of the cover patches. A coarser cover patch $\omega_{i,k-1}$ is *not* the union of the respective finer cover patches $\omega_{j,k}$. Thus the supports of the PU functions $\varphi_{i,k-1}$ and $\varphi_{j,k}$ obtained by Shepard's construction (3.13) do not align so that it is highly unlikely that we obtain a nested sequence of PPUM function spaces V_k^{PU}, i.e., in general there holds

$$V_{k-1}^{\mathrm{PU}} := \sum_{i=1}^{N_{k-1}} \varphi_{i,k-1} V_{i,k-1} \not\subset V_k^{\mathrm{PU}} := \sum_{i=1}^{N_k} \varphi_{i,k} V_{i,k}.$$

The sequence of covers C_Ω^k attained from Algorithm 5.1 however inherits a hierarchical property from the underlying tree-decomposition which is essential for the construction of an appropriate interlevel transfer operator in §5.2.1.

Lemma 5.1. *For each cover patch $\omega_{i,k} \in C_\Omega^k$ with $k > 0$ there exists exactly one cover patch $\omega_{\tilde{i},k-1} \in C_\Omega^{k-1}$ such that*

$$\omega_{i,k} \subseteq \omega_{\tilde{i},k-1}. \tag{5.3}$$

Another important property of the constructed sequence C_Ω^k with $k = 0, \dots, J$ is summarized in the following lemma, compare Figure 5.1.

Lemma 5.2. *Let $l_{i,k}$ denote the tree-level of the tree-cell $\mathcal{C} \subset \omega_{i,k}$. For all levels $k = 0, \dots, J$ there holds $l_{i,k} \leq k$ for all $\omega_{i,k} \in C_\Omega^k$ and there exists at least one patch $\omega_{i,k} \in C_\Omega^k$ such that $l_{i,k} = k$ is satisfied.*

Thus, the sequence C_Ω^k with $k = 0, \dots, J$ employs a minimal number of levels J and there holds

$$\min_{i=1,\dots,N_k} \operatorname{diam}(\omega_{i,k}) \asymp O(2^{-k})$$

for all $k = 0, \dots, J$. In general however the active levels $k = k_L^{\min}, \dots, k_L^{\max}$ of a particular patch ω_L are independent of the respective tree refinement level l_L of the associated tree-cell $\mathcal{C}_L$, i.e. $k_L^{\max}$ is independent of l_L.

Observe that the property $\omega_L = \omega_{i,k-1} = \omega_{j,k}$ does *not* imply that the respective Shepard functions $\varphi_{i,k-1}$ and $\varphi_{j,k}$ are identical. Recall that the PU function $\varphi_i i, k$ on level k employs information from the neighborhood

$$C_{i,k} := \{\omega_{l,k} \in C_\Omega^k \,|\, \omega_{l,k} \cap \omega_{i,k} \neq \emptyset\}$$

with respect to level k. These neighborhoods however may change, i.e. $C_{i,k} \neq C_{j,k-1}$, even if $\omega_{i,k-1} = \omega_{j,k}$.

Remark 5.1. The use of the minimal polynomial degree $p_L = \min p_S$ of the successor patches ω_S on a particular patch ω_L is motivated by the fact that our transfer operators from level $k-1$ to level k are exact for polynomials of degree $p_{k-1} := \min_j p_{j,k-1}$, see §5.2.1.

5.2 Multilevel Solver

In addition to the fact that coarser shape functions $\varphi_{i,k-1}\vartheta_{i,k-1}^n$ cannot be represented exactly on finer levels, i.e.

$$\varphi_{i,k-1}\vartheta_{i,k-1}^n \neq \sum_{i=1}^{N_k} \varphi_{j,k} \sum_{m=1}^{d_{i,k}} \beta_{j,k}^m \vartheta_{j,k}^m \quad \text{for all } \beta_{j,k}^m \in \mathbb{R} \tag{5.4}$$

due to the nonnestedness of the spaces V_k^{PU}, we also have to deal with shape functions $\varphi_{i,k-1}\vartheta_{i,k-1}^n$ and $\varphi_{j,k}\vartheta_{j,k}^m$ that do not satisfy the Kronecker property. Therefore, the two classical approaches to the interlevel transfer problem, natural injection and interpolation, are not available for our multilevel PPUM. Obviously, these issues render the design of appropriate interlevel transfer operators

$$I_{k-1}^k : V_{k-1}^{\mathrm{PU}} \to V_k^{\mathrm{PU}} \quad \text{and} \quad I_k^{k-1} : V_k^{\mathrm{PU}} \to V_{k-1}^{\mathrm{PU}}$$

a challenging task. In [62, 125] we have developed prolongation operators I_{k-1}^k for our nonnested sequence of PPUM spaces V_k^{PU} based on L^2-projections Π_{k-1}^k

$$I_{k-1}^k := \Pi_{k-1}^k : V_{k-1}^{\mathrm{PU}} \to V_k^{\mathrm{PU}}$$

from V_{k-1}^{PU} onto V_k^{PU}.[1] Let us shortly summarize this construction.

[1] In the context of nonconforming FEM discretizations L^2-projections are also used for the interlevel transfer, e.g. for the Crouzeix–Raviart element [23].

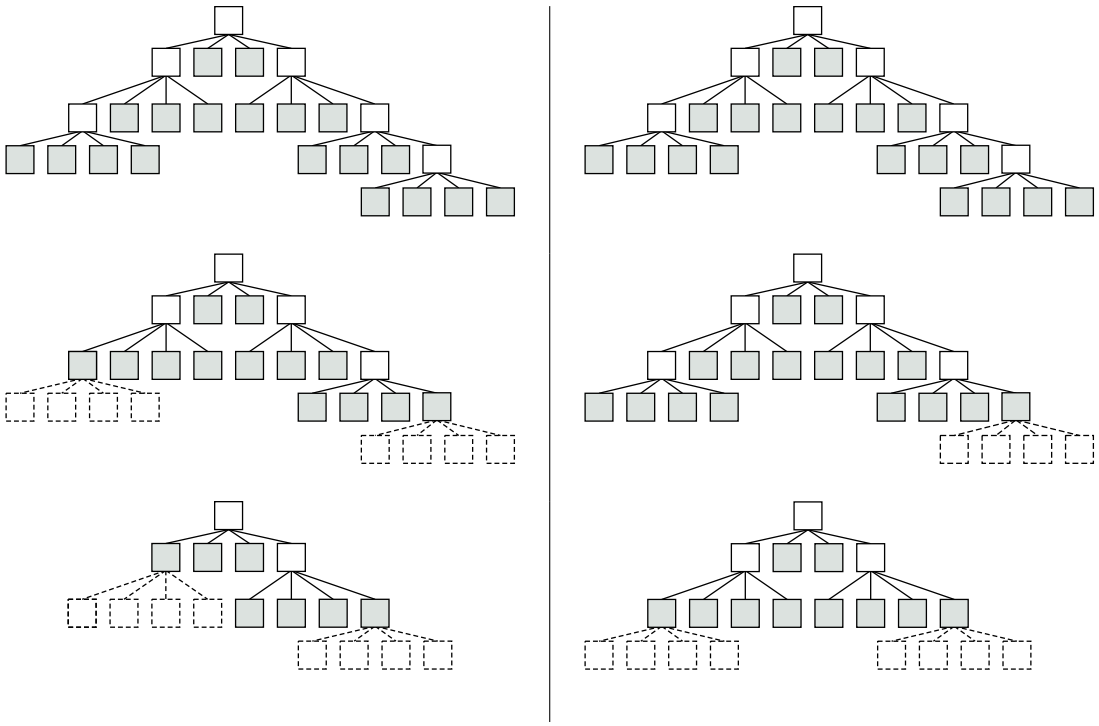


Figure 5.2. *Comparison of a tree-based maximal coarsening (left) and level-oriented coarsening (right). Depicted are three successive levels starting at the top with the finest level (white, solid lines:* INNER *tree nodes; gray shaded, solid lines:* LEAF *tree nodes; white, dashed lines: tree cells eliminated from finer level).*

5.2.1 Interlevel Transfer

In general an L^2-projection $\Pi_W^{\widetilde{W}} : W \to \widetilde{W}$ from $W \subset L^2(\Omega)$ onto $\widetilde{W} \subset L^2(\Omega)$ can be defined with the help of two moment matrices

$$(M_{\widetilde{W}}^{\widetilde{W}})_{ij} := \langle \phi_j^{\widetilde{W}}, \phi_i^{\widetilde{W}} \rangle_{L^2(\Omega)} \quad \text{and} \quad (M_W^{\widetilde{W}})_{ij} := \langle \phi_j^{W}, \phi_i^{\widetilde{W}} \rangle_{L^2(\Omega)}$$

where $\langle \phi_j^W \rangle$ denotes a basis for W and $\langle \phi_j^{\widetilde{W}} \rangle$ a basis for $\widetilde{W}$. The discrete L^2-projection $\Pi_W^{\widetilde{W}} : W \to \widetilde{W}$ can then be defined as

$$\Pi_W^{\widetilde{W}} := (M_{\widetilde{W}}^{\widetilde{W}})^{-1}(M_W^{\widetilde{W}}). \tag{5.5}$$

The projection $\Pi_W^{\widetilde{W}}$ maps coefficients $\widetilde{u}_W$ to coefficients $\widetilde{u}_{\widetilde{W}}$. Its transpose $(\Pi_W^{\widetilde{W}})^T$ obviously transports moment vectors $\hat{f}_{\widetilde{W}}$ to moment vectors $\hat{f}_W$. Hence, if we use an L^2-projection Π as the prolongation it is legitimate to use the transposed projection Π^T as the respective restriction operator.

The selection of $W = V_{k-1}^{\mathrm{PU}}$ with the PPUM basis $\langle \phi_j^W \rangle = \langle \varphi_{i,k-1} \vartheta_{i,k-1}^n \rangle$ and $\widetilde{W} = V_k^{\mathrm{PU}}$ with $\langle \phi_j^{\widetilde{W}} \rangle = \langle \varphi_{i,k} \vartheta_{i,k}^n \rangle$ leads to the mass matrix M_k^k on V_k^{PU} and the interlevel mass matrix M_{k-1}^k from V_{k-1}^{PU} to V_k^{PU}. The global L^2-projection Π_{k-1}^k is then given by

$$\Pi_{k-1}^k = (M_k^k)^{-1}(M_{k-1}^k) \tag{5.6}$$

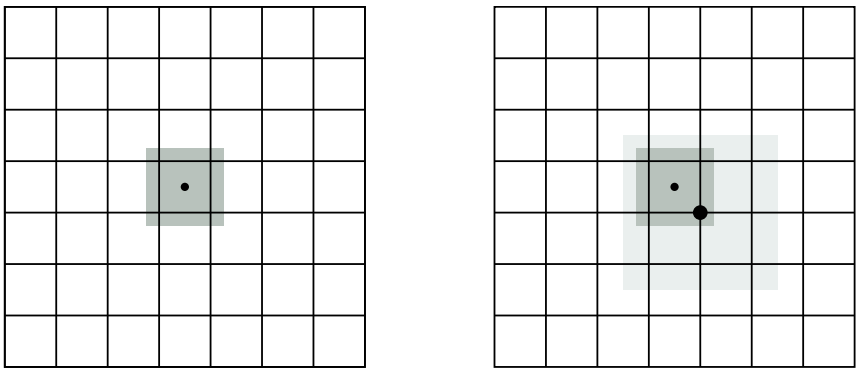


Figure 5.3. *Uniform cells on level l and support of a single shape function (dark gray shaded) on level l which overlaps 3^d cells (left). The support of a coarser shape function (gray shaded) on level $l-1$ overlaps 4^d cells on level l (right).*

where the storage requirement of Π_{k-1}^k is given by the sparsity patterns of M_k^k and M_{k-1}^k. Thus, we need to be concerned with the number of interlevel neighbors $\mathrm{card}(C_{j,k-1,k})$, where

$$C_{j,k-1,k} := \{\omega_{i,k} \in C_\Omega^k \,|\, \omega_{i,k} \cap \omega_{j,k-1} \neq \emptyset\}.$$

These neighborhoods however are rather large, e.g. $\mathrm{card}(C_{j,k-1,k}) = 4^D$ for uniform covers C_Ω^k and C_Ω^{k-1}, compare Figure 5.3. Furthermore, the projection (5.6) involves the inverse of the mass matrix. Thus, the use of the global L^2-projection Π_{k-1}^k as prolongation in a multilevel PPUM is too expensive in practice. We need to find a way to avoid the inversion of the mass matrix M_k^k and we also have to reduce the overall storage demand associated with the interlevel transfer. Let us first consider the mass matrix M_k^k and how we can avoid its involvement in the transfer operator. Then, in a second step we replace M_{k-1}^k to obtain the so-called local-to-local PPUM L^2-projection.

From the basic PUM error estimate (3.5) in $L^2(\Omega)$

$$\|v - v^{\mathrm{PU}}\|_{L^2(\Omega)}^2 \leq C \sum_{i=1}^N \|v - v_i\|_{L^2(\omega_i \cap \Omega)}^2, \tag{5.7}$$

where $v^{\mathrm{PU}} := \sum_{i=1}^N \varphi_i \sum_{n=1}^{d_i} u_i^n \vartheta_i^n$ and $v_i := \sum_{n=1}^{d_i} u_i^n \vartheta_i^n$ where $d_i := \dim(V_i)$, we know that it is sufficient to control the local errors $\|v - v_i\|_{L^2(\omega_i \cap \Omega)}$ on each cover patch ω_i. In the context of our projection problem we have $v = u_{k-1}^{\mathrm{PU}}$ and $v^{\mathrm{PU}} = I_{k-1}^k u_{k-1}^{\mathrm{PU}}$ and (5.7) yields the estimate

$$\|u_{k-1}^{\mathrm{PU}} - I_{k-1}^k u_{k-1}^{\mathrm{PU}}\|_{L^2(\Omega)}^2 \leq C \sum_i \|u_{k-1}^{\mathrm{PU}} - u_{i,k}\|_{L^2(\omega_{i,k} \cap \Omega)}^2 \tag{5.8}$$

for the interlevel transfer problem. Hence, we can localize the approximation of the coarse function u_{k-1}^{PU} to the fine patches $\omega_{i,k}$ without a significant loss of accuracy. Instead of the approximation of the global coarse function $u_{k-1}^{\mathrm{PU}} = \sum_{j=1}^{N_{k-1}} \varphi_{j,k-1} \sum_{m=1}^{d_{i,k-1}} u_{j,k-1}^m \vartheta_{j,k-1}^m$ by the global fine shape functions $\varphi_{i,k}\vartheta_{i,k}^n$ we approximate u_{k-1}^{PU} locally on the fine cover patches $\omega_{i,k}$ using the local basis functions $\vartheta_{i,k}^n$ of $V_{i,k}$ only. The moment matrices associated with this localized projection are

$$\begin{aligned} (\widetilde{M}_k^k)_{(i,n),(i,m)} &:= \langle \vartheta_{i,k}^m, \vartheta_{i,k}^n \rangle_{L^2(\omega_{i,k} \cap \Omega)} \quad \text{and} \\ (\widehat{M}_{k-1}^k)_{(i,n),(j,m)} &:= \langle \varphi_{j,k-1}\vartheta_{j,k-1}^m, \vartheta_{i,k}^n \rangle_{L^2(\omega_{i,k} \cap \Omega)}. \end{aligned} \tag{5.9}$$

The respective projection is given by

$$\widehat{\Pi}_{k-1}^k = (\widetilde{M}_k^k)^{-1}(\widehat{M}_{k-1}^k). \tag{5.10}$$

Here, the localized moment matrix $\widetilde{M}_k^k$ is block-diagonal, but the sparsity pattern of the localized interlevel moment matrix $\widehat{M}_{k-1}^k$ is identical to that of the global interlevel mass matrix M_{k-1}^k since we still approximate the global coarse function u_{k-1}^{PU}.

Recall from Lemma 5.1 that we find exactly one coarse patch $\omega_{\tilde{i},k-1}$ for every fine patch $\omega_{i,k}$ such that $\omega_{i,k} \subseteq \omega_{\tilde{i},k-1}$ holds. Consider the right-hand side of our PUM error estimate (5.8). We can introduce the coarse local function $u_{\tilde{i},k-1}$ associated with the unique coarse patch $\omega_{\tilde{i},k-1}$ into each term $\|u_{k-1}^{\mathrm{PU}} - u_{i,k}\|_{L^2(\omega_{i,k}\cap\Omega)}$ and obtain the estimate

$$\begin{aligned} \|u_{k-1}^{\mathrm{PU}} - u_{i,k}\|_{L^2(\omega_{i,k}\cap\Omega)} \quad \leq \quad & \|u_{k-1}^{\mathrm{PU}} - u_{\tilde{i},k-1}\|_{L^2(\omega_{i,k}\cap\Omega)} + \\ & \|u_{\tilde{i},k-1} - u_{i,k}\|_{L^2(\omega_{i,k}\cap\Omega)} \end{aligned} \tag{5.11}$$

by the triangle inequality. This estimate allows us to approximate each coarse local function $u_{\tilde{i},k-1}$, independent of all other local components $u_{j,k-1}$ of u_{k-1}^{PU}, on the respective fine cover patch $\omega_{i,k}$ with $\omega_{i,k} \subset \omega_{\tilde{i},k-1}$ since the first term of (5.11) is small by definition of u_{k-1}^{PU}. Hence, we need to consider the so-called hierarchical neighbors

$$\begin{aligned} C_{j,k-1,k}^H &:= \{\omega_{i,k} \in C_\Omega^k \,|\, \omega_{i,k} \subseteq \omega_{j,k-1}\}, \\ C_{i,k,k-1}^H &:= \{\omega_{j,k-1} \in C_\Omega^{k-1} \,|\, \omega_{i,k} \subseteq \omega_{j,k-1}\} \end{aligned}$$

only in the construction of the projection

$$\widetilde{\Pi}_{k-1}^k = (\widetilde{M}_k^k)^{-1}(\widetilde{M}_{k-1}^k) \tag{5.12}$$

where the respective moment matrices are defined by

$$\begin{aligned} (\widetilde{M}_k^k)_{(i,n),(i,m)} &:= \langle \vartheta_{i,k}^m, \vartheta_{i,k}^n \rangle_{L^2(\omega_{i,k}\cap\Omega)} \quad \text{and} \\ (\widetilde{M}_{k-1}^k)_{(i,n),(j,m)} &:= \langle \vartheta_{j,k-1}^m, \vartheta_{i,k}^n \rangle_{L^2(\omega_{i,k}\cap\Omega)}. \end{aligned} \tag{5.13}$$

The sparsity pattern of $\widetilde{M}_{k-1}^k$ is now given by the hierarchical condition $\omega_{i,k} \subseteq \omega_{j,k-1}$ instead of the geometric neighbor relation $\omega_{i,k} \cap \omega_{j,k-1} \neq \emptyset$. Therefore, the storage requirement for the projection $\widetilde{\Pi}_{k-1}^k$ is reduced to one block-entry $(\widetilde{\Pi}_{k-1}^k)_{ij}$ per fine level patch $\omega_{i,k}$ without a (significant) loss of accuracy of the approximation, i.e., we have

$$\mathrm{card}(C_{i,k,k-1}^H) = 1$$

independent of the dimension D.

Note that the local-to-local L^2-projection is exact for polynomials of degree $\min_{i=1}^{N_k} p_{i,k}$ and that the assembly involves only integrals of the local approximation functions $\vartheta_{i,k}^m$ which can be computed very efficiently. Hence, the local-to-local transfer operators are very cheap with respect to compute time and storage demand.

Remark 5.2. In general when the local approximation functions $\vartheta_{i,k}^m$ are just a generating system of $V_{i,k}$ the local mass matrix $\widetilde{M}_k^k$ is not invertible. Here, we need to apply the preconditioner S_k to obtain a stable L^2-projection. Then, the local-to-local L^2-projection becomes

$$\widetilde{\Pi}_{k-1}^k = (S_k \widetilde{M}_k^k S_k^T)^{-1}(S_k \widetilde{M}_{k-1}^k S_{k-1}^T) \tag{5.14}$$

where $S_k \widetilde{M}_k^k S_k^T$ is block-diagonal and invertible. In fact, if the preconditioner S_k of §4.2.1, see also §5.3.2, is based on the L^2-norm $S_k \widetilde{M}_k^k S_k^T$ is a (scaled) identity matrix.

5.2.2 Smoothing Operators

The smoothing operators $S^{\text{pre/post}}$ employed in our multilevel PPUM [59, 62, 125] can be conveniently interpreted in the framework of subspace correction methods (SCM) [25, 33, 67, 71, 110, 123, 132, 152, 154]. Hence, let us shortly review the abstract setting of an SCM.

The general idea is as follows: First, we write the employed discretization space $\mathcal{V} = \sum_{j=1}^{\mathcal{N}} \mathcal{V}_j$ as the sum[2] of subspaces $\mathcal{V}_j \subset \mathcal{V}$ with maps $P_j : \mathcal{V}_j \to \mathcal{V}$.[3] Then, we choose symmetric positive definite bilinear forms $b_j(\cdot,\cdot)$ on each $\mathcal{V}_j$ represented by operators B_j such that solutions to the systems of linear equations $B_j u_j = f_j$ on $\mathcal{V}_j$ are easily computable, and B_j^{-1} can be considered as an approximate inverse to the restriction of A to $\mathcal{V}_j$. Finally, we combine these local approximate inverses B_j^{-1} appropriately to define a global approximate inverse to A on the discretization space $\mathcal{V}$. There are essentially two approaches to the definition of an approximate inverse of A by the B_j^{-1}, the additive approach and the multiplicative approach.

In the so-called parallel subspace correction (PSC) or additive Schwarz method we set up an iterative solution process via the operator

$$M_{\text{PSC}} := \mathbb{I} - \omega \sum_{j=1}^{\mathcal{N}} P_j T_j = \mathbb{I} - \omega \Big(\sum_{j=1}^{\mathcal{N}} P_j B_j^{-1} R_j \Big) A, \tag{5.15}$$

where ω is a relaxation parameter and the involved operators are defined by

$$\begin{aligned} a(u,v) &= \langle Au, v \rangle_{\mathcal{V}}, \qquad & b_j(u_j, v_j) &= \langle B_j u_j, v_j \rangle_{\mathcal{V}_j}, \\ \langle R_j u, v_j \rangle_{\mathcal{V}} &= \langle u, P_j v_j \rangle_{\mathcal{V}}, & b_j(T_j u, v_j) &= a(u, P_j v_j). \end{aligned}$$

The iteration operator of the successive subspace correction (SSC) or multiplicative Schwarz method is given by

$$M_{\text{SSC}} := \prod_{j=1}^{\mathcal{N}} \big(\mathbb{I} - P_j T_j \big) = \prod_{j=1}^{\mathcal{N}} \big(\mathbb{I} - P_j B_j^{-1} R_j A \big). \tag{5.16}$$

Note that the PSC operator (5.15) can also be interpreted as a preconditioned Richardson iteration where the preconditioner is given by

$$C_{PSC} := \sum_{j=1}^{\mathcal{N}} P_j B_j^{-1} R_j. \tag{5.17}$$

For the construction of appropriate smoothing operators in our multilevel PPUM we restrict ourselves to the case of $B_j := A|_{\mathcal{V}_j}$ which means that we consider exact subspace solvers only. Then, we have essentially two degrees of freedom in the design of our smoothing scheme: The splitting of the discretization space and the type of the iteration, namely the additive scheme (5.15) or the multiplicative scheme (5.16). To define an appropriate splitting of our PUM space V^{PU} (we omit the level index k in the following), let us consider the specific structure of the PUM shape functions. The product structure of the shape functions $\varphi_i \vartheta_i^n$ implies two natural subspace definitions.

For instance, we can define the subspaces

$$\mathcal{V}_n := \text{span}_i \langle \varphi_i \vartheta_i^n \rangle = \{ v \in V^{\text{PU}} \,|\, v = \sum_{i=1}^{N} \varphi_i v_i^n \vartheta_i^n \}.$$

[2] Note that we do not assume that the splitting is a direct sum.
[3] Actually, it is sufficient to require $\mathcal{V} = \sum_j P_j \mathcal{V}_j$, i.e., the condition $\mathcal{V}_j \subset \mathcal{V}$ is not necessary.

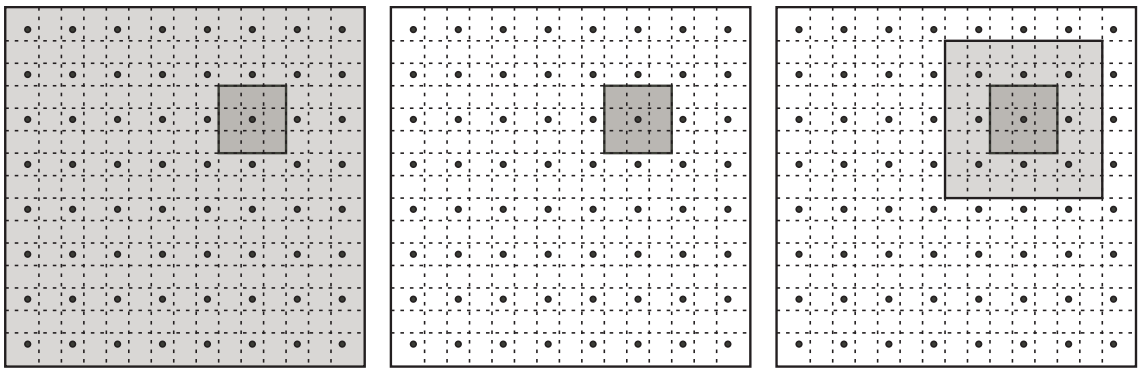


Figure 5.4. *Subdomains (light gray shaded) associated with the subspaces $\mathcal{V}_n$ (left), $\mathcal{V}_i$ (center), and $\widetilde{\mathcal{V}}_l$ (right) and the support of the a single shape function $\varphi_i\vartheta_i^n$ (dark gray shaded) based on a cover with $\alpha = 1.5$.*

These subspaces, however, contain functions with global support on the domain Ω, see Figure 5.4 (left), and the dimension of each subspace is of the order $O(N)$. Therefore, a direct solution of $A|_{\mathcal{V}_n}$ is not feasible. We would need to resort to fast iterative solution techniques for these subspace problems. Furthermore, we are interested in smoothing schemes S_k for Algorithm 4.4 based on our multilevel construction. Hence, there is no additional benefit from the fact that the solutions to $A|_{\mathcal{V}_n}$ contain global information and the computational cost associated with the solution of the subspace problems make this splitting unsuitable for our construction.

A more appropriate subspace definition is given by

$$\mathcal{V}_i := \varphi_i V_i = \operatorname{span}_n\langle \varphi_i\vartheta_i^n\rangle = \{v \in V^{\mathrm{PU}} \mid v = \sum_{n=1}^{d_i} \varphi_i v_i^n \vartheta_i^n\}, \tag{5.18}$$

where $d_i := \dim(V_i)$. These contain functions with local supports only, see Figure 5.4 (center). Furthermore, the dimension of the subspace $\mathcal{V}_i$ is given by the dimension d_i of the local approximation spaces V_i. Hence, we can compute the inverse $(A|_{\mathcal{V}_i})^{-1}$ of each of the subspace problems with acceptable complexity of $O(d_i^3)$; i.e., one iteration of a PSC or SSC iteration based on this splitting is of the order

$$O(\sum_{i=1}^{N} d_i^3) < O(N d_{\max}^3) \quad \text{with} \quad d_{\max} := \max_{i=1,\dots,N} d_i.$$

Note that both subspace definitions lead to a *direct* splitting of our PUM function space $V^{\mathrm{PU}} = \sum_i \mathcal{V}_i = \sum_n \mathcal{V}_n$; i.e., every basis function $\varphi_i\vartheta_i^n$ is contained in exactly one subspace. In terms of the index pairs (i,n) we have a disjoint decomposition of the index set $\{(i,n)\}$ which induces a specific partitioning of the PUM stiffness matrix $A = (A_{(i,n),(j,m)})$. Using the subspaces $\mathcal{V}_i$ we obtain the so-called polynomial block-form [125]. Here, a single block $A_{i,j} = (A_{(i,n),(j,m)})$ corresponds to a local discretization of the PDE on the domain $\omega_i \cap \omega_j \cap \Omega$.

Note that a PSC iteration (5.15) based on the direct splitting $V^{\mathrm{PU}} = \sum_{i=1}^{N} \mathcal{V}_i$ of (5.18) corresponds to the classical block-Jacobi iteration and the SSC iteration (5.16) corresponds to the block-Gauss–Seidel iteration (BGS) where we have only a small overlap of the supports of functions from different subspaces, see Figure 5.4 (center). In general the SSC iteration can be implemented by Algorithm 5.2 where P_l denotes the discrete extension operator which embeds the subspace $\mathcal{V}_l$ in the global discretization space $\mathcal{V}$.

Algorithm 5.2 (Successive Subspace Correction Method)**.**

1. For all $l = 1, \dots, N$:
   (a) Compute local residual $\hat{f}_l := P_l^T(\hat{f} - A\tilde{u})$.
   (b) Solve subspace problem $(P_l^T A\, P_l)\tilde{u}_l = A_{l,l}\tilde{u}_l = \hat{f}_l$.
   (c) Update global iterate $\tilde{u} = \tilde{u} + P_l\tilde{u}_l$.

Even though we consider a direct splitting and employ an exact solver $(A|_{\mathcal{V}_i})^{-1}$ within a specific subspace $\mathcal{V}_i$ there are still couplings between the subspaces due to the overlap of the supports via the global problem A. The quality of the PSC and SSC iterations is obviously determined by the strength of these couplings. The two parameters within our PUM which can influence the strength of the couplings between two different subspaces $\mathcal{V}_i$ and $\mathcal{V}_j$, and hence the quality of the iterations, are the overlap parameter α used in our cover construction and the specific local approximation spaces V_i. The results of [125] clearly show that the multilevel iteration given in Algorithm 4.4 with the local-to-local transfer operators of §5.2.1 and the BGS smoother yields a convergence rate (4.26) that is independent of the number $N = \mathrm{card}(C_\Omega)$ of cover patches $\omega_i \in C_\Omega$. However, the convergence behavior deteriorates with increasing polynomial degree $p = \max_{i=1,\dots N} p_i$ where $V_i \supset \mathcal{P}^{p_i}$. The convergence rate is dependent on p. Since the local error bounds of the local-to-local transfer operators are of order $p_i + 1$ this reduction in efficiency can be attributed to the smoother; i.e., we observe a deterioration of the smoothing property of the BGS iteration with increasing p.

One approach to overcome this limitation is the use of more involved smoothing schemes as it is proposed in [59]. Instead of employing a direct splitting into disjoint subspaces (5.18) we can consider overlapping subspaces. Here, a basis function $\varphi_i\vartheta_i^n$ may belong to several subspaces $\widetilde{\mathcal{V}}_l$. Consider for instance the overlapping subspace definition

$$\widetilde{\mathcal{V}}_l := \sum_{\omega_i \in C_l} \mathcal{V}_i = \mathrm{span}_{(i,n),i\in C_l}\langle \varphi_i\vartheta_i^n\rangle = \{v \in V^{\mathrm{PU}} \,|\, v = \sum_{\omega_i \in C_l} \varphi_i \sum_{n=1}^{d_i} v_i^n \vartheta_i^n\} \tag{5.19}$$

where $C_l \subset C_\Omega$ denotes the neighborhood (2.6) of the cover patch ω_l, see Figure 5.4 (right). The subspace $\widetilde{\mathcal{V}}_l$ contains all functions $\varphi_i\vartheta_i^n$ whose support ω_i has a non-vanishing intersection with the patch ω_l. Hence, when we solve the subspace problem $A|_{\widetilde{\mathcal{V}}_l}$ we resolve all couplings involving the basis functions $\varphi_l\vartheta_l^q$. Thus, for each patch ω_l there is one subspace problem $A|_{\widetilde{\mathcal{V}}_l}$ which resolves all couplings involving the associated basis functions $\varphi_l\vartheta_l^q$ independent of the overlap parameter α. In the following we refer to this iteration as a multiplicative overlapping Schwarz (MOS) smoother. Note that the splitting (5.19) is similar to the one employed in [112] where the issue of robustness with respect to the polynomial degree p is studied for higher order FEM.

The convergence rate of the resulting multilevel iteration is robust with respect to the polynomial degree p according to the results of [59]. However, the robustness is attained at a rather high price. Since the subspaces (5.19) involve all local approximation spaces V_i associated with the neighbors $\omega_j \in C_l$ their dimension is exponentially dependent on the space dimension D; i.e.

$$\dim(\widetilde{\mathcal{V}}_l) = O(3^D p^D) = O((3p)^D)$$

even for a uniform cover C_Ω.

Hence, the direct solution of $A|_{\widetilde{\mathcal{V}}_l}$ requires $C_{\mathrm{op}}^{\mathrm{MOS}} = O((3p)^{3D})$ operations and $C_{\mathrm{mem}}^{\mathrm{MOS}} = O((3p)^{2D})$ storage. Compared with the respective complexities $C_{\mathrm{op}}^{\mathrm{BGS}} = O(p^{3D})$ and $C_{\mathrm{mem}}^{\mathrm{BGS}} =$

$O(p^{2D})$ of the BGS scheme we find an increase by a factor of 3^{3D} in the number of operations and a factor of 3^{2D} with respect to storage demands. In essence, this means that the application of 3^{3D} iterations of the BGS smoother involves roughly the same number of operations as a single smoothing step of the MOS iteration; yet with 3^{-2D} times the storage cost. Thus, the question arises if the smoothing property of the BGS scheme and the overall convergence rate of our multilevel solver is robust with respect to the polynomial degree p (at least for a practical range of p) if we employ multiple iterations of the BGS smoother. The results presented in [125] indicate that this is the case. In fact we already obtain an optimal convergence rate of $\rho_a < 2^{-p-1}$, compare §4.4, with much less than 3^{3D} smoothing iterations.

Remark 5.3. A theoretical study closely related to the presented multiplicative multilevel iteration above is given in [34]. Our multilevel cover construction presented in §5.1 satisfies the assumptions of a sparse cover of [34]. The multilevel solvers presented in [32, 153] for the GFEM however are fundamentally different from our approach since they are based on classical finite element spaces.

Summary

Let us shortly summarize the complete construction of our multilevel iterative solver for the PPUM. Based on the sequence of covers C_Ω^k obtained by Algorithm 5.1 we define the respective sequence of PPUM space V_k^{PU}. For the transfer of information between these non-nested spaces we discretize the local-to-local L^2-projection of §5.2.1. As smoothing operators $S^{\mathrm{pre/post}}$ for the multilevel iteration of Algorithm 4.4 we employ the BGS iteration of Algorithm 5.2 with the splitting (5.18).

To further improve the speed of convergence we usually employ a Krylov method like the conjugate gradient (CG) or the generalized minimal residual (GMRES) method; i.e., we use our multilevel iteration as a preconditioner for the respective Krylov method. We obtain a good initial guess for our iterative solver via the nested iteration approach of Algorithm 4.5. Note that the outer Krylov iteration furthermore dampens the deterioration of the smoothing property with increasing polynomial degree p.

5.3 Hierarchical Enrichment

Let us now come back to the question of selecting problem-dependent local approximation spaces $V_{i,k}$ in the multilevel setting of our PPUM. Recall that the use of smooth polynomial local approximation spaces $V_{i,k} = \mathcal{P}^{p_{i,k}}$ in our PPUM is optimal only for the approximation of a smooth solution u. For the approximation of a discontinuous and singular solution u we augment a smooth local approximation space $\mathcal{P}^{p_{i,k}}$ by a problem-dependent enrichment space $\mathcal{E}_i$; i.e., we consider the local approximation spaces

$$V_{i,k} = \operatorname{span}\langle \vartheta_{i,k}^m \rangle = \mathcal{P}^{p_{i,k}} + \mathcal{E}_{i,k} = \operatorname{span}\langle \psi_{i,k}^s \rangle + \operatorname{span}\langle \eta_{i,k}^t \rangle.$$

We include problem-adapted enrichment functions $\eta_{i,k}^t$ in our PPUM space V^{PU} for two reasons: On the one hand, certain behavior (e.g. discontinuities) of the solution can be explicitly modeled by the functions $\eta_{i,k}^t$. More importantly, we can improve the asymptotic convergence behavior of our PPUM by completely resolving singular behavior of the solution u by the enrichment functions $\eta_{i,k}^t$. This property can also improve the robustness of a multilevel iterative solver since the corrections from level to level are essentially smooth functions.

Thus, we must be concerned with the approximation properties of the enriched spaces V_i and the stability of system $\langle \varphi_{i,k}\vartheta_{i,k}^m \rangle = \langle \varphi_{i,k}\psi_{i,k}^s, \varphi_{i,k}\eta_{i,k}^t \rangle$ which span the resulting PPUM space V^{PU}; i.e., with the condition number of the resulting stiffness matrix.

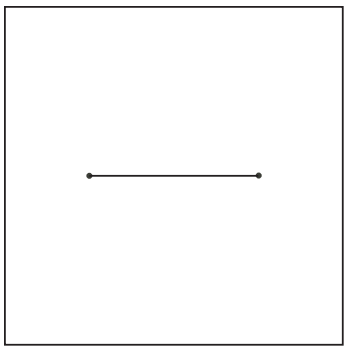


Figure 5.5. *Schematic of domain with center crack.*

Note that an enrichment approach is also pursued (in a similar way) in other meshfree methods [19, 109], the XFEM [16, 20, 100] or the GFEM [42, 44, 45]. Most of these enrichment schemes however focus on modeling issues only and are not concerned with the stability and the asymptotic convergence behavior of the respective numerical method or the impact on iterative solvers.

5.3.1 Enrichment Indicator

In this section we introduce an automatic hierarchical enrichment scheme for our multilevel PPUM that provides optimal convergence properties independent of the regularity of the solution and avoids an ill-conditioning of the resulting stiffness matrix due to enrichment. To this end, we consider a reference problem from linear elastic fracture mechanics

$$\begin{aligned} -\mathbf{div}\,\sigma(u) &= f && \text{in } \Omega = (-1,1)^2, \\ \sigma(u)\cdot n &= g_N && \text{on } \Gamma_N \subset \partial\Omega \cup C, \\ u &= g_D && \text{on } \Gamma_D = \partial\Omega \setminus \Gamma_N. \end{aligned} \tag{5.20}$$

The internal traction-free segment

$$C := \{(x,y) \in \Omega \,|\, x \in (-0.5, 0.5) \text{ and } y = 0\}$$

is referred to as a crack, see Figure 5.5. The crack C induces a discontinuous displacement field u across the crack line C with singularities at the crack tips $c_l := (-0.5, 0)$ and $c_u := (0.5, 0)$. Hence, the local approximation spaces $V_{i,k}$ employed in our PPUM must respect these features to provide good approximation.

A commonly used enrichment strategy employs simple geometric information only. Here, a patch $\omega_{i,k}$ (or an element in the XFEM or GFEM) is enriched by the discontinuous Heaviside function

$$H_{\pm}^{C}(x) = \begin{cases} 1 & \text{if } x \cdot n_C > 0 \\ -1 & \text{else} \end{cases}, \tag{5.21}$$

where n_C denotes the normal to the crack C, if the patch is (completely) cut by the crack C, i.e.

$$\mathcal{E}_{i,k} := H_{\pm}^{C}\mathcal{P}^{p_{i,k}} \quad \text{and} \quad V_{i,k} := \mathcal{P}^{p_{i,k}} + H_{\pm}^{C}\mathcal{P}^{p_{i,k}}. \tag{5.22}$$

Note that in fact most other enrichment procedures employ $\mathcal{E}_{i,k} = H_{\pm}^{C}$ only. If the patch $\omega_{i,k}$ contains a crack tip ξ_{tip}, i.e. $c_l \in \omega_{i,k}$ or $c_u \in \omega_{i,k}$, then the patch is enriched by the respective space of singular tip functions

$$W_{\text{tip}} := \text{span}\langle \sqrt{r}\cos\frac{\theta}{2}, \sqrt{r}\sin\frac{\theta}{2}, \sqrt{r}\sin\theta\sin\frac{\theta}{2}, \sqrt{r}\sin\theta\cos\frac{\theta}{2} \rangle \tag{5.23}$$

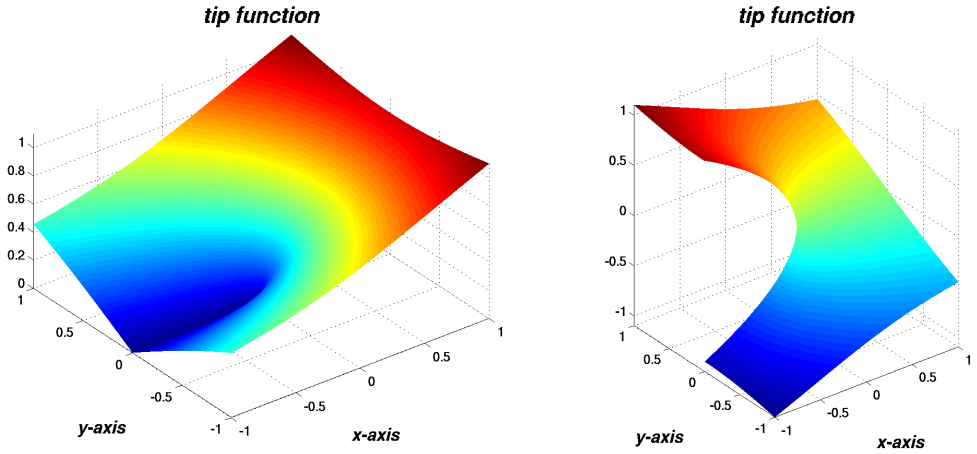


Figure 5.6. *Surface plots of the first two crack tip functions* (5.23).

given in local polar coordinates with respect to the tip ξ_{tip}, i.e. $\mathcal{E}_{i,k} = W_{\text{tip}}|_{\omega_{i,k}}$, compare Figures 5.6 and 5.7. This yields the local approximation space

$$V_{i,k} := \mathcal{P}^{p_{i,k}} + W_{\text{tip}}$$

for a patch $\omega_{i,k}$ that contains the tip ξ_{tip}. Let us summarize this geometric modeling enrichment scheme in the following classifier function $e^M : C^k_\Omega \to \{\texttt{lower\_tip}, \texttt{upper\_tip}, \texttt{jump}, \texttt{none}\}$

$$e^M(\omega_{i,k}) := \begin{cases} \texttt{lower\_tip} & \text{if } c_l \in \omega_{i,k} \text{ and } c_u \notin \omega_{i,k}, \\ \texttt{upper\_tip} & \text{if } c_l \notin \omega_{i,k} \text{ and } c_u \in \omega_{i,k}, \\ \texttt{jump} & \text{if } \{c_l, c_u\} \cap \omega_{i,k} = \emptyset \text{ and } C \cap \omega_{i,k} \neq \emptyset, \\ \texttt{none} & \text{else.} \end{cases} \tag{5.24}$$

The respective enrichment spaces are then given by

$$\mathcal{E}(e^M(\omega_{i,k})) := \begin{cases} W_{c_l} & \text{if } \texttt{lower\_tip} = e^M(\omega_{i,k}), \\ W_{c_u} & \text{if } \texttt{upper\_tip} = e^M(\omega_{i,k}), \\ H^C_\pm \mathcal{P}^{p_{i,k}} & \text{if } \texttt{jump} = e^M(\omega_{i,k}), \\ 0 & \text{else.} \end{cases} \tag{5.25}$$

Even though this enrichment scheme is sufficient to model a crack and captures the asymptotic behavior of the solution at the tip, this strategy suffers from various drawbacks. With respect to the discontinuous enrichment the main issue is that very small intersections of a patch with a crack cause an ill-conditioned stiffness matrix which can compromise the stability of the discretization; e.g. when the volumes of the sub-patches induced by the cut with the crack differ substantially in size. This is usually circumvented by a predefined geometric tolerance parameter which rejects such small intersections. In the case of a one-dimensional enrichment space $\mathcal{E}_{i,k} = H^C_\pm$ this approach is sufficient—if the tolerance parameter is chosen relative to the diameter of the patch. For a multi-dimensional enrichment space $\mathcal{E}_{i,k} = H^C_\pm \mathcal{P}^{p_{i,k}}$ as we employ in the PPUM, this simple geometric approach can be too restrictive to obtain optimal results or it may be too weak to maintain the stability.

The crack tip enrichment space W_{tip} given in (5.23) models the essential behavior of the solution at the tip. However, the singularity at the tip has a substantially larger zone of influence than

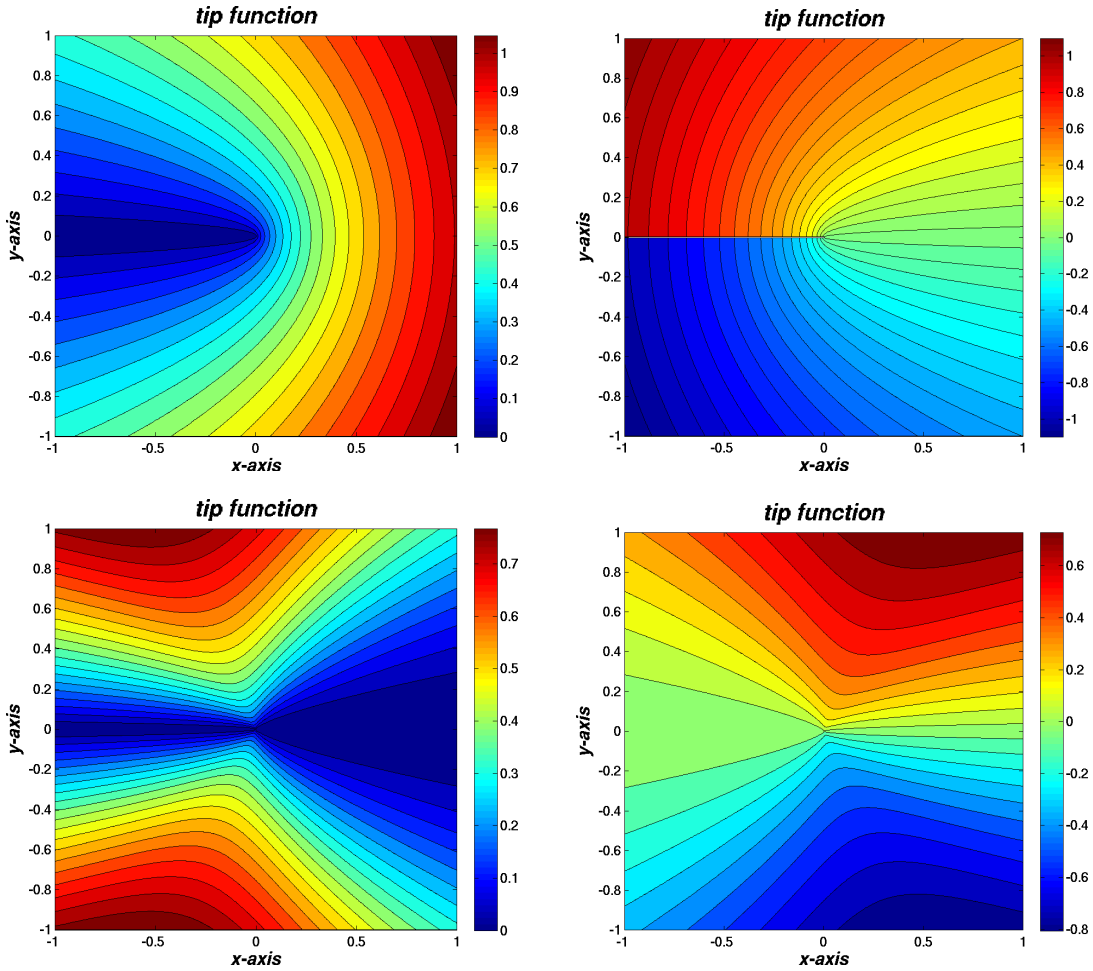


Figure 5.7. *Contour plots of the four crack tip functions* (5.23) *(upper left to lower right).*

just the containing patch $\omega_{i,k}$. Therefore, the simple geometric modeling enrichment (5.24) is not sufficient to improve the asymptotic convergence behavior of the employed numerical scheme.

With the help of our multilevel sequence of covers C_Ω^k we can easily generalize this enrichment scheme to overcome these issues and obtain a hierarchical enrichment scheme that yields enriched patches $\omega_{i,k}$ in an appropriate vicinity of the tips c_l and c_u of the crack C. To this end, we introduce the hierarchical enrichment classifier $e^H(\omega_{i,k})$ for all cover patches $\omega_{i,k}$ with $i = 1, \ldots, N_k$ and $k = 0, \ldots, J$ by the following algorithm. Note that we again make use of Lemma 5.1; i.e, of the geometric hierarchy of our cover patches, in Algorithm 5.3.

Algorithm 5.3 (Hierarchical Enrichment Classifier)**.**

1. Set $e^H(\omega_{i,k}) := \emptyset$.
2. If $k = 0$:
   (a) Set $E^H_{i,k} := \{e^M(\omega_{i,k})\}$.
   (b) Set $\mathcal{E}_{i,k} := \mathcal{E}(e^M(\omega_{i,k}))$.

Else:

(a) Get the unique $\omega_{\tilde{i},k-1} \in C_\Omega^{k-1}$ with $\omega_{\tilde{i},k-1} \supseteq \omega_{i,k} \in C_\Omega^k$.

(b) Set $E_{i,k}^H := E_{\tilde{i},k-1}^H \setminus \{\texttt{jump}\}$.

(c) Set $\mathcal{E}_{i,k} := \mathcal{E}_{\tilde{i},k-1} \setminus H_\pm^C \mathcal{P}^{p_{i,k}}$.

(d) If $\texttt{jump} \in E_{\tilde{i},k-1}^H$, $\{c_l, c_u\} \cap \omega_{i,k} = \emptyset$ and $C \cap \omega_{i,k} \neq \emptyset$:

i. Set $E_{i,k}^H = E_{i,k}^H \cup \{\texttt{jump}\}$.

ii. Set $\mathcal{E}_{i,k} = \mathcal{E}_{i,k} + H_\pm^C \mathcal{P}^{p_{i,k}}$.

3. Set $V_{i,k} := \mathcal{P}^{p_{i,k}} + \mathcal{E}_{i,k}$.

4. Set $e^H(\omega_{i,k}) = E_{i,k}^H$.

Observe that $e^H(\omega_{i,k}) \subset \{\texttt{lower\_tip}, \texttt{upper\_tip}, \texttt{jump}, \texttt{none}\}$ is a subset of the set of enrichment markers rather than a single marker as $e^M(\omega_{i,k})$. Thus, the respective enrichment spaces $\mathcal{E}_{i,k}$ constructed by Algorithm 5.3 are larger spaces than the modeling enrichment spaces of (5.25). For instance, we obtain the enrichment space

$$\mathcal{E}_{i,k} := W_{\text{tip}} + H_\pm^C \mathcal{P}^{p_{i,k}} \quad \text{so that} \quad V_{i,k} := \mathcal{P}^{p_{i,k}} + W_{\text{tip}} + H_\pm^C \mathcal{P}^{p_{i,k}} \tag{5.26}$$

for all patches $\omega_{i,k}$ with $e^H(\omega_{i,k}) = \{\texttt{jump}, \texttt{lower\_tip}\}$ or $e^H(\omega_{i,k}) = \{\texttt{jump}, \texttt{upper\_tip}\}$. Note that the recursive enrichment process defined in Algorithm 5.3 yields a constant enrichment zone

$$E_{\text{tip}} := \bigcap_{k=0}^{J \to \infty} E_{\text{tip},k} \quad \text{where} \quad E_{\text{tip},k} := \bigcup_{\substack{i=1 \\ \mathcal{E}_{i,k} \supset W_{\text{tip}}}}^{N_k} \omega_{i,k} \tag{5.27}$$

around each tip at c_l and c_u of the crack C. Hence, the singularities of the solution u are not only resolved by a constant number of patches $\omega_{i,k}$ on each level k but by an increasing number of patches on each level k. The singularities of the solution are resolved in the enrichment zones (5.27) on all levels k. Thus, our PPUM will converge with a rate that is independent of the regularity of the solution (i.e. regularity with respect to enrichment zones E_{tip}), see Theorem 5.1 below. However, this approach amplifies the ill-conditioning of the stiffness matrix due to enrichment since a large number of enrichment functions $\eta_{i,k}^t$ away from the singularities at c_l and c_u is employed.

5.3.2 Stability of Enriched Local Approximation Spaces

Observe that there are two sources of ill-conditioning in an enriched PPUM using the local approximation spaces $V_{i,k} = \mathcal{P}^{p_{i,k}} + \mathcal{E}_{i,k}$ with $\mathcal{P}^{p_{i,k}} = \text{span}\langle \psi_{i,k}^s \rangle$ and $\mathcal{E}_{i,k} = \text{span}\langle \eta_{i,k}^t \rangle$. First of all, the restrictions $\eta_{i,k}^t := \eta^t|_{\omega_{i,k}}$ of the global enrichment functions η^t to the local patches $\omega_{i,k}$ may be (almost) linearly dependent *locally* on $\omega_{i,k} \cap \Omega$ even though the global enrichment functions η^t are linearly independent (well-conditioned) *globally* on Ω. Furthermore, on patches $\omega_{i,k} \cap C = \emptyset$ the restrictions $\eta_{i,k}^t$ are regular functions and thus some elements $e_{i,k} \in \mathcal{E}_{i,k}$ may be approximated well by local polynomials $\psi_{i,k}^s$; i.e. $e_{i,k} \in \mathcal{P}^{p_{i,k}}$. Hence, the condition number associated with the system of functions $\langle \psi_{i,k}^s, \eta_{i,k}^t \rangle$ can deteriorate very rapidly.

In the following we present an efficient algebraic approach to overcome the ill-conditioning of the local basis (generating system) $\langle \psi_{i,k}^s, \eta_{i,k}^t \rangle$ (and ultimately of the resulting stiffness matrix)

due to enrichment where we drop the level subscript k for the ease of notation. The overall goal of our construction is to attain a direct splitting of a local approximation space

$$V_i = \mathcal{P}^{p_i} + \mathcal{E}_i = \mathcal{P}^{p_i} \oplus \mathcal{D}_i$$

into the smooth sub-space $\mathcal{P}^{p_i}$ and a pure enrichment space $\mathcal{D}_i := \mathcal{E}_i \setminus \mathcal{P}^{p_i}$. To this end let us introduce some short-hand notation and the spaces

$$\begin{aligned} \mathcal{P}^{p_i} &:= \operatorname{span}\langle \psi_i^s \rangle, \text{ with } d_i^{\mathcal{P}} := \dim(\mathcal{P}^{p_i}) = \operatorname{card}(\{\psi_i^s\}), \\ \mathcal{E}_i &:= \operatorname{span}\langle \hat{\eta}_i^t \rangle, \text{ with } d_i^{\mathcal{E}} := \dim(\mathcal{E}_i) \leq \operatorname{card}(\{\hat{\eta}_i^t\}) =: c_i^{\mathcal{E}}, \\ V_i &:= \mathcal{P}^{p_i} + \mathcal{E}_i, \text{ with } d_i := \dim(V_i) \leq d_i^{\mathcal{P}} + c_i^{\mathcal{E}} =: c_i; \end{aligned} \tag{5.28}$$

i.e., we assume that we have a stable basis $\langle \psi_i^s \rangle$ for $\mathcal{P}^{p_i}$ and only a generating system $\langle \hat{\eta}_i^t \rangle$ for $\mathcal{E}_i$. Thus, we also have a generating system $\langle \hat{\vartheta}_i^m \rangle := \langle \psi_i^s, \hat{\eta}_i^t \rangle$ for the local approximation space V_i on the respective patch ω_i only. Observe that this is a generalization of the situation considered in §4.2.1. Thus the preconditioner $S_{\mathcal{E}\setminus\mathcal{P}}$ constructed below serves as a replacement for the preconditioner S of §4.2.1.

With respect to the generating system $\langle \hat{\vartheta}_i^m \rangle = \langle \psi_i^s, \hat{\eta}_i^t \rangle$ we obtain the singular (ill-conditioned) local mass matrix M_i with entries $(M_i)_{n,m}$ on a particular patch ω_i by

$$(M_i)_{n,m} := \int_{\omega_i \cap \Omega} \hat{\vartheta}_i^n \hat{\vartheta}_i^m \, dx \quad \text{for all } m, n = 1, \ldots, c_i \tag{5.29}$$

which can be written in the block-form

$$M_i = \begin{pmatrix} M^i_{\mathcal{P},\mathcal{P}} & M^i_{\mathcal{P},\mathcal{E}} \\ M^i_{\mathcal{E},\mathcal{P}} & M^i_{\mathcal{E},\mathcal{E}} \end{pmatrix}$$

due to the additive representation $V_i = \mathcal{P}^{p_i} + \mathcal{E}_i$. For the ease of notation we furthermore drop the sub- and superscript i in the following; i.e., we consider the local mass matrix M on a particular patch ω_i in block-form

$$M := \begin{pmatrix} M_{\mathcal{P},\mathcal{P}} & M_{\mathcal{P},\mathcal{E}} \\ M_{\mathcal{E},\mathcal{P}} & M_{\mathcal{E},\mathcal{E}} \end{pmatrix} \tag{5.30}$$

where the blocks satisfy

$$M_{\mathcal{P},\mathcal{P}} \in \mathbb{R}^{d_i^{\mathcal{P}} \times d_i^{\mathcal{P}}}, \quad M_{\mathcal{E},\mathcal{E}} \in \mathbb{R}^{c_i^{\mathcal{E}} \times c_i^{\mathcal{E}}}, \quad M_{\mathcal{E},\mathcal{P}} = M_{\mathcal{P},\mathcal{E}}^T \in \mathbb{R}^{c_i^{\mathcal{E}} \times d_i^{\mathcal{P}}}. \tag{5.31}$$

Note that the linear dependencies among the enrichment functions $\hat{\eta}_i^t$ lead to a deterioration of the condition number of the block $M_{\mathcal{E},\mathcal{E}}$; i.e., the matrix $M_{\mathcal{E},\mathcal{E}}$ has a non-zero (numerical) kernel or (near-)null space. The elements of the (near-)null space are not necessary to construct any element of the function space $\mathcal{E}_i$ and hence can be eliminated. Thus, we look for a system of functions $\langle \tilde{\eta}_i^t \rangle$ that spans the space $\mathcal{E}_i$ and at the same time allows for a simple elimination of the kernel elements of $M_{\mathcal{E},\mathcal{E}}$. The eigenfunctions (eigenvectors) of $M_{\mathcal{E},\mathcal{E}}$ provide such a system $\langle \tilde{\eta}_i^t \rangle$. Hence, let us consider the eigenvalue decomposition

$$O_{\mathcal{E}}^T M_{\mathcal{E},\mathcal{E}} O_{\mathcal{E}} = D_{\mathcal{E}} \quad \text{with } O_{\mathcal{E}}, D_{\mathcal{E}} \in \mathbb{R}^{c_i^{\mathcal{E}} \times c_i^{\mathcal{E}}} \tag{5.32}$$

of the matrix $M_{\mathcal{E},\mathcal{E}}$ with the normal transformation $O_{\mathcal{E}}^T$ and the diagonal matrix $D_{\mathcal{E}}$, i.e.,

$$O_{\mathcal{E}}^T O_{\mathcal{E}} = \mathbb{I}_{c_i^{\mathcal{E}}}, \quad \text{and } (D_{\mathcal{E}})_{r,q} = 0 \quad \text{for all } r, q = 1, \ldots, c_i^{\mathcal{E}}, \text{ and } r \neq q.$$

From (5.32) we can easily identify the (numerical) kernel of $M_{\mathcal{E},\mathcal{E}}$ if we assume that the eigenvalues $(D_{\mathcal{E}})_{r,r}$ are given in decreasing order, i.e. $(D_{\mathcal{E}})_{m,m} \geq (D_{\mathcal{E}})_{m+1,m+1}$. Then the matrices $O_{\mathcal{E}}^T$ and $D_{\mathcal{E}}$ are partitioned as

$$O_{\mathcal{E}}^T = \begin{pmatrix} \tilde{O}_{\mathcal{E}}^T \\ K_{\mathcal{E}}^T \end{pmatrix}, \quad \text{and} \quad D_{\mathcal{E}} = \begin{pmatrix} \tilde{D}_{\mathcal{E}} & 0 \\ 0 & \kappa_{\mathcal{E}} \end{pmatrix} \tag{5.33}$$

where the rth row of the rectangular matrix $\tilde{O}_{\mathcal{E}}^T \in \mathbb{R}^{d_i^{\mathcal{E}} \times c_i^{\mathcal{E}}}$ is an eigenvector of $M_{\mathcal{E},\mathcal{E}}$ that is associated with an eigenvalue $(\tilde{D}_{\mathcal{E}})_{r,r}$ satisfying $(D_{\mathcal{E}})_{r,r} = (\tilde{D}_{\mathcal{E}})_{r,r} \geq \epsilon\, (\tilde{D}_{\mathcal{E}})_{0,0} = (D_{\mathcal{E}})_{0,0}$ and $K_{\mathcal{E}}^T$ involves all eigenvectors that are associated with small eigenvalues, i.e. $(\kappa_{\mathcal{E}})_{q,q} < \epsilon\, (\tilde{D}_{\mathcal{E}})_{0,0}$. Thus, the inverse of $\tilde{D}_{\mathcal{E}}$ can be computed stably and the operator

$$\tilde{D}_{\mathcal{E}}^{-1/2} \tilde{O}_{\mathcal{E}}^T : \mathcal{E} = \operatorname{span}\langle \hat{\eta}_i^t \rangle \to \mathcal{E} = \operatorname{span}\langle \tilde{\eta}_i^t \rangle$$

gives a stable basis $\langle \tilde{\eta}_i^t \rangle$ of the enrichment space $\mathcal{E}_i$ and there holds $d_i^{\mathcal{E}} = \dim(\mathcal{E}_i) = \operatorname{card}(\{\tilde{\eta}_i^t\}) \leq c_i^{\mathcal{E}} = \operatorname{card}(\{\hat{\eta}_i^t\})$. With respect to the basis $\langle \tilde{\eta}_i^t \rangle$ the block $M_{\mathcal{E},\mathcal{E}}$ of the local mass matrix M is given as

$$M_{\mathcal{E},\mathcal{E}}^* := \tilde{D}_{\mathcal{E}}^{-1/2} \tilde{O}_{\mathcal{E}}^T M_{\mathcal{E},\mathcal{E}} \tilde{O}_{\mathcal{E}} \tilde{D}_{\mathcal{E}}^{-1/2} = \mathbb{I}_{d_i^{\mathcal{E}}};$$

i.e., the operator $\tilde{D}_{\mathcal{E}}^{-1/2} \tilde{O}_{\mathcal{E}}^T$ yields an optimally conditioned orthonormal basis $\langle \tilde{\eta}_i^t \rangle$ for the enrichment space $\mathcal{E}_i$ on the patch ω_i. Applying the block-transformation

$$S_{\mathcal{E}} := \begin{pmatrix} \mathbb{I}_{d^{\mathcal{P}_i}} & 0 \\ 0 & \tilde{D}_{\mathcal{E}}^{-1/2} \tilde{O}_{\mathcal{E}}^T \end{pmatrix}$$

to the local mass matrix M of (5.30) we obtain

$$M_{\mathcal{E}} := S_{\mathcal{E}} M S_{\mathcal{E}}^T = \begin{pmatrix} M_{\mathcal{P},\mathcal{P}} & M_{\mathcal{P},\mathcal{E}}^* \\ M_{\mathcal{E},\mathcal{P}}^* & M_{\mathcal{E},\mathcal{E}}^* \end{pmatrix} \tag{5.34}$$

the local mass matrix with respect to the collection of functions $\langle \psi_i^s, \tilde{\eta}_i^t \rangle$ with the blocks

$$M_{\mathcal{P},\mathcal{E}}^* := (M_{\mathcal{E},\mathcal{P}}^*)^T, \qquad M_{\mathcal{E},\mathcal{P}}^* := \tilde{D}_{\mathcal{E}}^{-1/2} \tilde{O}_{\mathcal{E}}^T M_{\mathcal{E},\mathcal{P}}, \quad \text{and}$$

$$M_{\mathcal{E},\mathcal{E}}^* := \tilde{D}_{\mathcal{E}}^{-1/2} \tilde{O}_{\mathcal{E}}^T M_{\mathcal{E},\mathcal{E}} \tilde{O}_{\mathcal{E}} \tilde{D}_{\mathcal{E}}^{-1/2} = \mathbb{I}_{d_i^{\mathcal{E}}}.$$

Even though both bases $\langle \psi_i^s \rangle$ of $\mathcal{P}^{p_i}$ and $\langle \tilde{\eta}_i^t \rangle$ of $\mathcal{E}_i$ are stable their simple merger $\langle \psi_i^s, \tilde{\eta}_i^t \rangle$ may not be stable. That is we currently have a representation of the local approximation space

$$V_i = \mathcal{P}^{p_i} + \mathcal{E}_i = \operatorname{span}\langle \psi_i^s \rangle + \operatorname{span}\langle \tilde{\eta}_i^t \rangle$$

which is *not* a direct splitting; i.e., the spaces $\mathcal{P}^{p_i}$ and $\mathcal{E}_i$ overlap in general. In the following we construct a stable direct splitting of V_i where we separate the enrichment degrees of freedom from the polynomials completely; i.e., we derive the direct splitting

$$V_i = \mathcal{P}^{p_i} \oplus \mathcal{D}_i := \mathcal{P}^{p_i} \oplus \left(\mathcal{E}_i \setminus \mathcal{P}^{p_i} \right)$$

and compute an appropriate basis η_i for the space $\mathcal{D}_i := \mathcal{E}_i \setminus \mathcal{P}^{p_i}$ automatically. That is we eliminate all polynomial components in the enrichment space.[4] This is achieved by a block-elimination approach; i.e. the Schur complement.

[4] Since the matrix $M_{\mathcal{E},\mathcal{E}}^* = \mathbb{I}_{d_i^{\mathcal{E}}}$ is invertible due to our construction we can also eliminate the enrichment functions from the polynomials.

To this end, we consider the local mass matrix $M_{\mathcal{E}}$ in block-form (5.34) and since the basis of $\mathcal{P}^{p_i}$ is stable we can compute $(M_{\mathcal{P},\mathcal{P}})^{-1}$. Therefore the transformation

$$S_S := \begin{pmatrix} \mathbb{I}_{d_i^{\mathcal{P}}} & 0 \\ -M_{\mathcal{E},\mathcal{P}}^*(M_{\mathcal{P},\mathcal{P}})^{-1} & \mathbb{I}_{d_i^{\mathcal{E}}} \end{pmatrix}$$

from the system $\langle \psi_i^s, \tilde{\eta}_i^t \rangle$ to $\langle \psi_i^s, \bar{\eta}_i^t \rangle$ is stable and we obtain the block-diagonal local mass matrix

$$M_S := S_S M_{\mathcal{E}} S_S^T = \begin{pmatrix} M_{\mathcal{P},\mathcal{P}} & 0 \\ 0 & M_{\mathcal{D},\mathcal{D}} \end{pmatrix}$$

with

$$M_{\mathcal{D},\mathcal{D}} := M_{\mathcal{E},\mathcal{E}}^* - M_{\mathcal{E},\mathcal{P}}^*(M_{\mathcal{P},\mathcal{P}})^{-1} M_{\mathcal{P},\mathcal{E}}^*.$$

The matrix $M_{\mathcal{D},\mathcal{D}}$, i.e. the Schur complement matrix, is of dimension $d_i^{\mathcal{E}} \times d_i^{\mathcal{E}}$. However the dimension $d_i^{\mathcal{D}} := \dim(\mathcal{D}_i)$ of the space $\mathcal{D}_i = \mathcal{E}_i \setminus \mathcal{P}^{p_i} = \operatorname{span}\langle \bar{\eta}_i^t \rangle$ may be smaller. This is the case if a non-vanishing linear combination of the local restrictions η_i^t of the global enrichment functions η_t can be represented (or well-approximated) by polynomials. Thus, $M_{\mathcal{D},\mathcal{D}}$ in general has a (numerical) kernel and we need to eliminate these kernel elements to obtain a stable local mass matrix on ω_i. To this end, we employ the same approach as for the block $M_{\mathcal{E},\mathcal{E}}$. We compute the eigenvalue decomposition

$$O_{\mathcal{D}}^T M_{\mathcal{D},\mathcal{D}} O_{\mathcal{D}} = D_{\mathcal{D}} \quad \text{with } O_{\mathcal{D}}, D_{\mathcal{D}} \in \mathbb{R}^{d_i^{\mathcal{E}} \times d_i^{\mathcal{E}}} \tag{5.35}$$

of the Schur complement matrix $M_{\mathcal{D},\mathcal{D}}$ where

$$O_{\mathcal{D}}^T O_{\mathcal{D}} = \mathbb{I}_{d_i^{\mathcal{E}}}, \qquad (D_{\mathcal{D}})_{r,q} = 0 \quad \text{for all } r, q = 1, \dots, d_i^{\mathcal{E}}, \text{ and } r \neq q.$$

Again, we can easily identify the (numerical) kernel in the eigenbasis. Analogously to (5.33) we partition the matrices $O_{\mathcal{D}}^T$ and $D_{\mathcal{D}}$ of (5.35) as

$$O_{\mathcal{D}}^T = \begin{pmatrix} \tilde{O}_{\mathcal{D}}^T \\ K_{\mathcal{D}}^T \end{pmatrix}, \quad \text{and} \quad D_{\mathcal{D}} = \begin{pmatrix} \tilde{D}_{\mathcal{D}} & 0 \\ 0 & \kappa_{\mathcal{D}} \end{pmatrix} \tag{5.36}$$

where $\tilde{D}_{\mathcal{D}} \in \mathbb{R}^{d_i^{\mathcal{D}} \times d_i^{\mathcal{D}}}$ is invertible with maximal $d_i^{\mathcal{D}}$. Thus, we obtain the respective transformation to the eigenbasis $\langle \eta_i^t \rangle$ of the Schur complement $M_{\mathcal{D},\mathcal{D}}$ by

$$\tilde{D}_{\mathcal{D}}^{-1/2} \tilde{O}_{\mathcal{D}}^T : \mathcal{D}_i = \operatorname{span}\langle \bar{\eta}_i^t \rangle \to \mathcal{D}_i = \operatorname{span}\langle \eta_i^t \rangle.$$

Hence, the block-diagonal transformation

$$S_{\mathcal{D}} := \begin{pmatrix} \mathbb{I}_{d_i^{\mathcal{P}}} & 0 \\ 0 & \tilde{D}_{\mathcal{D}}^{-1/2} \tilde{O}_{\mathcal{D}}^T \end{pmatrix}$$

yields a symmetric positive definite local mass matrix $M_{\mathcal{D}}$

$$M_{\mathcal{D}} := S_{\mathcal{D}} M_S S_{\mathcal{D}}^T = \begin{pmatrix} M_{\mathcal{P},\mathcal{P}} & 0 \\ 0 & M_{\mathcal{D},\mathcal{D}}^* \end{pmatrix},$$

with $M_{\mathcal{D},\mathcal{D}}^* = \mathbb{I}_{d_i^{\mathcal{D}}}$; i.e., the system of functions $\langle \psi_i^s, \eta_i^t \rangle$ is a stable and well-conditioned basis of the local approximation space $V_i = \mathcal{P}^{p_i} + \mathcal{E}_i = \mathcal{P}^{p_i} \oplus \mathcal{D}_i$. The condition number of $M_{\mathcal{D}}$ is given by

the condition number of $M_{\mathcal{P},\mathcal{P}}$. There is no ill-conditioning due to enrichment and the presented construction yields an optimal local preconditioner. Moreover, there holds the representation

$$V_i = \mathcal{P}^{p_i} \oplus \mathcal{D}_i$$

as a direct sum, where the space $\mathcal{D}_i = \mathcal{E}_i \setminus \mathcal{P}^{p_i}$ consists of pure enrichment degrees of freedom that cannot be represented (or well-approximated) by polynomials.

The final local preconditioner $S_i^{\mathcal{E}\setminus\mathcal{P}} : V_i \to V_i$ which maps the original ill-conditioned local generating system $\langle \hat{\vartheta}_i^m \rangle = \langle \psi_i^s, \hat{\eta}_i^t \rangle$ to an optimally conditioned local basis $\langle \vartheta_i^m \rangle = \langle \psi_i^s, \eta_i^t \rangle$ of V_i is given by the product of the constructed transformations and projections

$$S_i^{\mathcal{E}\setminus\mathcal{P}} := S_{\mathcal{D}} \circ S_S \circ S_{\mathcal{E}}.$$

The respective global preconditioner $S_{\mathcal{E}\setminus\mathcal{P}}$ is given by

$$(S_{\mathcal{E}\setminus\mathcal{P}})_{i,j} := \begin{cases} S_i^{\mathcal{E}\setminus\mathcal{P}} & j = i, \\ 0 & j \neq i. \end{cases} \tag{5.37}$$

Due to the use of a flat top PU in our PPUM this local stability of $\langle \vartheta_i^m \rangle$ already yields stability of the global basis $\langle \varphi_i \vartheta_i^m \rangle$ of V^{PU}.[5]

Remark 5.4. Note that in the presentation above the equivalencies

$$V_i = \mathcal{P}^{p_i} \oplus \mathcal{D}_i = \mathcal{P}^{p_i} + \mathcal{E}_i$$

hold only up to the employed numerical cut-off parameter $\epsilon > 0$; i.e. we have

$$V_i := \mathcal{P}^{p_i} \oplus \mathcal{D}_i \approx \mathcal{P}^{p_i} + \mathcal{E}_i \quad \text{and} \quad \mathcal{D}_i \approx \mathcal{E}_i \setminus \mathcal{P}^{p_i}.$$

Remark 5.5. Even though we considered the identity operator $\mathbb{I}$ on the local patch ω_i, i.e. the local mass matrix M_i, we can construct the respective preconditioner for an arbitrary (definite) operator. A change of this operator impacts the absolute value of the constant condition number associated with the constructed basis only—in exact arithmetic and with $\epsilon = 0$. However, due to the employed cut-off parameter $\epsilon > 0$ we may obtain a (slightly) different space $\operatorname{span}\langle \vartheta_{i,k}^m \rangle$ for different operators with the same ϵ.

Let us summarize the presented local preconditioning approach on a particular patch $\omega_i \in C_\Omega$ by the following algorithm. Here, we include also a transformation for the space $\mathcal{P}^{p_i} = \operatorname{span}\langle \psi_i^s \rangle$ to obtain an orthonormal polynomial basis $\langle \psi_i^s \rangle$. Thus, we use Algorithm 5.4 for all patches $\omega_{i,k}$ not only the enriched patches. Observe that due to this preconditioning we also eliminate the stability problems encountered due to very small intersections $\omega_i \cap \Omega$. Note further that the constructed preconditioner $S_{\mathcal{E}\setminus\mathcal{P}}$ here replaces the preconditioner S of §4.2.1.

Algorithm 5.4 (Local Enrichment Preconditioning)**.**

1. Assemble the local matrix M on ω_i using the generating system $\langle \hat{\vartheta}_i^m \rangle = \langle \psi_i^s, \hat{\eta}_i^t \rangle$ analogously to (5.29) using the weak formulation of the considered operator, see Remark 5.5. Define the respective sub-matrices $M_{\mathcal{P},\mathcal{P}}$, $M_{\mathcal{E},\mathcal{E}}$, $M_{\mathcal{E},\mathcal{P}}$ of (5.31) due to the partitioning (5.30).
2. Compute the eigenvalue decomposition (5.32) of the block $M_{\mathcal{E},\mathcal{E}}$ and define the respective sub-matrices $\tilde{O}_{\mathcal{E}}^T$ and $\tilde{D}_{\mathcal{E}}$ according to (5.33) such that $\tilde{D}_{\mathcal{E}}$ is invertible.

[5] Formally, we need to compute the entries (5.29) with respect to $\omega_{\mathrm{FT},i}$.

3. Compute the eigenvalue decomposition

$$O_{\mathcal{P}}^T M_{\mathcal{P},\mathcal{P}} O_{\mathcal{P}} = D_{\mathcal{P}} \quad \text{with } O_{\mathcal{P}}, D_{\mathcal{P}} \in \mathbb{R}^{d_i^{\mathcal{P}} \times d_i^{\mathcal{P}}} \tag{5.38}$$

of the block $M_{\mathcal{P},\mathcal{P}}$ with the normal transformation $O_{\mathcal{P}}^T$ and the invertible diagonal matrix $D_{\mathcal{P}}$.

4. Define the block-diagonal transformation

$$S_{\mathcal{P},\mathcal{E}} := \begin{pmatrix} D_{\mathcal{P}}^{-1/2} O_{\mathcal{P}}^T & 0 \\ 0 & \tilde{D}_{\mathcal{E}}^{-1/2} \tilde{O}_{\mathcal{E}}^T \end{pmatrix}$$

from $V_i = \operatorname{span}\langle \hat{\psi}_i^s, \hat{\eta}_i^t \rangle$ to $V_i = \operatorname{span}\langle \psi_i^s, \tilde{\eta}_i^t \rangle$.

5. Define the sub-matrix

$$M_{\mathcal{E},\mathcal{P}}^* := \tilde{D}_{\mathcal{E}}^{-1/2} \tilde{O}_{\mathcal{E}}^T M_{\mathcal{E},\mathcal{P}} O_{\mathcal{P}} D_{\mathcal{P}}^{-1/2} \tag{5.39}$$

of the transformed local matrix $T_B^T M T_B$.

6. Define the block-triangular transformation

$$S_S := \begin{pmatrix} \mathbb{I}_{d_i^{\mathcal{P}}} & 0 \\ -M_{\mathcal{E},\mathcal{P}}^* & \mathbb{I}_{d_i^{\mathcal{E}}} \end{pmatrix}$$

from $V_i = \operatorname{span}\langle \psi_i^s, \tilde{\eta}_i^t \rangle$ to $V_i = \operatorname{span}\langle \psi_i^s, \tilde{\eta}_i^t \rangle$.

7. Define the Schur complement matrix

$$\begin{aligned} M_{\mathcal{D},\mathcal{D}} &:= \mathbb{I}_{d_i^{\mathcal{E}}} - M_{\mathcal{E},\mathcal{P}}^* M_{\mathcal{P},\mathcal{E}}^* \\ &= \mathbb{I}_{d_i^{\mathcal{E}}} - \tilde{D}_{\mathcal{E}}^{-1/2} \tilde{O}_{\mathcal{E}}^T M_{\mathcal{E},\mathcal{P}} O_{\mathcal{P}} D_{\mathcal{P}}^{-1} O_{\mathcal{P}}^T M_{\mathcal{P},\mathcal{E}} \tilde{O}_{\mathcal{E}} \tilde{D}_{\mathcal{E}}^{-1/2}. \end{aligned}$$

8. Compute the eigenvalue decomposition (5.35) of the Schur complement matrix $M_{\mathcal{D},\mathcal{D}}$ and define the respective sub-matrices $\tilde{O}_{\mathcal{D}}^T$ and $\tilde{D}_{\mathcal{D}}$ according to (5.36) such that $\tilde{D}_{\mathcal{D}}$ is invertible.

9. Define the block-diagonal transformation

$$S_{\mathcal{D}} := \begin{pmatrix} \mathbb{I}_{d_i^{\mathcal{P}}} & 0 \\ 0 & \tilde{D}_{\mathcal{D}}^{-1/2} \tilde{O}_{\mathcal{D}}^T \end{pmatrix}$$

from $V_i = \operatorname{span}\langle \psi_i^s, \tilde{\eta}_i^t \rangle$ to $V_i = \operatorname{span}\langle \psi_i^s, \eta_i^t \rangle$.

10. Define the local preconditioner $S_i^{\mathcal{E}\setminus\mathcal{P}}$ on ω_i as $S_i^{\mathcal{E}\setminus\mathcal{P}} := S_{\mathcal{D}} \circ S_S \circ S_{\mathcal{P},\mathcal{E}}$, i.e.,

$$S_i^{\mathcal{E}\setminus\mathcal{P}} := \begin{pmatrix} D_{\mathcal{P}}^{-1/2} O_{\mathcal{P}}^T & 0 \\ -\tilde{D}_{\mathcal{D}}^{-1/2} \tilde{O}_{\mathcal{D}}^T M_{\mathcal{E},\mathcal{P}}^* D_{\mathcal{P}}^{-1/2} O_{\mathcal{P}}^T & \tilde{D}_{\mathcal{D}}^{-1/2} \tilde{O}_{\mathcal{D}}^T \tilde{D}_{\mathcal{E}}^{-1/2} \tilde{O}_{\mathcal{E}}^T \end{pmatrix} \tag{5.40}$$

with $M_{\mathcal{E},\mathcal{P}}^*$ given in (5.39), which maps the ill-conditioned generating system $\langle \hat{\psi}_i^s, \hat{\eta}_i^t \rangle$ to the optimally conditioned basis $\langle \psi_i^s, \eta_i^t \rangle$.

The respective global preconditioner $S_{\mathcal{E}\setminus\mathcal{P}}$ is then obtained by (5.37) and the stiffness matrix with respect to the computed stable global basis $\langle \varphi_i \vartheta_i^m \rangle = \langle \varphi_i \psi_i^s, \varphi_i \eta_i^t \rangle$ is obtained as the triple-product

$$A_{\mathcal{E}\setminus\mathcal{P}} := S_{\mathcal{E}\setminus\mathcal{P}} A S_{\mathcal{E}\setminus\mathcal{P}}^T$$

where A denotes the stiffness matrix with respect to the generating system $\langle \varphi_i \psi_i^s, \varphi_i \hat{\eta}_i^t \rangle$. The computational complexity C_P of this preconditioning scheme is given by

$$C_P := O\Big(N\big((d_i^{\mathcal{P}})^3 + (c_i^{\mathcal{E}})^3 + (d_i^{\mathcal{E}})^3\big)\Big).$$

The presented approach scales linearly in the number of patches N and with respect to the number of operations per patch ω_i the presented preconditioner is cheaper than that of §4.2.1 which scales as $O(N(d_i^{\mathcal{P}} + c_i^{\mathcal{E}})^3)$. Moreover, the preconditioner $S_{\mathcal{E}\setminus\mathcal{P}}$ preserves the separability of the degrees of freedom into smooth approximation functions ψ_i^s and pure enrichment functions η_i^t which simplifies the error analysis of our hierarchically enriched PPUM summarized in Theorem 5.1 below.

Recall that we assumed the availability of analytical (singular) enrichment functions a priori in the construction above. Note however that we may also employ numerical enrichment functions often referred to as handbook functions [136, 137] which are e.g. pre-computed discrete approximations to certain reference problems. Since the solutions of such reference problems are singular functions we need to employ an adaptive numerical scheme for their approximation. The construction of an hp-adaptive PPUM which is suitable for the efficient computation of numerical enrichment functions is the subject of §5.4. But before we focus on the important issue of adaptivity for the PPUM let us close this section on enrichment with a refined error analysis for the presented hierarchically enriched PPUM.

5.3.3 Error Analysis

One of the goals of the presented enrichment scheme was to obtain a numerical method which converges with a rate that is independent of the regularity of the solution u. The following theorem shows that we achieve this goal with our hierarchical enrichment scheme. Moreover, we attain a kind of superconvergence in the enrichment zone (5.27). The local error in the enrichment zone E which contains all singularities of u decays faster than the global error on Ω.

Theorem 5.1. *Let $\Omega \subset \mathbb{R}^D$ with Lipschitz boundary $\partial\Omega$ be given. Let $\{\varphi_i \,|\, i = 1, \dots, N\}$ with $\omega_i := \operatorname{supp}(\varphi_i)$ and $\operatorname{diam}(\omega_i) \asymp h$ be a partition of unity which satisfies Definition 3.1 and the flat top property according to Definition 3.2. Let a collection of local approximation spaces $V_i := \mathcal{P}^p \oplus \mathcal{D}_i \subset H^1(\Omega \cap \omega_i)$ with a stable basis $\langle \psi_i^s, \eta_i^t \rangle$ be given for all $i = 1, \dots, N$ such that $\mathcal{D}_i = \{0\}$ for $i = 1, \dots, M-1$. Assume that the spaces $\mathcal{D}_i = \operatorname{span}\langle \eta_i^t \rangle$ for $i = M, \dots, N$ resolve all singularities of the solution $u \in H^1(\Omega)$; i.e., there holds the decomposition*

$$u = u_p + \tilde{\chi}_E u_s \quad \text{with} \quad u_p \in H^k(\Omega) \text{ with } k > 1 \tag{5.41}$$

and a mollified characteristic function $\tilde{\chi}_E \in C^\infty$ of the enrichment zone

$$E := \bigcup_{i=M}^{N} \omega_i \quad \text{such that} \quad \tilde{\chi}_E \sum_{i=1}^{M-1} \varphi_i \equiv 0 \tag{5.42}$$

and $\tilde{E} := \operatorname{supp}(\tilde{\chi}_E) \subset E$. Then the basis $\langle \varphi_i \psi_i^s, \varphi_i \eta_i^t \rangle$ of the space

$$V^{\mathrm{PU}} := \sum_{i=1}^{N} \varphi_i V_i$$

is stable and there hold the estimates

$$\begin{aligned} \|u^{\mathrm{PU}} - u\|_{H^1(\Omega)} &\leq O(h^{\min\{p,k\}}), \\ \|u^{\mathrm{PU}} - u\|_{H^1(\Omega\setminus\tilde{E})} &\leq O(h^{\min\{p,k\}}), \\ \|u^{\mathrm{PU}} - u\|_{H^1(E)} &\leq O(h^{\min\{p,k\}+\delta}), \end{aligned}$$

for the unique best approximation $u^{\mathrm{PU}} \in V^{\mathrm{PU}}$

$$u^{\mathrm{PU}} := \sum_{i=1}^{N} \varphi_i v_i = \sum_{i=1}^{N} \varphi_i \sum_{s=1}^{d_i^{\mathcal{P}}} v_i^s \psi_i^s + \sum_{t=1}^{d_i^{\mathcal{D}}} v_i^{t+d_i^{\mathcal{P}}} \eta_i^t$$

with $\delta = \delta(\{\mathcal{D}_i\}) > 0$.

Proof. Multiplication of the splitting (5.41) with $1 \equiv \sum_{i=1}^{N} \varphi_i$ yields

$$u = \sum_{i=1}^{N} \varphi_i u_p + \sum_{i=1}^{N} \varphi_i \tilde{\chi}_E u_s$$

and since an arbitrary PPUM function u^{PU} is given as

$$u^{\mathrm{PU}} = \sum_{i=1}^{M-1} \varphi_i \omega_i + \sum_{i=M}^{N} \varphi_i (\omega_i + e_i)$$

with $\omega_i \in \mathcal{P}^{p_i}$ and $e_i \in \mathcal{D}_i$ we obtain the error between the analytic solution u and our PPUM approximation u^{PU} by

$$u^{\mathrm{PU}} - u = \sum_{i=1}^{M-1} \varphi_i (\omega_i - u_p) + \sum_{i=M}^{N} \varphi_i ((\omega_i + e_i) - (u_p + \tilde{\chi}_E u_s)). \tag{5.43}$$

Appealing to the triangle inequality we obtain the estimate

$$\begin{aligned} \|u - u^{\mathrm{PU}}\| \leq\ & \|\sum_{i=1}^{M-1} \varphi_i (\omega_i - u_p)\| \\ & + \|\sum_{i=M}^{N} \varphi_i ((\omega_i + e_i) - (u_p + \tilde{\chi}_E u_s))\|. \end{aligned} \tag{5.44}$$

The first term on the right-hand side corresponds to the error of a PPUM approximation of the regular function $u_p \in H^k$ with polynomial local approximation spaces $V_i = \mathcal{P}^p$. According to Theorem 3.1 we can bound this error-term by $O(h^{\min\{p,k\}})$ in the H^1-norm, i.e.

$$\|\sum_{i=1}^{M-1} \varphi_i (\omega_i - u_p)\|_{H^1} \leq O(h^{\min\{p,k\}}). \tag{5.45}$$

To obtain an upper bound for the second term of the right-hand side of (5.44)

$$J_E := \|\sum_{i=M}^{N} \varphi_i ((\omega_i + e_i) - (u_p + \tilde{\chi}_E u_s))\|$$

we consider the equality

$$u_p + \tilde{\chi}_E u_s = u_p + (\tilde{\chi}_E - 1)u_s + u_s.$$

Here the triangle inequality yields an upper bound of J_E by

$$\begin{aligned} J_E &= \| \sum_{i=M}^{N} \varphi_i((\varpi_i + e_i) - (u_p + (\tilde{\chi}_E - 1)u_s + u_s))\| \\ &\leq \| \sum_{i=M}^{N} \varphi_i(\varpi_i - (u_p + (\tilde{\chi}_E - 1)u_s))\| \\ &\quad + \| \sum_{i=M}^{N} \varphi_i(e_i - u_s)\|. \end{aligned}$$

Observe that the function $u_p + (\tilde{\chi}_E - 1)u_s$ is regular since $\tilde{\chi}_E = 1$ in the vicinity of the singular points of u_s. Hence, we can bound the first term on the right-hand side again by $O(h^{\min\{p,k\}})$, i.e.,

$$\| \sum_{i=M}^{N} \varphi_i(\varpi_i - (u_p + (\tilde{\chi}_E - 1)u_s))\|_{H^1} \leq O(h^{\min\{p,k\}}).$$

Assuming that the enrichment functions resolve the singular part u_s of the solution u we can choose $e_i = u_s$ so that the second term vanishes and we obtain the upper bound

$$\| \sum_{i=M}^{N} \varphi_i((\varpi_i + e_i) - \tilde{\chi}_E(u_p + u_s))\|_{H^1} \leq O(h^{\min\{p,k\}}) \tag{5.46}$$

for the error in $\operatorname{supp}(\tilde{\chi}_E) \subset E$. Together with (5.45) this yields the error bound

$$\|u - u^{\mathrm{PU}}\|_{H_1} \leq O(h^{\min\{p,k\}})$$

for the global error on the domain Ω. Within the enrichment zone $\tilde{E}$ however we can obtain a better estimate. To this end, consider the case $u_s = 0$, i.e., the approximation of a regular solution $u = u_p$ by an enriched PPUM. Then, J_E becomes

$$J_E = \| \sum_{i=M}^{N} \varphi_i((\varpi_i + e_i) - u_p))\| \tag{5.47}$$

and the standard error bound $O(h^{\min\{p,k\}})$ ignores all degrees of freedom collected in e_i which are associated with the enrichment functions $\eta_i^t \in \mathcal{D}_i$. Outside of the singular points of u these functions are regular and cannot be well-approximated by the polynomials $\psi_i^s \in \mathcal{P}^p$ due to our construction. Hence, the functions $\eta_i^t \in \mathcal{D}_i$ provide additional approximation power to $V_i = \mathcal{P}^p \oplus \mathcal{D}_i$ even for the approximation of smooth functions. Therefore, (5.47) can in fact be bounded by $O(h^{\min\{p,k\}+\delta})$ in the H^1-norm with $\delta > 0$ for regular solutions $u = u_p$.

For a singular solution u, i.e. $u_s \neq 0$, we can utilize this observation by considering the splitting of the enrichment part e_i on a particular patch ω_i in two local components $e_i = e_i^s + e_i^p$. On each patch ω_i with $i = M, \ldots, N$ this splitting can be chosen to balance the two error terms on

the right-hand side of the inequality

$$\begin{aligned} J_E &= \|\sum_{i=M}^{N} \varphi_i((\omega_i + e_i^p + e_i^s) - (u_p + \tilde{\chi}_E u_s))\| \\ &\leq \|\sum_{i=M}^{N} \varphi_i((\omega_i + e_i^p) - (u_p + (\tilde{\chi}_E - 1)u_s))\| \\ &\quad + \|\sum_{i=M}^{N} \varphi_i(e_i^s - u_s)\|. \end{aligned} \tag{5.48}$$

This can yield a much smaller error bound since the regular function $u_p + (\tilde{\chi}_E - 1)u_s$ is now approximated by more degrees of freedom, i.e., by all polynomials and a number of enrichment functions η_i^t on ω_i. Yet, at the expense that $e_i^s - u_s \neq 0$. Hence, the Galerkin solution which minimizes J_E (i.e. minimizes (5.43) with respect to the energy-norm) shows a better convergence of $O(h^{\min\{p,k\}+\delta})$ with $\delta > 0$ in the enrichment zone than the global $O(h^{\min\{p,k\}})$ behavior. □

The absolute value of δ is of course dependent on the specific original enrichment functions η^t and the polynomial degree p employed since the constructed pure enrichment space is $\mathcal{D}_i = \mathcal{E}_i \setminus \mathcal{P}^{p_i}$. Note that δ is *not* related to the capability of the global enrichment functions η^t to resolve a specific singularity but rather to the approximation property of the localized enrichment functions η_i^t for regular solutions. The approximation property of the enrichment functions with respect to the singularities of the solution is required to bound the second term of the right-hand side of (5.48) and for the validity of the decomposition (5.41) with $\tilde{\chi}_E$ satisfying (5.42).

The impact of this improved convergence in the enrichment zone is that we can extract the coefficients of the asymptotic expansion of the singular solution u, the so-called stress intensity factors [140], with higher accuracy in the enrichment zone E and therefore near the singularities of u than the global error evolution implies.

5.4 Adaptive Refinement

If no analytical enrichment information is available a priori we must resort to a more classical approach to make the convergence behavior of our PPUM independent of the regularity of the solution u—adaptive hp-refinement. The fundamental prerequisite of any adaptive computation is local error estimation [10, 13, 111, 138, 145, 157] to determine the part of the domain Ω where the resolution of the approximation space V^{PU} is not sufficient to provide a uniform distribution of the error on Ω. Within our PPUM such estimates allow us to identify the cover patches ω_i and the respective local approximation spaces V_i which should be refined. Here the question arises whether it is more appropriate to refine the cover patch ω_i (h-refinement) or if it is more efficient to refine the local approximation space V_i (p-refinement). To answer this question, we employ a simple extrapolation technique for the local error estimates [99]. Thus, we construct a refinement indicator function

$$r_S : \{(\omega_i, C_i, V_i)\} \rightarrow \{\texttt{null}, \texttt{h}, \texttt{p}, \texttt{hp}\} \tag{5.49}$$

which marks the respective component(s) of our PPUM space on the finest level J

$$V_J^{\mathrm{PU}} = \sum_{i=1}^{N_J} \varphi_{i,J} V_{i,J},$$

where $N_J := \mathrm{card}(C_\Omega^J)$ and $\omega_{i,J} = \mathrm{supp}(\varphi_{i,J})$, for refinement from a single local error estimate, see also [65, 128]. To maintain the usefulness of the error bounds of Theorem 3.1 and Theorem 5.1

it is essential that the constructed sequence of refined PPUM spaces admit uniform bounds of the constants of Definition 3.1.

In the following we drop the level subscript k, since the adaptive refinement will always be carried out for the currently finest level J of our sequence V_k^{PU} with $k = 0, \ldots, J$.

5.4.1 Error Estimation

A very natural approach to error estimation in a PUM is the so-called subdomain estimator [9, 10, 65, 128, 138]. Here, the error equation is localized to certain subdomains $\Omega_i \subset \Omega$ and then (approximately) solved. The energy-norm of such a local solution on Ω_i provides a good estimate of the energy-norm with respect to Ω_i of the error. For a detailed construction of this subdomain error estimator let us consider the simple Poisson problem

$$\begin{aligned} -\Delta u &= f && \text{in } \Omega \subset \mathbb{R}^D, \\ u &= g_D && \text{on } \Gamma_D \subset \partial\Omega, \\ \frac{\partial u}{\partial n} &= g_N && \text{on } \Gamma_N = \partial\Omega \setminus \Gamma_D. \end{aligned} \tag{5.50}$$

Due to our algebraic construction of conforming local approximation spaces V_i presented in §4.2.2 we can discretize (5.50) via the classical weak formulation

$$a(u,v) = \langle \nabla u, \nabla v \rangle_{L^2(\Omega)} = \langle l, v \rangle = \langle f, v \rangle_{L^2(\Omega)} - \langle g_N, v \rangle_{L^2(\Gamma_N)}$$

for all $v \in H^1_{\Gamma_D,0}(\Omega) := \{v \in H^1(\Omega) \,|\, v = 0 \text{ on } \Gamma_D\}$. Thus, our conforming PPUM approximation u^{PU} satisfies

$$a(u^{\mathrm{PU}}, v) = \langle l, v \rangle \quad \text{for all } v \in C_K(V^{\mathrm{PU}}) \subset H^1_{\Gamma_D,0}(\Omega) \tag{5.51}$$

with the projection $C_K : V^{\mathrm{PU}} \to V_K^{\mathrm{PU}}$ of §4.2.2. For each patch ω_i let us introduce the local problem: Find $e_i^{\mathrm{PU}} \in T_{i,\Gamma_D,g_D}$ such that

$$a(e_i^{\mathrm{PU}}, w_i) = a(u - u^{\mathrm{PU}}, w_i) = a(e^{\mathrm{PU}}, w_i) \quad \text{holds for all } w_i \in T_{i,0}(\omega_i \cap \Omega) \tag{5.52}$$

where $T_{i,0} : \{w_i \in H^1(\omega_i \cap \Omega) \,|\, w_i = 0 \text{ on } \partial(\omega_i \cap \Omega) \setminus \Gamma_N\}$. The respective strong formulation of (5.52) is given by

$$\begin{aligned} -\Delta e_i^{\mathrm{PU}} &= (f + \Delta u^{\mathrm{PU}})|_{\omega_i} && \text{in } \omega_i \cap \Omega \subset \mathbb{R}^d, \\ e_i^{\mathrm{PU}} &= 0 && \text{on } \partial(\omega_i \cap \Omega) \setminus \Gamma_N \setminus \Gamma_D, \\ e_i^{\mathrm{PU}} &= (g_D - u^{\mathrm{PU}})|_{\omega_i} && \text{on } \omega_i \cap \Gamma_D, \\ \frac{\partial e_i^{\mathrm{PU}}}{\partial n} &= (g_N - \frac{\partial u^{\mathrm{PU}}}{\partial n})|_{\omega_i} && \text{on } \omega_i \cap \Gamma_N, \end{aligned} \tag{5.53}$$

compare [9–11, 65, 111, 128, 138].

The following theorem from [9] shows that we obtain a reliable and efficient error estimator from the local solutions e_i^{PU} of (5.53) for each $\omega_i \in C_\Omega$.

Theorem 5.2. *Let $\Omega \subset \mathbb{R}^D$ with Lipschitz boundary $\partial\Omega$ be given. Let $\{\varphi_i \,|\, i = 1, \ldots, N\}$ with $\omega_i := \mathrm{supp}^\circ(\varphi_i)$ be a partition of unity which satisfies Definition 3.1 and the flat top property according to Definition 3.2. Let a collection of local approximation spaces V_i with a respective direct splitting $V_i = V_{i,K} \oplus V_{i,I}$ such that $v|_{\Gamma_D} = 0$ for all $v \in V_{i,K}$. Furthermore, assume that the tuples $\{(\omega_i, V_{i,K})\}$ satisfy the uniform Poincaré property of Definition 5.1. Then there hold the estimates*

$$C^{-1} \left(\sum_{i=1}^N \|e_i^{\mathrm{PU}}\|^2_{H^1(\omega_i \cap \Omega)} \right)^{1/2} \leq \|u - u^{\mathrm{PU}}\|_{H^1(\Omega)} \leq C \left(\sum_{i=1}^N \|e_i^{\mathrm{PU}}\|^2_{H^1(\omega_i \cap \Omega)} \right)^{1/2} \tag{5.54}$$

for a constant $C > 0$ where u^{PU} is the approximate solution to (5.50) in $V^{\mathrm{PU}} = \sum_{i=1}^{N} \varphi_i V_i$ and e_i^{PU} is the solution to (5.53), i.e. u^{PU} satisfies (5.51) and e_i^{PU} (5.52).

Definition 5.1. A collection of tuples $\{(\omega_i, V_i)\}$ with $i = 1, \dots, N$ has the uniform Poincaré property with respect to Ω, $\Gamma_D \subset \partial\Omega$, and $\Gamma_N = \partial\Omega \setminus \Gamma_D$ if there exists a constant $C_P > 0$ independent of i such that for $\omega_i \cap \Gamma_D = \emptyset$ the inequality

$$\inf_{\lambda \in \mathbb{R}} \|v - \lambda\|_{L^2(\omega_i \cap \Omega)} \leq C_P \operatorname{diam}(\omega_i) \|v\|_{H^1(\omega_i \cap \Omega)}$$

holds for all $v \in H^1(\omega_i \cap \Omega)$ and $1 \in V_i$. In the case $\omega_i \cap \Gamma_D \neq \emptyset$ there holds the inequality

$$\|v\|_{L^2(\omega_i \cap \Omega)} \leq C_P \operatorname{diam}(\omega_i) \|v\|_{H^1(\omega_i \cap \Omega)}$$

for all $v \in H^1(\omega_i \cap \Omega)$ with $v = 0$ on Γ_D.

For the proof of Theorem 5.2 we refer to [9] where also the constant C of (5.54) is explicitly given. The uniform Poincaré property of Definition 5.1 is essentially a geometric condition on the intersections $\omega_i \cap \Omega$.

In practice, however, we can only approximate the solution e_i^{PU} of (5.53) on each patch ω_i and the bounds (5.54) are not guaranteed for an approximation of e_i^{PU}. Here, we choose the local approximation spaces

$$W_{i,q} := \varphi_i V_{i,q}^* \quad \text{with} \quad V_{i,q}^* := \mathcal{P}^{p_i+q} \oplus (\mathcal{E}_i \setminus \mathcal{P}^{p_i+q}) \tag{5.55}$$

with $q > 0$ for the approximate solution of (5.53). Observe that due to our hierarchical enrichment scheme and the multiplicative construction of the enrichment spaces (5.22) the spaces $V_{i,q}^*$ have better approximation properties than V_i not only for regular errors e_i^{PU} but also for discontinuous e_i^{PU}.

We discretize the operators on the right-hand side of (5.53) via a Petrov-Galerkin approach using V^{PU} as the trial space and $W_{i,q}$ as the test space. Thus, (5.52) becomes

$$a(e_{i,q}^{\mathrm{PU}}, \varphi_i v_{i,q}^*) = \langle l, \varphi_i v_{i,q}^* \rangle - a(u^{\mathrm{PU}}, \varphi_i v_{i,q}^*)$$

for all $v_{i,q}^* \in V_{i,q}^*$ and the essential boundary conditions of (5.53) on Γ_D are implemented with respect to $V_{i,q}^*$ via

$$b_D(e_{i,q}^{\mathrm{PU}}, v_{i,q}^*) = \langle g_d, v_{i,q}^* \rangle - b_D(u^{\mathrm{PU}}, v_{i,q}^*),$$

compare §4.2.2. The vanishing essential boundary conditions on $\partial(\omega_i \cap \Omega) \setminus \Gamma_N \setminus \Gamma_D$ are trivially satisfied in $W_{i,q}$ (5.55) due to the multiplication with φ_i which vanishes on $\partial\omega_i$.

Finally, we define our computable local error estimator (indicator) $\epsilon_{i,q}$ and the respective global estimate ϵ_q via

$$\epsilon_{i,q}^2 := a(e_{i,q}^{\mathrm{PU}}, e_{i,q}^{\mathrm{PU}}), \quad \epsilon_q := \Big(\sum_{i=1}^{N} \epsilon_{i,q}^2 \Big)^{1/2}. \tag{5.56}$$

Remark 5.6. If we assume that the employed PU functions φ_i satisfy $\varphi_i \in \mathcal{C}^2(\Omega)$, for instance by choosing a cubic B-spline $\mathcal{W} \in \mathcal{C}^2(\Omega)$ in (3.14), we can directly evaluate $-\Delta u^{\mathrm{PU}}$ on the right-hand side of (5.53).

Remark 5.7. Note that the presented subdomain estimator can also be employed with Nitsche's method [65, 128]. Here, however we need to compute an appropriate regularization parameter for the error estimation. Note that the regularization parameter employed in the discretization does not ensure the solvability of the local problems (5.53).

5.4.2 Refinement Indicator

To attain our refinement indicator function (5.49) from (5.56) we first define a Boolean indicator

$$b : C_\Omega \to \{\texttt{true}, \texttt{false}\}$$

by simple thresholding

$$b(\omega_i) := \begin{cases} \texttt{true} & \text{if } \epsilon_i \geq \sigma_b \epsilon_{\text{avg}}, \\ \texttt{false} & \text{else}, \end{cases}$$

with $\epsilon_{\text{avg}}^2 := N^{-1}\epsilon^2 = N^{-1}\sum_{i=1}^N \epsilon_i^2$ as it is done in many adaptive procedures. Note that we drop the subscript q of (5.56) in the following for the ease of notation. The second component of (5.49) is an assumed error reduction classifier

$$t : C_\Omega \to \{\texttt{constant}, \texttt{algebraic}, \texttt{exponential}\}$$

defined by

$$t(\omega_i) := \begin{cases} \texttt{constant} & \text{if } b(\omega_i) = \texttt{false}, \\ \texttt{algebraic} & \text{if } b(\omega_i) = \texttt{true} \text{ and } \epsilon_i \geq \epsilon_{i,*}, \\ \texttt{exponential} & \text{else}, \end{cases} \tag{5.57}$$

where $\epsilon_{i,*}$ is a predicted error based on the estimator (5.56) from the previous refinement step, i.e., we employ an extrapolation of a one-level history for the prediction $\epsilon_{i,*}$.

If a particular patch ω_i was h-refined, i.e. $r(\omega_i) = \texttt{h}$, we anticipate that the error with respect to the energy-norm on ω_i is reduced by a factor 2^{-p_i}. Furthermore, we assume that the error is distributed uniformly among the 2^d successor patches, so that we predict the error on a successor patch $\omega_l \subset \omega_i$ to be $\epsilon_{l,*} := \sigma_\texttt{h} 2^{-d} 2^{-p_i} \epsilon_i$ with $\sigma_\texttt{h} > 0$. In the case of a p-refined patch ω_i we anticipate an exponential convergence on ω_i and set $\epsilon_{i,*} := \sigma_\texttt{p}\epsilon_i$ with $0 < \sigma_\texttt{p} < 1$. For patches ω_i that are not refined we set $\epsilon_{i,*} := \sigma_\texttt{const}\epsilon_i$ with $\sigma_\texttt{const} > 0$ and we obtain the overall definition for a patch ω_i after refinement of C_Ω

$$\epsilon_{i,*} := \begin{cases} \sigma_\texttt{const}\epsilon_i & \text{if } \omega_i \in C_\Omega \text{ and } t(\omega_i) = \texttt{constant}, \\ \sigma_\texttt{p}\epsilon_i & \text{if } \omega_i \in C_\Omega \text{ and } t(\omega_i) = \texttt{exponential}, \\ \sigma_\texttt{h} 2^{-d} 2^{-p_j}\epsilon_j & \text{if } \omega_j \in C_\Omega \text{ with } t(\omega_j) = \texttt{algebraic} \text{ and } \omega_j \supset \omega_i. \end{cases} \tag{5.58}$$

Obviously, this prediction assumes that the respective refinement yields an optimal error reduction. Hence, if $\epsilon_i < \epsilon_{i,*}$ the refinement reduced the error more efficiently than predicted which indicates a higher regularity of the solution on the respective patch ω_i and that an `exponential` convergence may be achieved. If on the other hand $\epsilon_i \geq \epsilon_{i,*}$, we infer that the smoothness of the solution u is limited on ω_i and only an `algebraic` convergence is attainable. This consideration motivates the definition (5.57).

If there is no local refinement of the patch ω_i or the local approximation space V_i, i.e.

$$r((\omega_i, V_i)) = \texttt{null},$$

we assume an almost `constant` error on the patch ω_i which disregards effects due to the overlap of the patches. Note however that we can account for variation in the local error due to a change of the respective PU function φ_i stemming from the h-refinement of neighboring patches $\omega_j \in C_i$ by using a local parameter $\sigma_{\texttt{const},i}$ in the definition (5.58).

With these components in place, we define a completely local refinement indicator function

$$r((\omega_i, V_i)) := \begin{cases} \texttt{null} & \text{if } t(\omega_i) = \texttt{constant}, \\ \texttt{h} & \text{if } t(\omega_i) = \texttt{algebraic}, \\ \texttt{p} & \text{if } t(\omega_i) = \texttt{exponential}, \end{cases} \tag{5.59}$$

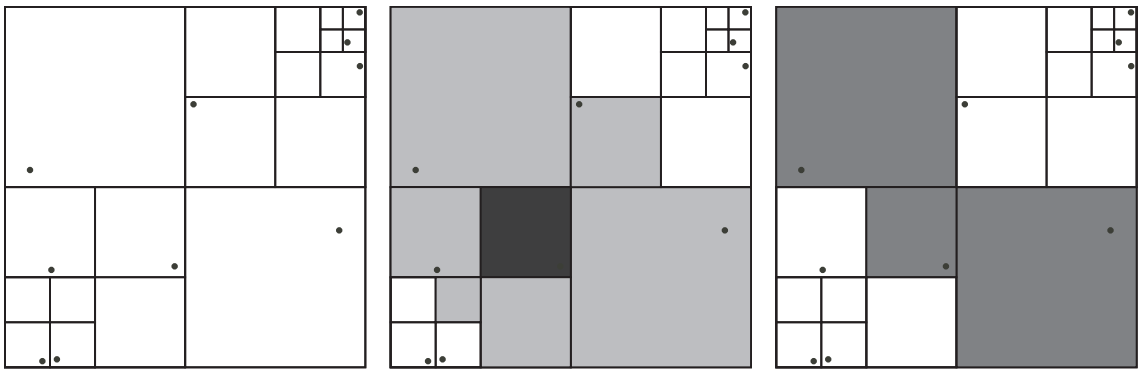


Figure 5.8. *Cell decomposition corresponding to an initial cover (left). Cell decomposition with cells of neighborhood C_i (light gray) and cell corresponding to the patch ω_i (dark gray) marked for refinement by (5.59) (center). Cell decomposition with cells (gray) marked for refinement by (5.63) (right).*

which may be used to steer the refinement of the PPUM space V^{PU} defined on the cover C_Ω.

The refinement of the p-components of V^{PU} is straightforward. In the case of $r((\omega_i, V_i)) = \mathtt{p}$ we simply increase the current polynomial degree p_i of the polynomial part $\mathcal{P}^{p_i}$ of V_i on ω_i; i.e.

$$V_i = \mathcal{P}^{p_i} \oplus (\mathcal{E}_i \setminus \mathcal{P}^{p_i}) \quad \longrightarrow \quad V_i = \mathcal{P}^{p_i+1} \oplus (\mathcal{E}_i \setminus \mathcal{P}^{p_i+1}) \tag{5.60}$$

(after the application of the respective local preconditioner $S_i^{\mathcal{E}\setminus\mathcal{P}}$ of §5.3).

If $r((\omega_i, V_i)) = \mathtt{h}$ we refine the respective h-component of our PPUM space V^{PU}; i.e., we refine the PU function φ_i. To this end, we refine the respective cover patch ω_i by subdividing its associated tree-cell to obtain the respective successor patches $\omega_j \subset \omega_i$ by (5.1). Since we employ the cell-centers (5.2) in the definition of the weight functions (3.14) for the Shepard construction (3.13) this procedure is already sufficient for our PPUM. However, we may also consider the refinement of the sampling set $\mathcal{X}_N$; i.e., the creation of new particles. Here, we can for instance use a center of mass approach [65] or we can define new particles in the (local) maxima of the respective local error approximation e_i^{PU}.

Note however, that (5.59) does not allow to bound the maximal level difference

$$L := \lim_{J\to\infty} \max_{k=0,\dots,J} L_k, \quad \text{with } L_k := \max_{\omega_{i,k}\in C_\Omega^k} \max_{\omega_{j,k}\in C_{i,k}} |l_{i,k} - l_{j,k}|. \tag{5.61}$$

of the resulting cover sequence C_Ω^k a priori, see Figure 5.8. Therefore, with a level-independent choice of α we cannot satisfy (4.5) a priori to ensure the flat top property (3.8) for all cover patches $\omega_{i,k} \in C_\Omega^k$ with $i = 1, \dots, N_k$ on all levels $k = 0, \dots, J$. Recall however that the flat top property of the PU functions is a sufficient condition for the linear independence of the employed shape functions only. It is not a necessary condition and in practice we observe no stability problems if (3.8) is violated for a few cover patches $\omega_{i,k}$.

More importantly, the constants $(C_{\nabla,k}, C_{\mathrm{FT},k}, M_k, C_{S,k}, C_{N,k})$ of Definitions 3.1 and 3.3 cannot be uniformly bounded independently of the level k. Recall that $C_{\nabla,k}$ and M_k are involved in the error estimates (3.5) and (3.6) of Theorem 3.1. Hence, let us consider the impact on the error bound (3.6)

$$\|\nabla(u - u_k^{\mathrm{PU}})\|_{L^2(\Omega)} \leq \sqrt{2M_k}\Big(\sum_{i=1}^{N_k} \big(\frac{C_{\nabla,k}}{\operatorname{diam}(\omega_{i,k})}\big)^2 \hat{e}_{i,k}^2 + C_{\infty,k}^2 \tilde{e}_{i,k}^2\Big)^{1/2},$$

where $\tilde{\epsilon}_{i,k}$ and $\hat{\epsilon}_{i,k}$ denote local error bounds in $V_{i,k}$, if L_k increases with finer levels k. Since $C_{\infty,k} \leq 1$ holds independently of the level difference L_k it is sufficient to focus on the term

$$E^2_{\nabla,k} := M_k \sum_{i=1}^{N_k} \left(\frac{C_{\nabla,k}}{\operatorname{diam}(\omega_{i,k})}\right)^2 \hat{\epsilon}^2_{i,k}$$

only. According to Lemma 4.1 and (4.6) the constant $C_{\nabla,k}$ is given by

$$C_{\nabla,k} = (2^{L_k} + 1)\, M_k\, C_T\, C_{\mathcal{W}} C^{-2}_{\mathcal{W},\partial}\, \frac{\alpha - 1}{2}$$

for a level-independent scaling parameter $\alpha \in (1,2)$. Hence, $C_{\nabla,k}$ grows exponentially with increasing L_k and therefore may completely compensate the convergence of the local error bounds $\hat{\epsilon}_{i,k}$. Then, the error term $E_{\nabla,k}$ is constant on all levels (or may even grow with increasing level k) and the convergence of our hp-adaptive PPUM may stall.

Thus, we need to generalize the local refinement indicator (5.59). To this end, we introduce a smoothness indicator

$$s(C_i) := \begin{cases} \texttt{true} & \text{if for all } \omega_j \in C_i \text{ there holds } l_j - l_i < L_{\max} \\ \texttt{false} & \text{else} \end{cases} \tag{5.62}$$

which determines if a patch $\omega_{i,k}$ is too coarse with respect to its neighbors $\omega_j \in C_{i,k}$ to bound the maximal level difference by $L_{\max} \geq 0$. With the help of (5.62) we define our smoothed refinement indicator, compare Figure 5.8, as

$$r_S((\omega_i, C_i, V_i)) := \begin{cases} \texttt{null} & \text{if } t(\omega_i) = \texttt{constant} \text{ and } s(C_i) = \texttt{true}, \\ \texttt{h} & \text{if } t(\omega_i) = \texttt{algebraic} \text{ and } s(C_i) = \texttt{true}, \\ \texttt{p} & \text{if } t(\omega_i) = \texttt{exponential} \text{ and } s(C_i) = \texttt{true}, \\ \texttt{h} & \text{if } t(\omega_i) = \texttt{constant} \text{ and } s(C_i) = \texttt{false}, \\ \texttt{hp} & \text{if } t(\omega_i) = \texttt{exponential} \text{ and } s(C_i) = \texttt{false}. \end{cases} \tag{5.63}$$

Since we now allow for simultaneous hp-refinement $r_S((\omega_i, C_i, V_i)) = \texttt{hp}$ of a patch ω_i and its associated approximation space V_i and enforce additional h-refinements with (5.63) we need to generalize the error prediction of (5.58) appropriately. To this end, we define the error prediction on a refined patch $\omega_i \in R_C(C_\Omega)$ for the smoothed refinement scheme (5.63) as

$$\epsilon_{i,*} := \begin{cases} \sigma_{\texttt{const}} \epsilon_i & \text{if } \omega_i \in C_\Omega \text{ and } r_S((\omega_i, C_i, V_i)) = \texttt{null}, \\ \sigma_{\texttt{p}} \epsilon_i & \text{if } \omega_i \in C_\Omega \text{ and } r_S((\omega_i, C_i, V_i)) = \texttt{p}, \\ \sigma_{\texttt{h}} 2^{-d} 2^{-p_j} \epsilon_j & \text{if } \omega_j \in C_\Omega \text{ with } r_S((\omega_j, C_j, V_j)) = \texttt{h}, \\ \sigma_{\texttt{p}} \sigma_{\texttt{h}} 2^{-d} 2^{-p_j} \epsilon_j & \text{if } \omega_j \in C_\Omega \text{ with } r_S((\omega_j, C_j, V_j)) = \texttt{hp} \text{ and } \omega_j \supset \omega_i. \end{cases} \tag{5.64}$$

Let us now consider the incorporation of the refined PPUM space in our sequence V_k^{PU} with $k = 0, \ldots, J$. For the ease of notation we denote the cover resulting from the above refinement as $R_C(C_\Omega^J)$ and the refined PPUM space as $R_V(V_J^{\mathrm{PU}})$ in the following. In a naive approach we simply extend our sequences C_Ω^k and V_k^{PU} by $R_C(C_\Omega^J)$ and $R_V(V_J^{\mathrm{PU}})$ respectively; we define

$$C_\Omega^{J+1} := R_C(C_\Omega^J), \quad V_{J+1}^{\mathrm{PU}} := R_V(V_J^{\mathrm{PU}}).$$

This procedure however yields a new level $J+1$ for every refinement step which might spoil the overall complexity of our multilevel PPUM and can lead to a dramatic increase in the storage requirements; i.e., the attained sequence of covers C_Ω^k with $k = 0, \ldots, J+1$ violates the minimal property of Lemma 5.2.

To ensure that our sequence of covers C_Ω^k satisfies Lemma 5.2 after refinement we use the following replacement strategy

$$(C_\Omega^{J+1}, V_{J+1}^{\mathrm{PU}}) := \begin{cases} (R_C(C_\Omega^J), R_V(V_J^{\mathrm{PU}})) & \text{if } \omega_{i,J} \in C_\Omega^J \text{ with } l_{i,J} = J \\ & \text{and } r((\omega_{i,J}, V_{i,J})) = \mathtt{h}, \\ (\emptyset, \operatorname{span}\langle 0 \rangle) & \text{else,} \end{cases} \tag{5.65}$$

and

$$(C_\Omega^J, V_J^{\mathrm{PU}}) := \begin{cases} (C_\Omega^J, V_J^{\mathrm{PU}}) & \text{if } \omega_{i,J} \in C_\Omega^J \text{ with } l_{i,J} = J \\ & \text{and } r((\omega_{i,J}, V_{i,J})) = \mathtt{h}, \\ (R_C(C_\Omega^J), R_V(V_J^{\mathrm{PU}})) & \text{else,} \end{cases} \tag{5.66}$$

Here, we generate a new level $J+1$ of the sequence C_Ω^k and V_k^{PU} only if a patch $\omega_{i,J} \in C_\Omega^J$ with $l_{i,J} = J$, i.e., with minimal diameter, is h-refined. If no such patch $\omega_{i,J}$ exists we replace the currently finest level J with its refined version. Obviously, the attained covers satisfy Lemma 5.2 so that C_Ω^k and V_k^{PU} employ a minimal number of levels.

Let us summarize the properties of our hp-adaptive refinement scheme using the refinement indicator (5.63) in the following lemma.

Lemma 5.3. *Assume that the sequence of covers C_Ω^k generated by Algorithm 5.1 satisfies $L_k \leq L_{\max}$ for some $L_{\max} \geq 0$ on all levels $k = 0, \dots, J$. Then the adaptive refinement $R_C(C_\Omega^J)$ of C_Ω^J due to (5.63) satisfies $L_{R_C} \leq L_{\max}$ where L_{R_C} denotes the maximal level difference (5.61) of the refined cover $R_C(C_\Omega^J)$.*

The constants of Definition 3.1 for the Shepard functions (3.13) with weight functions satisfying (3.14) defined on C_Ω^k for all $k = 0, \dots, J$ and $R_C(C_\Omega^J)$ are uniformly bounded. Assuming that $\alpha < 1 + 2^{-L_{\max}-1}$ the covers C_Ω^k for all $k = 0, \dots, J$ and $R_C(C_\Omega^J)$ are admissible and the respective Shepard functions (3.13) have the flat top property according to Definition 3.2. The covers C_Ω^k for all $k = 0, \dots, J$ and $R_C(C_\Omega^J)$ employ a minimal number of levels.

Proof. It is sufficient to show that the maximal level difference L_{R_C} of $R_C(C_\Omega^J)$ satisfies $L_{R_C} \leq L_{\max}$. The remaining propositions then follow from Lemma 3.1 and Lemma 4.1, (5.65) and (5.66).

Consider $\omega_{i,J} \in C_\Omega^J$ and $\omega_{j,J} \in C_{i,J}$ with $l_{i,J} - l_{j,J} = L_{\max}$ such that $s(C_i) = \mathtt{false}$. Therefore, $r_S((\omega_{i,J}, C_{i,J}, V_{i,J})) \in \{\mathtt{h}, \mathtt{hp}\}$ and the respective level difference L_{R_C} will decrease if there holds $r_S((\omega_{j,J}, C_{j,J}, V_{j,J})) = \mathtt{null}\}$ or it stays constant at $L_{R_C} = L_{\max}$ if $r_S((\omega_{j,J}, C_{j,J}, V_{j,J})) \in \{\mathtt{h}, \mathtt{hp}\}$.

If there holds $l_{i,J} - l_{j,J} < L_{\max}$ for a particular patch $\omega_{i,J} \in C_\Omega^J$ and all $\omega_{j,J} \in C_{i,J}$ the refinement step can at most increase the maximal L_{R_C} level difference to $L_{\max}$. □

Finally, we employ the hierarchical enrichment classifier e^H of Algorithm 5.3 to transfer the enrichment information of V_J^{PU} to its refined version $R_V(V_J^{\mathrm{PU}})$. Note that due to our multiplicative enrichment approach (5.22) not only the resolution of the smooth polynomial part $\mathcal{P}^{p_{i,J}}$ of a p- or hp-refined patch $\omega_{i,J}$ is increased but also the resolution of the enrichment space $\mathcal{E}_{i,J}$. In the case of an h-refined patch $\omega_{i,J}$ Algorithm 5.3 enriches all its successor patches such that the resolution of the global enrichment space is again improved. Hence, the combination of the multilevel approach of §5, Algorithm 5.3 and the hp-refinement scheme presented above yields a multilevel ehp-adaptive PPUM.

The optimal operation count of the multilevel iteration $M_\gamma^{\nu_1,\nu_2}$ for our ehp-adaptive PPUM has to be ensured employing similar strategies as in multigrid for adaptive grids, see [158] and the references cited therein for details. In general the global series (4.25) may not converge for an adaptively refined sequence of PPUM spaces. This would lead (at least) to a logarithmic complexity of the global multilevel iteration $M_\gamma^{\nu_1,\nu_2}$. However, we can still achieve an optimal complexity implementation. The basic idea is to restrict the iteration on each level to a subset of patches in

such a way that the respective series converges yet without deterioration in the quality of the iteration; i.e., we define a partitioning of the degrees of freedom into an active subset and an inactive subset. To this end we introduce an Boolean activity classifier $a(C_{i,J}, V_{i,J})$ which indicates if the degrees of freedom associated with the local approximation space $V_{i,J}$ or the partition of unity function $\varphi_{i,J}$ of the respective cover patch $\omega_{i,J}$ were affected by the latest refinement step, see Algorithm 5.5.

Remark 5.8. Note that we can utilize the activity classifier information in the discretization to reduce the computational costs. We can re-use the entries of the load vector which are associated with the degrees of freedom of $\omega_{i,J}$ with $a(C_{i,J}, V_{i,J}) = \texttt{false}$. Similarly, the entries of the stiffness matrix corresponding to a pair of patches $\omega_{i,J}$ and $\omega_{j,J} \in C_{i,J}$ such that $a(C_{i,J}, V_{i,J}) = a(C_{j,J}, V_{j,J}) = \texttt{false}$ are not affected by the refinement and can be re-used from the previous discretization. Thus, if all neighbors $\omega_{j,J} \in C_{i,J}$ of a particular patch $\omega_{i,J}$ are inactive the complete block-row associated with $\omega_{i,J}$ can be re-used.

Remark 5.9. Similarly we can also reduce the computational cost associated with the error estimation with the help of the activity classifier. Observe that if all neighbors of a particular patch $\omega_{i,J}$ are inactive, i.e. there holds $a(C_{j,J}, V_{j,J}) = \texttt{false}$ for all $\omega_{j,J} \in C_{i,J}$, then the right-hand side of (5.53) is not affected by the latest refinement. Thus, the respective error estimate (5.56) is constant and does not have to be re-computed.

Summary

We close this chapter with a summary of the resulting multilevel ehp-adaptive PPUM approximation in algorithmic form. Here, we include the construction of the respective stability preconditioner of §5.3.2 and the direct splitting of §4.2.2 for completeness.

Algorithm 5.5 (Multilevel hp-Adaptive PPUM with Hierarchical Enrichment)**.**

Compute covers C_Ω^k via Algorithm 5.1 with $N_k := \operatorname{card}(C_\Omega^k)$, define respective Shepard functions (3.13) with weight (3.14) and global PPUM spaces V_k^{PU} for $k = 0, \dots, J$ with maximal level difference $L_{\max}$. Choose $q > 0$ and $\epsilon_{\mathrm{tol}} \geq 0$.

For all levels $k = 0, \dots, J$:

1. If $k = 0$, define $u_k^{\mathrm{PU}} := 0$, $\epsilon_{i,*}^J := -1$, and $a(C_{i,J}, V_{i,J}) = \texttt{true}$ for all all $i = 1, \dots, N_J$.
2. Apply hierarchical enrichment classifier e^H of Algorithm 5.3.
3. Discretize boundary value problem via steps 5–13 of Algorithm 4.3 using V_k^{PU} as trial and test space and the preconditioner $S_{\mathcal{E}\setminus\mathcal{P}}$ of Algorithm 5.4. Re-use entries of stiffness matrix, load vector, and transfer operators based on activity classifier, see Remark 5.8. In step 12 of Algorithm 4.3 employ the iterative solver of Algorithm 4.5 with initial guess u_{k-1}^{PU} (use local-to-local transfer operator if necessary) and Algorithm 4.4 as interior iteration using the local-to-local transfer operators of §5.2.1 and the smoothers of §5.2.2 to compute the approximate solution u_k^{PU}. Iterate over all patches $\omega_{i,J} \in C_\Omega^J$ with $a(C_{i,J}, V_{i,J}) = \texttt{true}$ only.
4. If $k = J$:
   (a) For all patches $\omega_{i,J} \in C_\Omega^J$ with $a(C_{j,J}, V_{j,J}) = \texttt{true}$ for all neighbors $\omega_{j,J} \in C_{i,J}$, compute local error estimates $\epsilon_{i,q}^J$ of (5.56) for all patches $\omega_{i,k} \in C_\Omega^J$. For all patches $\omega_{i,J}$ with $a(C_{i,J}, V_{i,J}) = \texttt{false}$, set $\epsilon_{i,q}^J = \epsilon_{n,q}^{J-1}$ where $\omega_{i,J} = \omega_{n,J-1}$.
   (b) Compute global error estimate ϵ_q^J of (5.56). If $\epsilon_q^J \leq \epsilon_{\mathrm{tol}}$, stop computation.

(c) Define refinement indicator r_S of (5.63) based on the local estimates $\epsilon^J_{i,q}$ and the respective predictions $\epsilon^J_{i,*}$.

(d) Define refined cover $R_C(C^J_\Omega)$ and respective PPUM space $R_V(V^{\mathrm{PU}}_J)$. Initialize $a(C^R_i, V^R_i) = \texttt{false}$ for all neighborhoods $C^R_i \subset R_C(C^J_\Omega)$ and local approximation spaces V^R_i. For the refined patches $\omega^R_i \in R_C(C^J_\Omega)$ define the activity classifier $a(C^R_i, V^R_i) = \texttt{true}$ and the respective error prediction $\epsilon^R_{i,*}$ according to (5.64).

(e) Extend the sequences of covers C^l_Ω and PPUM spaces V^{PU}_l for $l = 0, \dots, J$ according to (5.65) and (5.66).

(f) If $(C^{J+1}_\Omega, V^{\mathrm{PU}}_{J+1}) \neq (\emptyset, \mathrm{span}\langle 0 \rangle)$, set $J = J + 1$.

(g) Else:

i. Compute local-to-local approximation of u^{PU}_J in refined PPUM space $R_V(V^{\mathrm{PU}}_J)$ and set initial value $u^{\mathrm{PU}}_{J-1} = u^{\mathrm{PU}}_J$. (Observe that $u^{\mathrm{PU}}_{J-1} \notin V^{\mathrm{PU}}_{J-1}$ and the local-to-local transfer must not be applied in step 3.)

ii. Delete all computed data on level J that cannot be re-used, compare Remark 5.8.

In the next chapter we focus on the implementation of the above algorithm and all of its respective components. Here, the most important aspect is numerical integration since our PPUM shape functions are non-polynomial. The PU functions (3.13) with spline weights (3.14) are in general piecewise rational functions and the enrichment functions $\eta_{i,k} \in \mathcal{E}_{i,k} \subset V_{i,k}$ are by definition discontinuous or even singular functions. Thus, the numerical integration of our PPUM shape functions is in general more challenging than the numerical integration of e.g. classical FEM shape functions.

Part III

Implementation and Validation

Chapter 6
Implementation

In the previous sections we presented the specific ingredients and analytic properties of meshfree methods and the PPUM in particular. In the following we are concerned with the efficient implementation of the PPUM. Yet, many of the presented concepts and techniques can be applied to other meshfree or even mesh-based methods.

Even with the rapid growth of processing speed it is necessary to employ parallel computers for large scale simulations to reduce the computing time to an acceptable range. Another issue to keep in mind is that the amount of local memory of a computer is limited. Thus, very large problems demand the use of parallel computers with distributed memory. Here, the data necessary for the local computations in a single processing core must be explicitly exchanged between participating processors by the programmer which renders the implementation more challenging. In [63, 125, 126] we presented a detailed discussion of the parallelization of the PPUM and therefore give a compact summary of the key concepts and most important aspects only.

But before we come to the parallelization of our PPUM let us first consider the single most challenging task in the implementation of a meshfree Galerkin method, the construction of an appropriate integration scheme [15, 30, 31, 39, 60, 61, 124] for the efficient and reliable assembly of the stiffness matrix. Since the meshfree shape functions are in general *not* piecewise polynomial the integration in meshfree methods must be more involved than in the FEM. Another issue which complicates the assembly of the stiffness matrix is the fact that there is no fixed connectivity among the degrees of freedom and the supports of the meshfree shape function overlap in a rather arbitrary fashion. Finally, it is time to be concerned with the geometry of the computational domain Ω and its boundary $\partial\Omega$. Recall that the construction of the shape function does not employ any information about Ω or its boundary. In the FEM the shape functions align with the discrete approximation to the boundary of the domain. So far we do not even have an approximation to the domain or the boundary and the supports of our shape functions ω_i are allowed to overlap $\Omega \cup \partial\Omega$. In the assembly of the stiffness matrix however we must carry-out the integration of the weak form of the operator on the domain Ω and its boundary $\partial\Omega$ so that we need to restrict the evaluation of our shape functions to the intersections $\omega_i \cap \Omega$ and $\omega_i \cap \partial\Omega$. Due to the complex algebraic structure of our meshfree shape functions and the possibly complicated geometry of $\omega_i \cap \Omega$ these integrals cannot be computed analytically in general and we must resort to numerical integration.

6.1 Numerical Integration and Geometry Approximation

In the FEM the (numerical) integration of the weak form of the considered PDE is simpler than in meshfree methods due to the availability of a mesh. This fact often leads to the perception that a mesh is required for numerical integration also in meshfree methods and that therefore meshfree Galerkin methods are not truly meshfree. However, a mesh is not required for the reliable and stable numerical integration of the weak form — neither in the FEM nor in meshfree methods. We only need an appropriate decomposition of the integration domain into cells with pairwise disjoint interiors (a far more general partitioning of the domain than a mesh). It is just due to the specific construction of FE shape functions that the resulting decomposition in the FEM is in fact the employed mesh itself.

To see this let us quickly review how (numerical) integration of the weak form actually works for a Poisson problem in the FEM. In general the entries $A_{i,j}$ of the stiffness matrix $A = (A_{i,j})$ are given by

$$A_{i,j} := a(\phi_j, \phi_i) = \int_\Omega \nabla\phi_j \nabla\phi_i \, dx = \int_{\Omega \cap \operatorname{supp}(\phi_j) \cap \operatorname{supp}(\phi_i)} \nabla\phi_j \nabla\phi_i \, dx$$

where ϕ_i and ϕ_j denote FEM shape functions. Let T_α denote the cells of the considered mesh $\mathcal{T}$, i.e. there holds

$$\Omega = \bigcup_\alpha T_\alpha, \qquad T_\alpha^\circ \cap T_\beta^\circ = \emptyset \quad \text{for all } \alpha \neq \beta$$

and

$$\operatorname{supp}(\phi_j) = \bigcup_{\alpha(j)} T_{\alpha(j)}, \qquad \operatorname{supp}(\phi_i) = \bigcup_{\alpha(i)} T_{\alpha(i)}.$$

Therefore, the supports of the shape functions are perfectly aligned to each other and to the (respective approximation of the) domain. Hence, the computation of the integration domains $\Omega \cap \operatorname{supp}(\phi_j) \cap \operatorname{supp}(\phi_i)$ is trivial in the FEM and there holds

$$\Omega \cap \operatorname{supp}(\phi_j) \cap \operatorname{supp}(\phi_i) = \bigcup_{\alpha(i,j)} T_{\alpha(i,j)}$$

so that

$$A_{i,j} = \int_{\Omega \cap \operatorname{supp}(\phi_j) \cap \operatorname{supp}(\phi_i)} \nabla\phi_j \nabla\phi_i \, dx = \sum_{\alpha(i,j)} \int_{T_{\alpha(i,j)}} \nabla\phi_j \nabla\phi_i \, dx.$$

Observe that the integrand $\nabla\phi_j \nabla\phi_i$ is smooth on each cell since $\phi_i|_{T_\alpha} \in \mathcal{P}^p$ for all $T_\alpha \in \mathcal{T}$ and some $p > 0$. Thus, the integrals can be well approximated on $T_{\alpha(i,j)}$. Since the cells $T_{\alpha(i,j)}$ come from a valid globally consistent mesh $\mathcal{T}$ and $\Omega \cap \operatorname{supp}(\phi_j) \cap \operatorname{supp}(\phi_i)$ is connected we do in fact integrate the entries of the stiffness matrix over a local mesh. Yet, the above line of argument does not rely on the fact that the $T_{\alpha(i,j)}$ provide a mesh topology. We make use of the following two properties only:

1. Each integration domain $\Omega \cap \operatorname{supp}(\phi_j) \cap \operatorname{supp}(\phi_i)$ is given by the union of unique cells $T_{\alpha(i,j)}$ with pairwise disjoint interiors.
2. The integrands of the weak form restricted to the interiors of these cells are smooth.

Observe for instance that we can in principle allow for hanging nodes of arbitrary order in the union of the cells $T_{\alpha(i,j)}$ a property that is usually not acceptable for FE meshes. It is only due to the fact that we started with a valid mesh $\mathcal{T}$ and the specific structure of classical finite element shape functions that the integration domains are decomposed by a local mesh.

Thus, there is absolutely no need for a so-called background mesh in meshfree Galerkin methods. In fact the use of a background mesh which violates the above properties will spoil the accuracy and the stability of the resulting numerical scheme. Hence, the integration cells cannot be chosen a priori in meshfree methods. We must construct the cells $T_{\alpha(i,j)}$ explicitly within our implementation with respect to the algebraic structure of the employed shape functions and the geometry of the domain Ω. Yet, this construction is a much simpler task than full-blown mesh-generation.

Recall that the global regularity of the our PPUM shape functions $\varphi_i \vartheta_i^m \in V^{\mathrm{PU}}$ is dominated by the regularity of the PU functions φ_i of (3.13), i.e.

$$\varphi_i(x) = \frac{W_i(x)}{\sum_{\omega_k \in C_i} W_k(x)} = \frac{W_i(x)}{\sum_{k=1}^{N} W_k(x)}.$$

Thus, let us first focus on the PU functions φ_i and how we can construct a decomposition $\{ic_{i,\alpha} \subset \omega_i\}$ of its support ω_i such that $\varphi_j|_{ic_{i,\alpha}}$ is smooth for all $\omega_j \in C_i \subset C_\Omega$. To this end, we carry out the differentiation in

$$\int_\Omega \nabla(\varphi_i \vartheta_i^n) \nabla(\varphi_j \vartheta_j^m)\, dx = \int_{\Omega \cap \omega_i \cap \omega_j} \nabla(\varphi_i \vartheta_i^n) \nabla(\varphi_j \vartheta_j^m)\, dx.$$

With the notation

$$S := \sum_{k=1}^{N} W_k, \quad T := \sum_{k=1}^{N} \nabla W_k, \quad \text{and} \quad G_i := \nabla W_i S - W_i T$$

we end up with the integrals

$$\begin{aligned} a(\varphi_j \vartheta_j^m, \varphi_i \vartheta_i^n) \quad = \quad & \int_{\Omega \cap \omega_i \cap \omega_j} S^{-4} G_i \vartheta_i^n G_j \vartheta_j^m \, dx + \int_{\Omega \cap \omega_i \cap \omega_j} S^{-2} W_i \nabla \vartheta_i^n W_j \nabla \vartheta_j^m \, dx + \\ & \int_{\Omega \cap \omega_i \cap \omega_j} S^{-3} G_i \vartheta_i^n W_j \nabla \vartheta_j^m \, dx + \int_{\Omega \cap \omega_i \cap \omega_j} S^{-3} W_i \nabla \vartheta_i^n G_j \vartheta_j^m \, dx \end{aligned} \tag{6.1}$$

for the stiffness matrix and the integrals

$$\langle f, \varphi_i \vartheta_i^n \rangle_{L^2} = \int_{\Omega \cap \omega_i} S^{-1} W_i \vartheta_i^n f \, dx \tag{6.2}$$

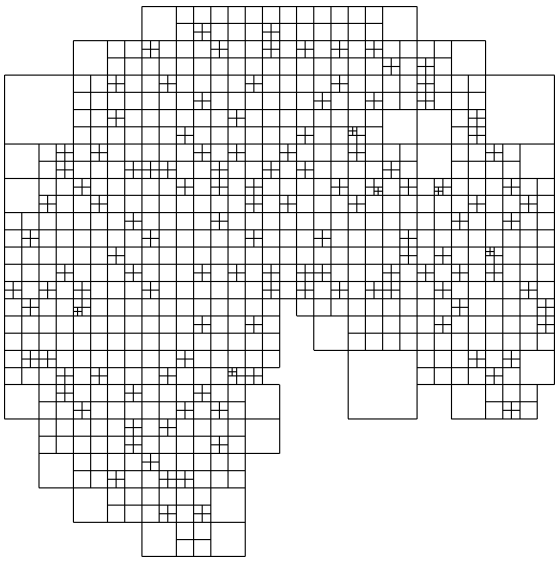
for the right-hand side. Thus, we need to assess the regularity of the functions S, T, W_j and G_j for all $\omega_j \in C_i$ to construct a decomposition $\{ic_{i,\alpha}\}$ of the patch ω_i such that their restrictions to the interiors of the cells $ic_{i,\alpha}$ are smooth functions. In essence this means that the weight functions W_j must be smooth functions on the cells $ic_{i,\alpha}$.

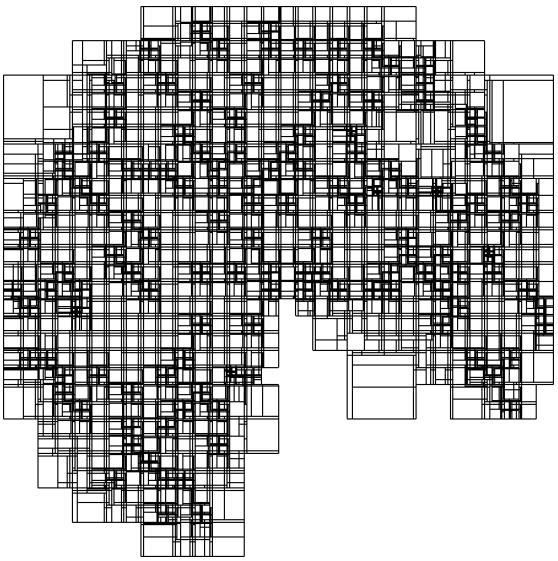
In a first step we initialize our decomposition of the patch ω_i by the tree-cells $\mathcal{C}_j$ that are intersected by ω_i

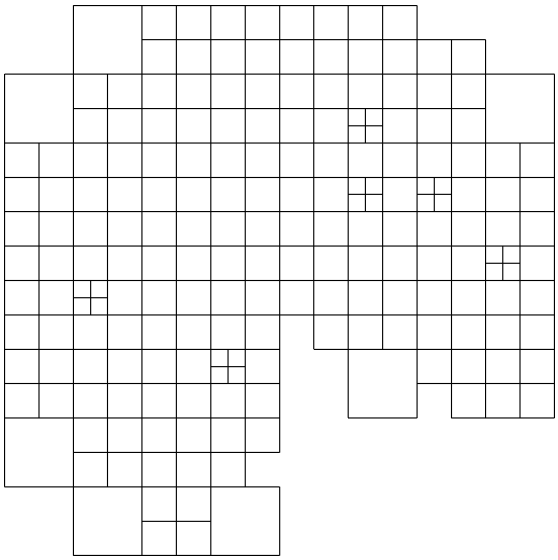
$$\widetilde{ic}_{i,j} := \mathcal{C}_j \quad \text{for all } \alpha \mathcal{C}_j = \omega_j \in C_i \subset C_\Omega.$$

The union of these cells for all $i = 1, \ldots, N$ corresponds to a very crude initial approximation to the domain Ω, see Figures 6.1 and 6.7.

Since the supports of the PU functions φ_j are not aligned with these cells due to the employed scaling, i.e. due to the overlap of the supports, we need to refine the decomposition $\{\widetilde{ic}_{i,j}\}$ by considering the intersections of the cells $\widetilde{ic}_{i,j}$ with the supports ω_k for all $\omega_k \in C_i \cap C_j$, see







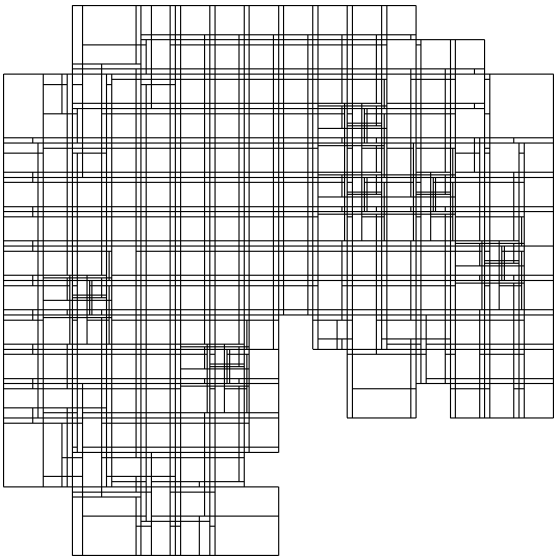


Figure 6.1. *Initial decomposition based on the tree-cells $\mathcal{C}_j$ only (top: level 7, bottom: level 5). The respective tree decomposition was generated by sampling with Halton points.*

Figure 6.2. *Refined decomposition respecting the supports ω_k (top: level 7, bottom: level 5).*

Figures 6.2 and 6.7. Observe that we decompose the cells $\widetilde{ic}_{i,j}$ into a set of sub-cells $\widetilde{ic}_{i,j,l}$ such that every intersection

$$\widetilde{ic}_{i,j} \cap \omega_k = \bigcup_{l \in \Lambda(k)} \widetilde{ic}_{i,j,l}$$

is the union of these sub-cells. Such a decomposition resolves the piecewise constant covering index λ_{C_Ω} of (3.3) and contains so-called hanging nodes.

However, the PU functions φ_j are in general not smooth on the cells $\widetilde{ic}_{i,j,l}$ since the weight

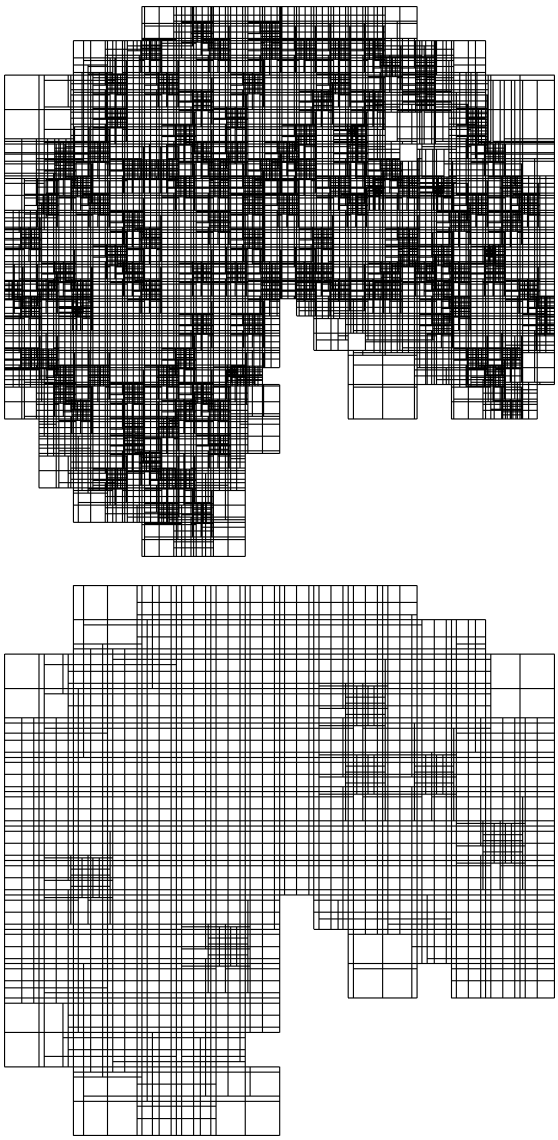


Figure 6.3. *Refined decomposition using all spline decompositions* $\{sc_{i,n}\}$ *(top: level* 7*, bottom: level* 5*).*

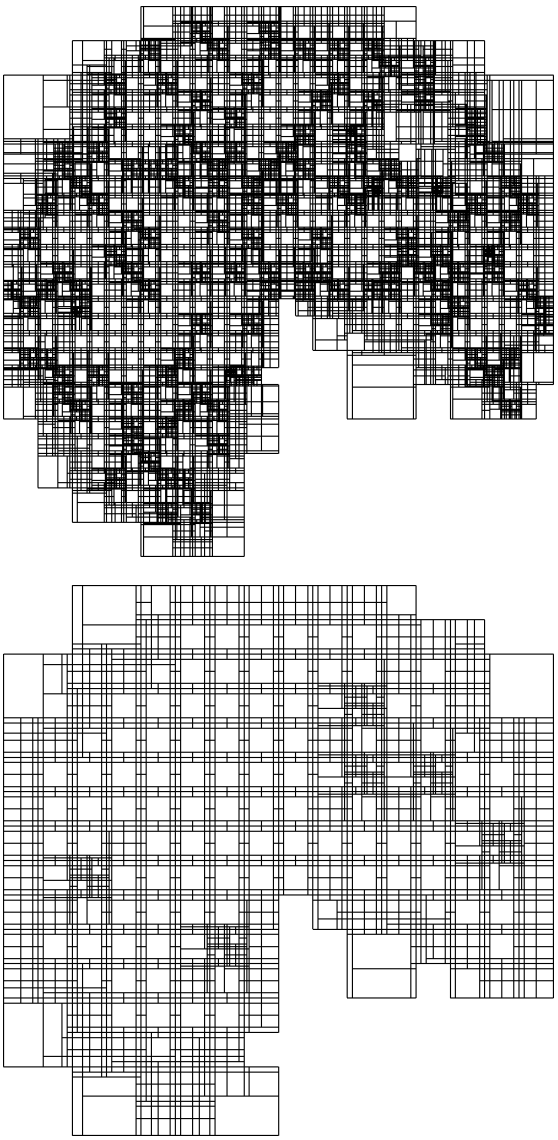


Figure 6.4. *Refined decomposition using the necessary spline decompositions* $\{sc_{i,n}\}$ *only (top: level* 7*, bottom: level* 5*).*

functions W_j are usually products of splines and therefore defined in a piecewise fashion. Hence, we need to refine the cells $\widetilde{ic}_{i,j,l}$ further to resolve the regularity of the weight functions. To this end, we consider the decomposition $\{sc_{k,n}\}$ of ω_k such that the tensor product spline W_k is smooth on $sc_{k,n}$. In a naive approach we decompose all cells $\widetilde{ic}_{i,j,l}$ by the decompositions $\{sc_{k,n}\}$ to obtain the cells $ic_{i,j,l,s}$, see Figure 6.3. Yet, this yields a large number of unnecessary integration cells. Recall that $\phi_i \equiv 1$ holds on all cells $\widetilde{ic}_{i,j,l}$ with $\lambda_{C_\Omega}|_{\widetilde{ic}_{i,j,l}} = 1$ independently of the employed weight functions. Hence, the jumps of S, T, W_i, and G_i cancel out on this cell and must *not* be considered in its decomposition, see Figures 6.4 and 6.7. Obviously, the decompositions depicted in Figure 6.4

employ a much smaller number of cells than those of Figure 6.3. Moreover, for each $i = 1, \dots, N$ there is a (rather large) cell $ic_{i,j,\alpha,W}$ on which φ_i is constant. For the ease of notation we denote the cells $ic_{i,j,l,s}$ in the following as $ic_{i,\alpha}$.

The constructed decompositions $\{ic_{k,\alpha}\}$ with $k = 1, \dots, N$ satisfy

$$\omega_i \cap \omega_j = \bigcup ic_{i,\alpha(j)} = \bigcup ic_{j,\alpha(i)}, \quad \text{with} \quad ic_{j,\alpha(i)} \cap ic_{i,\alpha(j)} \neq \emptyset \Rightarrow ic_{j,\alpha(i)} = ic_{i,\alpha(j)} \tag{6.3}$$

and

$$\varphi_j|_{ic_{i,\alpha(j)}} \text{ is smooth for all } \omega_j \in C_i.$$

Thus the algebraic structure of our PU functions is resolved by the constructed decomposition. Let us now consider the local approximation spaces V_j. If there is no enrichment there holds $V_j = \mathcal{P}^{p_j}$ so that all local approximation functions ψ_j^s are smooth on the cells $ic_{i,\alpha}$. Therefore, the restrictions of our PPUM shape functions $\varphi_j \psi_j^s$ to the cells are smooth functions and no further refinement is required. In the presence of enrichment functions, i.e. $V_j = \mathcal{P}^{p_j} + \mathcal{E}_j$, however we need to refine the above decomposition appropriately.[1]

To this end, let us assume that each enrichment function η^t is equipped with a decomposition $\{ec_{k,n}\}$ of its support such that the integrals

$$\int_{\text{supp}(\eta^t)} (\eta^t)^2 \, dx, \qquad \text{and} \qquad \int_{\text{supp}(\eta^t)} (\nabla \eta^t)^2 \, dx$$

can be approximated well by some numerical integration rule defined on the cells $ec_{k,n}$. For analytical enrichment functions we can construct such a decomposition for instance by an adaptive sub-division integration scheme a priori. If the enrichment functions η^t are pre-computed numerically then the respective decomposition is automatically provided by the respective decompositions of the basis functions employed in their approximation. We obtain our PPUM decomposition from the intersection of the cells $ic_{i,\alpha}$ with the cells of the decomposition $ec_{i,n}$ of the local enrichment functions η_j^t for all $\omega_j \in C_i$, see Figures 6.5 and 6.7.

Observe that the presented construction utilizes information from V^{PU} only and the number of integration cells depends on the local neighborhoods C_i only. Thus, we can use the construction on each level $k = 1, \dots, J$ of our multilevel sequence V_k^{PU} and maintain the optimal complexity of our multilevel PPUM.

Finally, we need to consider the domain Ω and its boundary $\partial\Omega$. To this end, let us first consider the case when the computation domain Ω is given as a volumetric mesh. Obviously, such an assumption would destroy the meshfree character of our PPUM however it provides a fair amount of insight into the challenges arising from the consideration of general domains Ω in our multilevel context. With $\Omega = \bigcup T$ we trivially obtain an appropriate decomposition of the PPUM integration domains $\Omega \cap \omega_i \cap \omega_j$ by the intersections of the cells $ic_{i,\alpha}$ with the domain cells T. Observe though that this yields a number of integration cells that cannot be bounded by the the number of degrees of freedom of the respective PPUM space V_k^{PU} on level k. Hence, we loose the optimal complexity of our multilevel PPUM by this approach. The numerical integration is (almost) equally expensive on all levels $k = 1, \dots, J$. In a first step to overcome this limitation we can restrict the computation of these intersections only to those patches ω_i with intersect the domain boundary $\partial\Omega$. This somewhat reduces the number of integration cells yet the overall optimal complexity may still be lost.

Hence, we must approximate the domain Ω on each level k independently so that the number of integration cells can be bounded by the number of degrees of freedom on level k to maintain

[1] Note that an appropriate refinement of the mesh for the integration of the non-smooth enrichment functions must also be employed in the GFEM and XFEM.

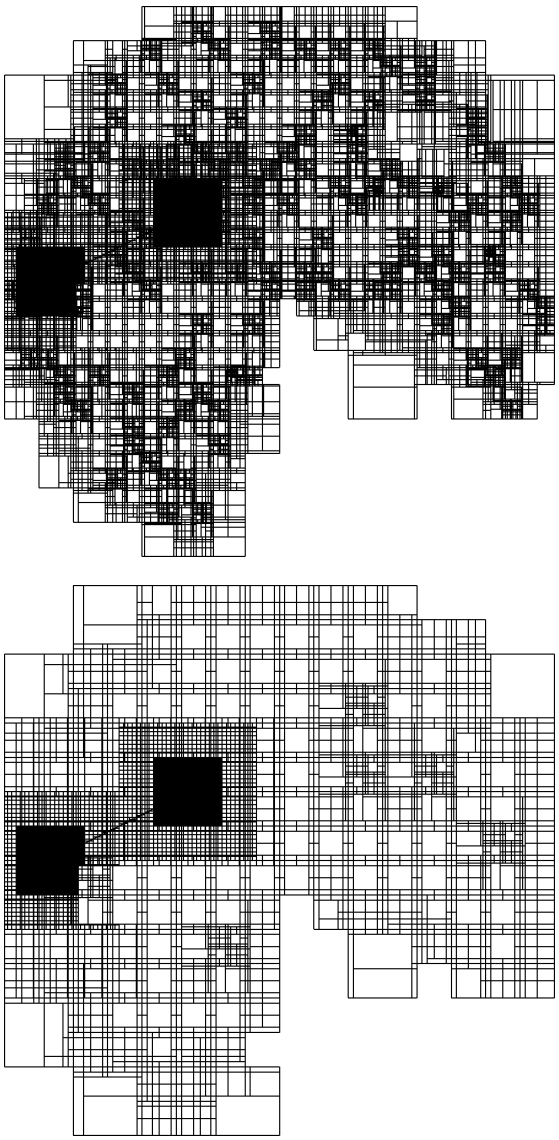


Figure 6.5. *Refined decomposition using the enrichment decompositions* $\{ec_{i,n}\}$ *(top: level* 7*, bottom: level* 5*).*

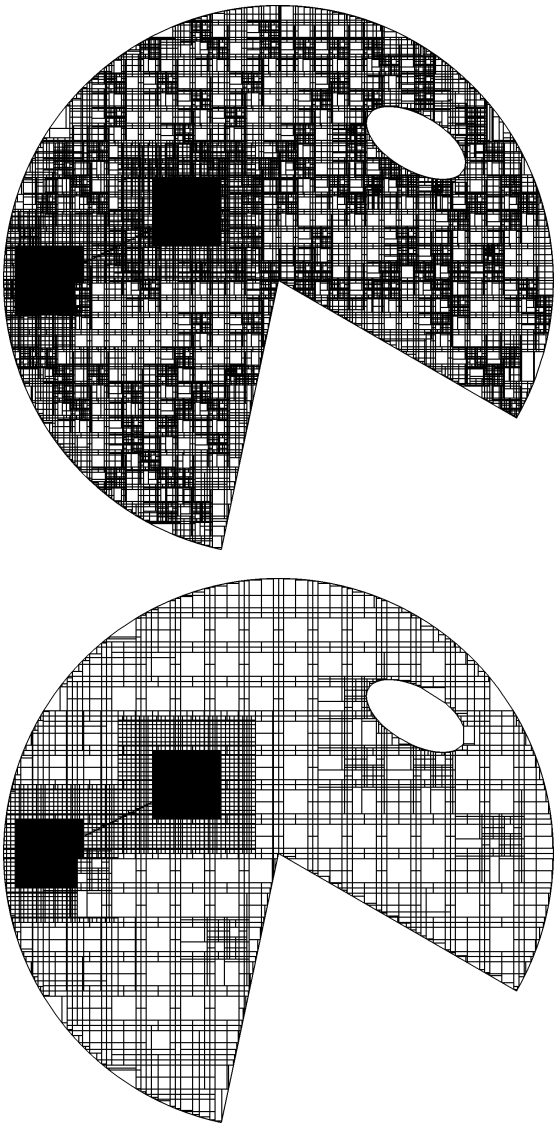


Figure 6.6. *Refined decomposition with parametric cells at the boundary (top: level* 7*, bottom: level* 5*).*

the optimal complexity of our multilevel PPUM. To this end we make the assumption that we can compute the intersections of the boundary $\partial ic_{i,\alpha}$ of an integration cell $ic_{i,\alpha}$ (an axis-aligned D-rectangle) with the boundary $\partial\Omega$ of the domain Ω. Note that this operation is available for far more general input data than a volumetric mesh and hence the meshfree character of our PPUM is not compromised by this assumption.

With the help of the intersections $\partial ic_{i,\alpha} \cap \partial\Omega$ we refine the integration cell $ic_{i,\alpha}$ such that the boundaries of the refined cells intersect with the boundary $\partial\Omega$ of the domain only in corners of the refined cells. Observe that this assumption does not imply that the complete boundary $\partial\Omega$ is resolved (unlike in the above situation). The boundary segment $\partial\Omega \cap ic_{i,\alpha}$ covered by an

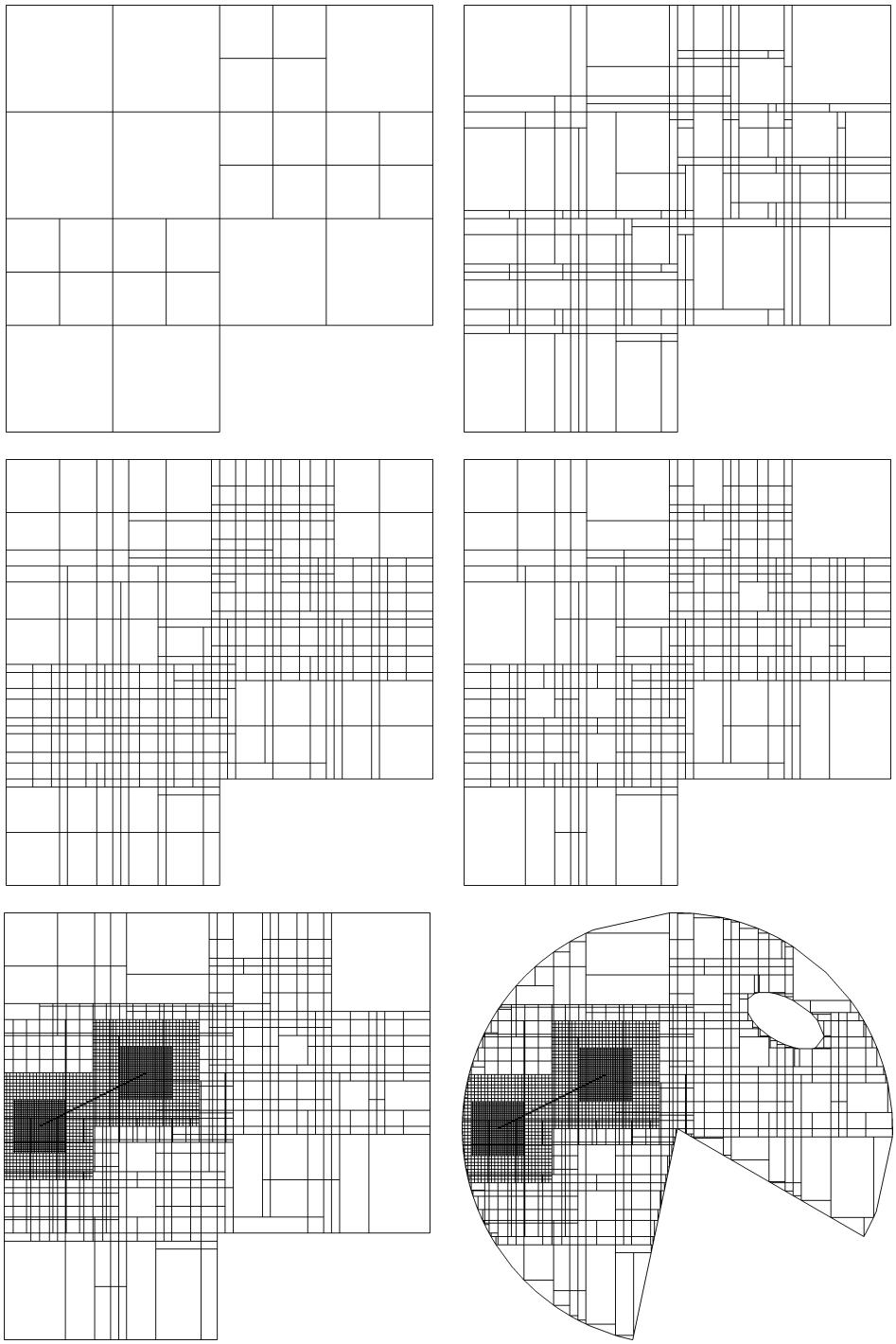


Figure 6.7. *The construction of the integration decomposition on level* 3*. Starting from tree-cells (upper left), refining by the supports ω_j (upper right), refining by the decompositions $\{sc_{i,n}\}$ associated with the weight functions (center; left: all weights, right: necessary weights), refining by the decompositions $\{ec_{i,m}\}$ associated with the enrichment functions (lower left), finally approximating the domain and boundary by parametric cells (lower right).*

integration cell can still be arbitrarily complex. We only enforce that $\partial\Omega \cap \partial ic_{i,\alpha}$ is a subset of the corners of $ic_{i,\alpha}$. Therefore the number of integration cells is not dominated by the complexity of the input data on all levels and we maintain the optimal complexity of our multilevel PPUM. From the corner configuration of an integration cell $ic_{i,\alpha}$ with $ic_{i,\alpha} \cap \partial\Omega \neq \emptyset$ we can construct an appropriate parametric mapping[2] to approximate the domain Ω and its boundary $\partial\Omega$ locally, see Figures 6.6 and 6.7. Therefore, we obtain a level-dependent approximation to the domain.

The above construction is based on the assumption that we need to integrate the product functions $\varphi_i \vartheta_i^m$; i.e. a bilinear form involving the global PPUM shape functions. Recall, however, that not all weak forms that we must consider in our PPUM Galerkin scheme are evaluated for the global shape functions $\varphi_i \vartheta_i^m$. A major advantage of the the local-to-local transfer operator of §5.2.1 over the global transfer operators is the fact that it involves the local approximation functions ϑ_i^m only. The PU functions φ_i do not need to be considered in the assembly. Hence, in the construction of the numerical integration scheme for such bilinear forms we can skip the refinement steps associated with the resolution of the algebraic structure of the PU functions φ_i and obtain less integration cells. Furthermore, on most of these integration cells we now deal with polynomial integrands (if there are no enrichment functions involved) so that these integrals can be computed very efficiently. Note that we find this situation not only in the assembly of the transfer operators but also in the construction of the direct splitting for our conforming boundary treatment, compare §4.2.2, and the computation of the stability preconditioners of §4.2.1 and §5.3.2.

Thus, we employ different decompositions of the same integration domains in our PPUM and storing these decompositions would drive-up the storage requirements of the method. Furthermore, the above decomposition approach can be implemented rather efficiently so that a computation of the required local decomposition of a patch ω_i can be carried out on the fly.

It remains to select appropriate integration rules on the constructed integration cells to obtain a complete numerical integration scheme for our PPUM. The first observation that guides us is the fact that jumps of the derivatives of our PU functions can occur only across the boundaries of the integration cells. Hence, we opt for the use of so-called open integration rules that employ integration nodes away from the boundary of the integration cells only. Recall that we can assume that all integrands are smooth on each integration cell. Hence, higher order integration rules can yield exponential convergence on each cell $ic_{i,\alpha}$ of our decomposition. For reasons of computational efficiency we are of course interested in the selection of the cheapest integration rule on each cell which does not compromise the overall convergence behavior of our PPUM. In essence we look for a result analogous to the Lemmata of Strang, see e.g. [22], in the FEM. Yet this is an open question for all meshfree Galerkin methods and we must rely on heuristic arguments to select the order of our integration rule a priori. Similarly, we can use heuristics only for the selection of appropriate stopping criteria for adaptive integration schemes on each integration cell.

Since all interior integration cells $ic_{i,\alpha}$ are axis-aligned D-rectangular, i.e. they are tensor product domains, we propose the use of sparse grid integration schemes [54] based on the univariate Gauß–Patterson rules, see Figure 6.8. These integration rules show exponential convergence for smooth integrands yet employ a substantially smaller number of integration nodes than respective tensor product rules especially in higher dimensions. Yet, on cells which involve only polynomial integrands (i.e. no enrichment functions and $\varphi_i \equiv 1$) we may also use an optimal order tensor product Gauß-Legendre scheme or we can easily compute the integrals analytically.

Observe however that it is of great importance that the overall numerical integration procedure is consistently applicable to the discretization of the operator and the discretization of the right-hand side. Furthermore, the assembly of the discrete operator must preserve key prop-

[2] Observe that the construction presented here is somewhat different from that of [47] where the authors are concerned with the approximation of the domain on a single level only and hence use additional refinements of the cover itself.

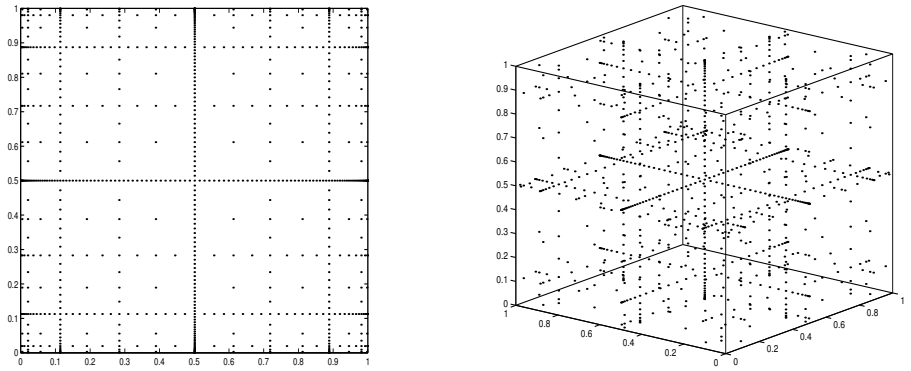


Figure 6.8. *Quadrature nodes of two sparse grid Gauß–Patterson rules, level $l = 6$ with 769 nodes in two dimension (left) and level $l = 5$ with 1023 nodes in three dimension (right).*

erties of the differential operator in its weak form. To this end let us consider the weak form $\langle \nabla \cdot, \nabla \cdot \rangle_{L^2(\Omega)}$ of the Laplacian $-\Delta$. Obviously, the constant functions c are in its kernel since $\nabla c \equiv 0$, i.e. $\langle \nabla c, \nabla \cdot \rangle_{L^2(\Omega)} \equiv 0$. This property of the *exact* operator must be preserved for the *discrete* operator; i.e. for the stiffness matrix A. Thus, we must ensure that the entries

$$A_{(i,n),(j,m)} = I_\Omega(\nabla(\varphi_j \vartheta_j^m)\nabla(\varphi_i \vartheta_i^n)) \approx \int_\Omega \nabla(\varphi_j \vartheta_j^m)\nabla(\varphi_i \vartheta_i^n)\, dx$$

of our stiffness matrix A satisfy the condition

$$\sum_{j=1}^{N} A_{(i,n),(j,0)} = 0 \quad \text{for all } (i,n) \text{ with } i = 1,\dots,N \text{ and } n = 1,\dots,\dim(V_i) \tag{6.4}$$

where we assume that $\vartheta_j^0 \equiv 1$ holds and I_Ω denotes the employed numerical integration scheme. If we compute each entry of the stiffness matrix independently the entries are in fact given by

$$A_{(i,n),(j,m)} = I_\Omega^{(i,n),(j,m)}(\nabla(\varphi_j \vartheta_j^m)\nabla(\varphi_i \vartheta_i^n)) = \sum_{q=1}^{NI} s_q^{i,j}(\nabla(\varphi_j(x_q^{i,j})\vartheta_j^m(x_q^{i,j}))\nabla(\varphi_i(x_q^{i,j})\vartheta_i^n(x_q^{i,j})))$$

where the integration nodes $x_q^{i,j}$ and weights $s_q^{i,j}$ *depend* on i, n, j and m (for the ease of notation we only employ the superscript i,j). Observe that the sums

$$\sum_{j=1}^{N} A_{(i,n),(j,0)} = \sum_{j=1}^{N} I_\Omega^{(i,n),(j,0)}(\nabla\varphi_j)\nabla(\varphi_i \vartheta_i^n)) = \sum_{j=1}^{N}\sum_{q=1}^{NI} s_q^{i,j}(\nabla\varphi_j(x_q^{i,j}))\nabla(\varphi_i(x_q^{i,j})\vartheta_i^n(x_q^{i,j}))$$

not necessarily vanish since each PU function φ_j is now evaluated at *different* locations $x_q^{i,j}$. Thus, the property $\sum_{j=1}^{N} \nabla\varphi_i(x) \equiv 0$ cannot be utilized to infer the validity of (6.4). To overcome this limitation it was suggested in [6, 12] that (6.4) is strictly enforced a posteriori by some algebraic manipulations of the assembled stiffness matrix. However it was also observed that such a procedure can have an adverse effect on the convergence behavior of the overall numerical method. Thus, we pursue a different approach to ensure the validity of (6.4) for the assembled stiffness

matrix. To this end, we enforce that the numerical integration scheme employed for the computation of the entries $A_{(i,n),(j,0)}$ is *independent* of j. Then, all φ_j are evaluated at the same location and (6.4) follows directly from the PU property

$$\sum_{j=1}^{N} \nabla\varphi_i(x) = \nabla \sum_{j=1}^{N} \varphi_i(x) \equiv \nabla 1 \equiv 0.$$

Hence, all integrals involving a particular PU function φ_j must employ the *same* integration nodes and weights. Since our decomposition $\{ic_{i,\alpha}\}$ satisfies (6.3) we only need to ensure that on a specific integration cell $ic_{i,\alpha}$ we always use the same integration rule. This of course can be easily ensured if we select the integration rule a priori. Then, each entry of the stiffness matrix can be computed independently. However, if we use an adaptive integration scheme on each cell we must check if the restrictions of *all* entries to that cell satisfy the employed stopping criteria for the same set of integration nodes and weights.

Therefore, we carry-out the assembly of the stiffness matrix not in an element-by-element fashion but rather by block-row operations. For an arbitrary but fixed patch ω_i we compute all entries

$$A_{(i,n),(j,m)} = I_{\Omega\cap\omega_i}(\nabla(\varphi_j\vartheta_j^m)\nabla(\varphi_i\vartheta_i^n)) = \sum_{q=1}^{NI} s_q(\nabla(\varphi_j(x_q)\vartheta_j^m(x_q))\nabla(\varphi_i(x_q)\vartheta_i^n(x_q)))$$

with $\omega_j \in C_i$ simultaneously for all j, m, and n as well as the respective block of the right-hand side using the same integration nodes and weights. Observe that this block-row assembly also yields a substantial reduction of operations since we evaluate all non-vanishing PU functions φ_j simultaneously. Recall that the evaluation of a single PU function φ_i at a particular point x assumes the availability of all values $W_j(x)$ for all $j \in C_i$ and the evaluation in principle requires $\mathrm{card}(C_i)$ operations. Thus, a naive element-by-element assembly approach employs

$$C_{ee} = O(\mathrm{card}(C_i)\,\dim(V_i)\,\mathrm{card}(C_j)\,\dim(V_j))$$

operations per integration node. Since we now compute all entries involving non-vanishing weights W_j and all respective local basis functions ϑ_j^m simultaneously we can re-use the values W_j to evaluate all φ_j with a minimal number of operations. Here, each weight function must be evaluated only once for the complete block-row and we obtain an optimal operation count of $C_{br} = 1$ per entry and integration node.

6.2 Visualization and Post-Processing

Another important aspect of numerical simulation in general is the visualization of the computed results and the extraction of relevant quantities from the discrete solution. There exist many visualization tools some even with a post-processing engine, however, they are usually based on low-order piecewise polynomial interpolation. Hence, the data must be imported together with a mesh or cell-complex structure.

Therefore, the accurate visualization of our meshfree solution is in general not straightforward. But with the decomposition $\{ic_{i,\alpha}\}$ of the previous section we have an unstructured collection of cells which can be processed by a fair number of visualization tools. Some care in the interpretation of the displayed results however is necessary. Due to the hanging nodes in our cell decomposition the visualization may show discontinuities at cell boundaries which are *not* present in the computed solution. Another fact to keep in mind is that the gradients of our PPUM

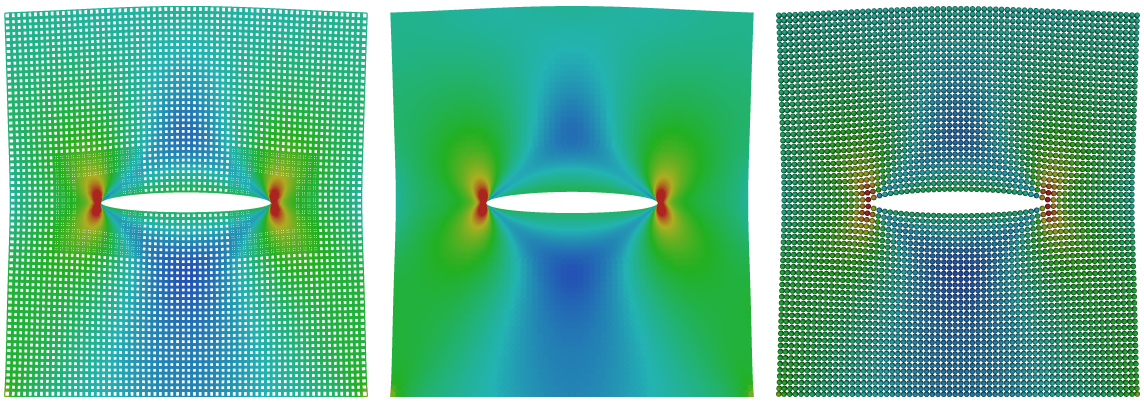


Figure 6.9. *Wireframe representation of PPUM approximation based on the decomposition* $\{ic_{i,\alpha}\}$ *(left), respective surface visualization (center), and a particle visualization (right).*

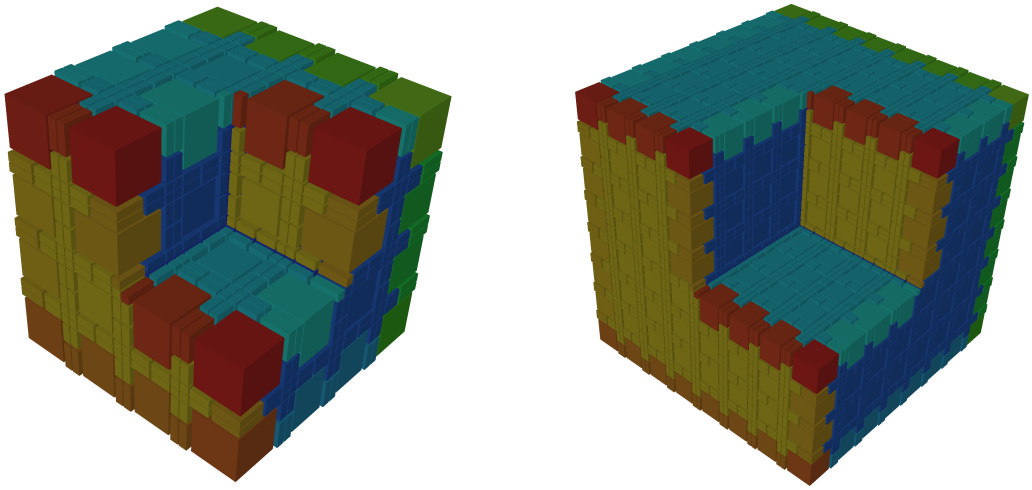


Figure 6.10. *Domain representation based on the decomposition* $\{ic_{i,\alpha}\}$ *using scaled disjoint cells.*

shape functions may have discontinuities across cell boundaries, yet unlike linear finite elements they are *not* constant within the cells.

Thus, we propose to use some additional refinements of the decomposition $\{ic_{i,\alpha}\}$ of the previous section (depending on the employed polynomial degrees p_i and the enrichment functions η_i^t) for visualization. But still we need to remind ourselves in the interpretation of the visualization that the displayed results do *not* agree with the computed results exactly, compare Figures 6.9 and 6.10.

Another approach to visualization which avoids some of these issues is a simple particle visualization. Here, only the values of the computed solution at particular points e.g. $x_i \in \mathcal{X}_N$ are displayed without interpolation between these discrete values, compare Figure 6.9. Therefore, the displayed results agree exactly with the computed data. Observe however that a PPUM space employed many more degrees of freedom than there are points $x_i \in \mathcal{X}_N$ so that the visualization gives only a coarse impression of the true result and possible oscillations between the data sites $x_i \in \mathcal{X}_N$ are not visible. To overcome this issue to some extent we may use the decomposition $\{ic_{i,\alpha}\}$ and the respective integration nodes employed on each cell for visualization.

Similarly we must be rather careful in the numerical post-processing of our computed results and should not only rely on the post-processing capabilities of the employed visualization tool. Error measurement (assuming we know the analytic or a good reference solution u) for instance requires the evaluation of the integrals

$$e_{H^1}^2 := \frac{\|\nabla(u - u^{\mathrm{PU}})\|_{L^2}}{\|\nabla u\|_{L^2}} = \frac{\displaystyle\int_\Omega |\nabla(u - u^{\mathrm{PU}})|^2 \, dx}{\displaystyle\int_\Omega |\nabla u|^2 \, dx}$$

which in general cannot be evaluated analytically. From the relative errors e_{l,H^1} and e_{l-1,H^1} on two consecutive levels of refinement we then compute the algebraic convergence rate of our PPUM via

$$\rho_{H^1} := -\frac{\log\left(\frac{e_{l,H^1}}{e_{l-1,H^1}}\right)}{\log\left(\frac{\mathsf{dof}_l}{\mathsf{dof}_{l-1}}\right)}, \quad \text{where } \mathsf{dof}_k := \sum_{i=1}^{N_k} \dim(V_{i,k}).$$

In a naive approach we use the constructed numerical integration scheme $I_{\Omega,l}$ of the previous section on the respective level l for the computation of e_{l,H^1}, i.e.

$$e_{H^1}^2 = \frac{I_{\Omega,l}(|\nabla(u - u_l^{\mathrm{PU}})|^2)}{I_{\Omega,l}(|\nabla u|^2)} \approx \frac{\displaystyle\int_\Omega |\nabla(u - u_l^{\mathrm{PU}})|^2 \, dx}{\displaystyle\int_\Omega |\nabla u|^2 \, dx}.$$

This however may be a crude approximation to the error since the construction of $I_{\Omega,l}$ was based on the assumption of integrating u_l^{PU} only. Even more problematic is the fact that the convergence rates cannot be computed reliably from these data since

$$\frac{e_{l,H^1}}{e_{l-1,H^1}} = \frac{I_{\Omega,l}(|\nabla(u - u_l^{\mathrm{PU}})|^2)}{I_{\Omega,l-1}(|\nabla(u - u_{l-1}^{\mathrm{PU}})|^2)} \frac{I_{\Omega,l-1}(|\nabla u|^2)}{I_{\Omega,l}(|\nabla u|^2)}$$

with $I_{\Omega,l-1}(|\nabla u|^2) \neq I_{\Omega,l}(|\nabla u|^2)$. Observe that even if $u_l^{\mathrm{PU}} = u_{l-1}^{\mathrm{PU}}$ holds we analogously find

$$I_{\Omega,l}(|\nabla(u - u_l^{\mathrm{PU}})|^2) \neq I_{\Omega,l-1}(|\nabla(u - u_{l-1}^{\mathrm{PU}})|^2).$$

Thus, we see that it is very much important to employ the same integration scheme to compute the convergence rate ρ_{H^1} in a reliable fashion. Obviously, it is necessary to compute e_{l,H^1} at least by the numerical integration scheme $I_{\Omega,l}$ on level l to obtain an acceptable approximation to the error. Hence, we always use the decomposition on the finest level J of our sequence of PPUM space V_k^{PU} with $k = 0, \ldots, J$ to compute the errors e_{l,H^1} and the respective convergence rates ρ_{H^1} on *all* levels $l = 0, \ldots, J$.

6.3 Parallelization and Dynamic Load Balancing

With the algorithms given in the previous sections we can carry out simulations with several hundred thousand degrees of freedom efficiently on a single processor. However, the storage limitations of a single processor machine in general render a simulation with millions of degrees of freedom not feasible. For very large simulations we must resort to distributed memory parallel computers. Hence, we need to parallelize the algorithms given above to be able to deal with large scale problems.

Our parallelization follows the data decomposition approach. Here, the main ingredients are a parallel key-based tree implementation and a space filling curve load balancing scheme. The overall method can be split into three major steps: The initial tree construction and load balancing step, the assembly step where we set up the stiffness matrices A_k on all levels $k = 0, \ldots, J$ and the interlevel transfers I_k^{k-1} and I_{k-1}^k, and finally the solution step where we use a multiplicative multilevel iteration to solve the linear system $A_J \tilde{u}_J = \hat{f}_J$. The load balancing step as well as the assembly step require some information about the neighboring patches. The neighbor search in parallel computations is the most challenging task since we need to determine the communication pattern and have to exchange the appropriate data between the processors. This is further complicated by our multilevel construction and the necessary increase in the support sizes on coarser levels.

6.3.1 Parallel Data Decomposition

In general there are two main tasks associated with the efficient parallelization of any numerical computation on distributed memory computers. The first is to evenly split up the data among the participating processors; i.e., the associated computational work should be well-balanced. The second is to allow for an efficient access to data stored by another processor; i.e. on distributed memory parallel computers also the amount of remote data needed by a processor should be small.

In a data decomposition approach we partition the data, e.g. the computational domain or mesh, among the participating processors [116]. Then, we simply restrict the operations of the global numerical method to the assigned part of the data/domain. A processor has read and write access to its local data but only read access to remote data it may need to complete its local computation. On distributed memory machines these required data have to be exchanged explicitly in distinct communication steps.

The quality of the partition of the domain/data essentially determines the efficiency of the resulting parallel computation. The local parts of the data assigned to each processor should induce a similar amount of computational work so that each processor needs roughly the same time to complete its local computation. Here, a processor may need to access the data of the neighboring sub-domains to solve its local problem. Hence, the geometry of the sub-domains should be simple to limit the number of communication steps and the communication volume. The number of neighboring processors (which determines the number of communication steps) should be small and the geometry of the local boundary (which strongly influences the communication volume) should be simple, i.e. its size should be small.

Key Based Tree Implementation

In a classical tree implementation the topology of the tree is explicitly encoded via pointers from a tree node to its successors. Such a pointer based implementation, however, is not easily parallelized especially on distributed memory machines. Hence, we use a different implementation of a D-binary tree [58, 125, 146, 147]. Here, the tree is realized with the help of a hashed associative container. To this end, a unique label is assigned to each possible tree cell and instead of linking a cell directly to its successor cells, the labeling scheme implicitly defines the topology of the tree and allows for the easy access to successors and ancestors of a particular tree cell. Furthermore, we can randomly access any cell of the tree via its unique label. This allows us to catch accesses to non-local data in parallel computations and we can easily compute the communication pattern and send and receive all necessary data to complete the local computation.

The labeling scheme must encode the topology of the tree. To this end, the labeling scheme maps a particular tree cell $\mathcal{C}_L = \prod_{i=1}^D [c_L^i, c_L^i + h_L^i] \subset \mathbb{R}^D$ to a single integer value $\mathsf{k}_L \in \mathbb{N}_0$, the

Table 6.1. *Path key values for the successor cells of a tree cell* $\mathcal{C}_L = \prod_{i=1}^{D} [c_L^i, c_L^i + h_L^i]$ *with associated key* k_L *in two dimensions.*

successor cell	binary key value	integer key value
$[c_L^1, c_L^1 + \frac{1}{2}h_L^1] \times [c_L^2, c_L^2 + \frac{1}{2}h_L^2]$	$\mathsf{k}_L 00$	$4\mathsf{k}_L$
$[c_L^1, c_L^1 + \frac{1}{2}h_L^1] \times [c_L^2 + \frac{1}{2}h_L^2, c_L^2 + h_L^2]$	$\mathsf{k}_L 01$	$4\mathsf{k}_L + 1$
$[c_L^1 + \frac{1}{2}h_L^1, c_L^1 + h_L^1] \times [c_L^2, c_L^2 + \frac{1}{2}h_L^2]$	$\mathsf{k}_L 10$	$4\mathsf{k}_L + 2$
$[c_L^1 + \frac{1}{2}h_L^1, c_L^1 + h_L^1] \times [c_L^2 + \frac{1}{2}h_L^2, c_L^2 + h_L^2]$	$\mathsf{k}_L 11$	$4\mathsf{k}_L + 3$

key. For instance, we can use the *D-binary path* as the key value k_L associated with a tree cell $\mathcal{C}_L$. The D-binary path k_L is defined by the search path that has to be completed to find the respective cell in the tree. Starting at the root of the tree, we set $\mathsf{k}_L = 1$ and descend the tree in the direction of the cell $\mathcal{C}_L$. Here we concatenate the current key value k_L (in binary representation) and the d Boolean values 0 and 1 associated with the decisions to which successor cell the descent continues to reach the respective tree cell $\mathcal{C}_L$. In Table 6.1 we give the resulting path key values k_L for a two dimensional example. Note that the key value $\mathsf{k}_L = 1$ for the root cell is essentially a stop bit which is necessary to ensure the uniqueness of the key values.

Parallel Key Based Tree Implementation

The data structure which describes the computational domain in our PUM is a D-binary tree (quadtree, octree) used for the cover construction and the fast neighbor search for the evaluation of the Shepard PU functions (3.13). The use of a global unique integer key for each cell of the tree allows for a simple description of a partitioning of the computational domain. The set of all admissible[3] keys $\{0, 1, \ldots, \mathsf{k}_{\max}\}$ is simply split into $\wp$ subsets which are then assigned to the $\wp$ processors. We subdivide the range of keys into $\wp$ intervals

$$0 = r_0 \leq r_1 \leq \cdots \leq r_\wp = \mathsf{k}_{\max}$$

and assign the interval $[r_q, r_{q+1})$ to the qth processor, i.e. the set of tree cells assigned to the qth processor is $\{\mathcal{C}_L \,|\, \mathsf{k}_L \in [r_q, r_{q+1})\}$. With this very simple decomposition each processor can identify which processor stores a particular tree cell $\mathcal{C}_L$. A processor has to compute only the key value k_L for the tree cell $\mathcal{C}_L$ and the respective interval $[r_q, r_{q+1})$ with $\mathsf{k}_L \in [r_q, r_{q+1})$ to determine the processor q which stores this tree cell $\mathcal{C}_L$. The question now arises if such a partition of the domain with the path keys k_L is a reasonable choice? Obviously the partitioning of the tree should be done in such a fashion that complete sub-trees are assigned to a processor to allow for efficient tree traversals. But the path key labeling scheme given above orders the tree cells rather horizontally (see Figure 6.11) instead of vertically. Therefore, we need to transform the path keys k_L to so-called domain keys k_L^D.

A simple transformation which leads to a vertical ordering of the tree cells is the following: First, we remove the leading bit (the initial root key value) from the key's binary representation. Then we shift the remaining bits all the way to the left so that the leading bit of the path information is now stored in the most significant bit.[4] Assume that the key values are stored as an 32 bit integer and that we are in two dimensions. Then this simple transformation of a path key value

[3] The maximal key value $\mathsf{k}_{\max}$ is a constant depending on the architecture of the parallel computer.

[4] This transformation needs $O(1)$ operations if we assume that the current refinement level of the tree is known, otherwise it is of the order $O(J)$, where J denotes the number of levels of the tree.

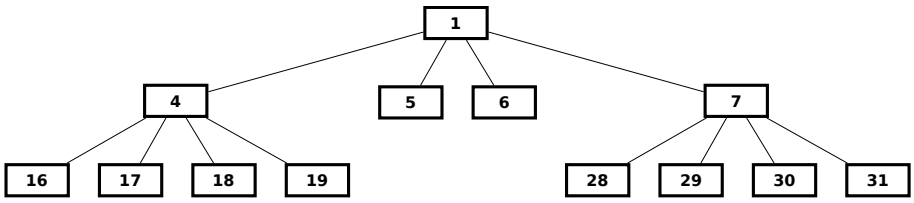


Figure 6.11. *Horizontal ordering of a tree induced by the path key values* k.

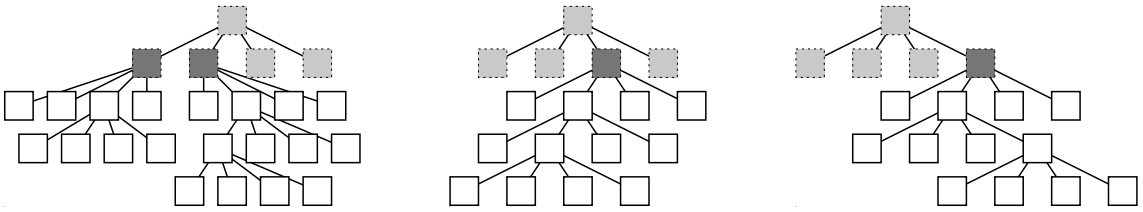


Figure 6.12. *Common global tree (dashed, gray shaded) for a partition onto three processors. Local sub-tree roots (dark gray shaded) and the local sub-tree cells (white) for the first (left), second (center) and third processor (right).*

k_L to a respective domain key value k_L^D e.g. yields

$$\begin{aligned} \mathsf{k}_L^D &= \underbrace{01110010}_{\text{path}}000000000000000000000000 \quad \text{for} \\ \mathsf{k}_L &= 000000000000000000000001\underbrace{01110010}_{\text{path}}. \end{aligned} \tag{6.5}$$

With these domain keys k_L^D the tree is now ordered vertically and we can assign complete sub-trees to a processor using the simple interval domain description $[r_q, r_{q+1})$.

Remark 6.1. Note that the transformed keys are no longer unique and cannot be used as the key value for the associative container to store the tree itself. Obviously, a successor cell $\mathcal{C}_S$ of a tree cell $\mathcal{C}_L$ can be assigned the same domain key as the tree cell, i.e. $\mathsf{k}_S^D = \mathsf{k}_L^D$. Hence, we use the unique path keys k_L for the container and the associated domain keys k_L^D for the domain description, i.e. for the associated interval boundaries $[r_q, r_{q+1})$.

Note that the description of the data partition via the intervals $[r_q, r_{q+1})$ defines a minimal refinement stage of the tree which has to be present on all processors to insure the consistency of the tree. In the following we refer to this top part of the tree as the *common global tree*. The leaves $\mathcal{C}_L$ of the common global tree are characterized by the fact that they are the coarsest tree cells for which all possible successor cells are stored on the same processor, see Figure 6.12. The domain key values k_S^D of all possible successor cells $\mathcal{C}_S$ lie in the same interval $[r_q, r_{q+1})$ as the domain key k_L^D. We therefore refer to the leaves of the common global tree as *local sub-tree roots*.

The order of the tree cells induced by the domain keys k_L^D given above is often referred to as bit-interleaving, the Morton-order, the Z-order or the N-order. The curve induced by mapping the domain keys to the associated cell centers corresponds to the Lebesgue curve (Figure 6.13 (upper left)) which is a space filling curve [118]. There are many space filling curves with different properties which might be more suitable for our needs; e.g. the sub-domains generated by the Lebesgue curve may be not connected [159] even for a D-rectangle, see Figure 6.13 (upper right). This increases the size of the local boundary and thereby the communication volume and possibly

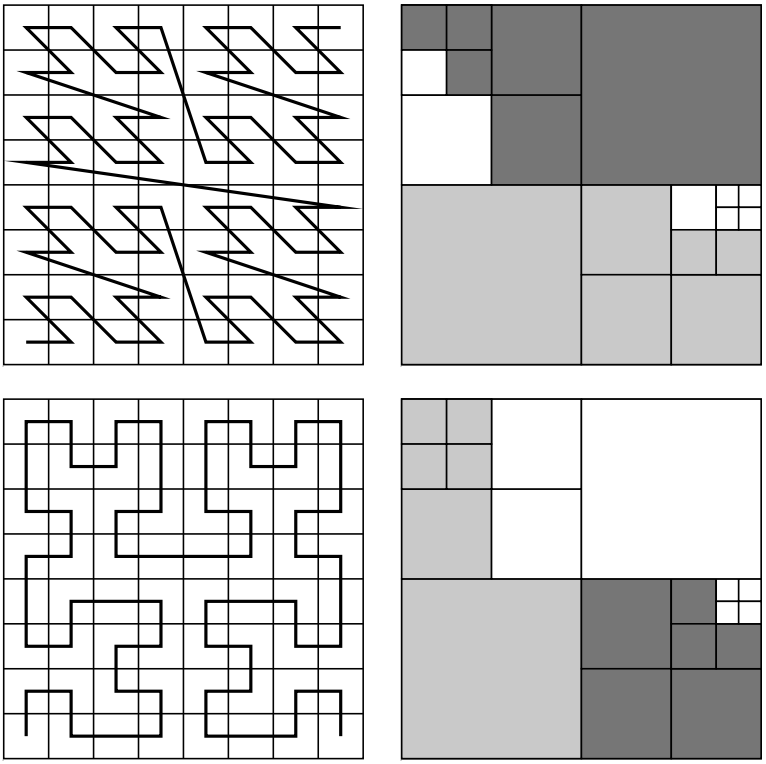


Figure 6.13. *The Lebesgue curve (upper left) and the constructed sub-domains (upper right) for a partition onto three processors. The sub-domains are not connected since the curve does not have the locality property. The Hilbert curve (lower left) and the constructed sub-domains (lower right) for a partition onto three processors. The sub-domains are connected due to the locality property of the curve.*

the number of communication steps.

6.3.2 Load Balancing with Space Filling Curves

The properties of space filling curves with respect to partitioning data for parallel computations have been studied in [158, 159]. Here, it turns out that the Hilbert curve (Figure 6.13 (lower left)) is more suitable for partitioning irregular data than the Lebesgue curve, see Figure 6.14. It provides a better data locality, e.g. the constructed sub-domains for a D-rectangle are connected (Figure 6.13 (lower right)) and the size of the local boundaries is of optimal order. Hence, we use the Hilbert curve instead of the Lebesgue curve to order the tree in our implementation; i.e., we use a different transformation than (6.5) to map the path keys k_L to domain keys k_L^D. This transformation of the path key values to Hilbert curve keys is more involved than the transformation (6.5) to Lebesgue curve keys, but it can also be realized with fast bit manipulations, see [125, Appendix B] for details.[5] By changing the interval boundaries $\{r_q \,|\, q = 0, \ldots, \wp\}$, which describe the decomposition of our tree, we can balance the load among the processors. To this end we assign estimated work loads w_L as weights to the leaves $\mathcal{C}_L$ of the tree. Then we compute the current

[5] In general the transformation of a given key k_L to its associated Hilbert domain key k_L^D requires $O(J)$ operations, even if the current tree level J is known. But since we are interested in the domain keys k_L^D keys for all cells (or at least for all leaves) of the tree we can merge the transformation with the tree traversal which reduces the complexity of the transformation of a single key to $O(1)$.

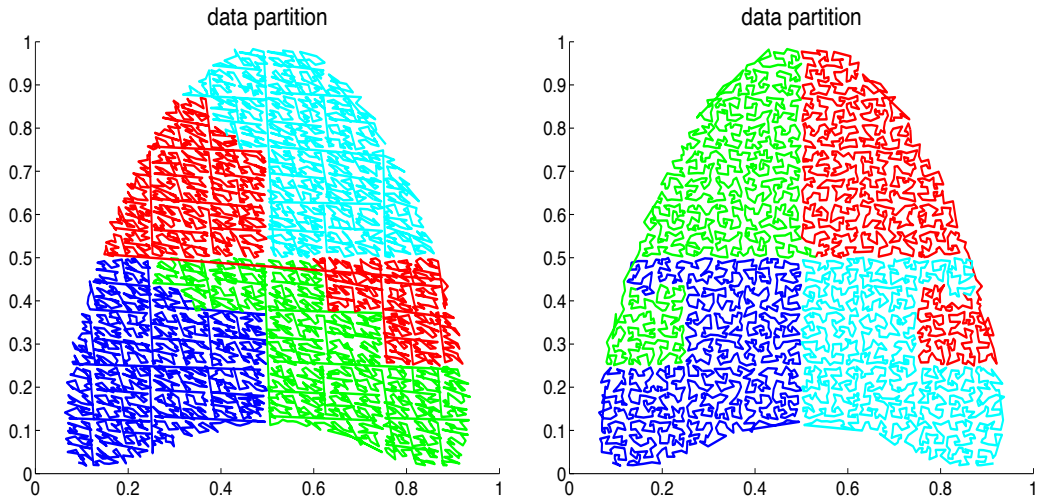


Figure 6.14. *A partition of the point set P_J onto four processors (color coded) using the domain keys k_L^D based on the Lebesgue curve (left), and the domain keys k_L^D based on the Hilbert curve (right).*

load estimate $w^{\hat{q}} = \sum w_L$ on every processor $\hat{q}$ and gather all remote load estimates w^q with $q \neq \hat{q}$. In the next step, the global load estimate $w = \sum_{q=0}^{\wp-1} w^q$, and the balanced load distribution $w_b^q = \frac{qw}{\wp}$ are computed. Then, every processor $\hat{q}$ iterates over its current set of leaves $\mathcal{C}_L$ of the tree in ascending order of the domain keys k_L^D and sets new (intermediate and inconsistent) local interval boundaries $\{\tilde{r}_q^{\hat{q}} \,|\, q = 0, \ldots, \wp\}$ accordingly. Finally, a reduction operation over all (local intermediate) sets $\{\tilde{r}_q^{\tilde{q}} \,|\, q = 0, \ldots, \wp\}$ of the $\wp$ participating processors $\tilde{q}$ gives the new (global and consistent) interval boundaries $\{r_q \,|\, q = 0, \ldots, \wp\}$ which balance the estimated load w. Note that this load balancing scheme itself is completed in parallel.

Algorithm 6.1 (Load Balancing)**.**

1. For all local leaves $\mathcal{C}_L$ of the tree:

   Assign estimated work load w_L.

2. Compute local estimate $w^{\hat{q}} = \sum_L w_L$ (on processor $\hat{q}$).

3. Gather remote estimates w^q with $q = 0, \ldots, \wp - 1$ and $q \neq \hat{q}$.

4. Compute global load estimate $w = \sum_{q=0}^{\wp-1} w^q$.

5. Set local estimate $w_g^{\hat{q}} = \sum_{q=0}^{q<\hat{q}} w^q$ (on processor $\hat{q}$).

6. Set balanced load distribution $w_b^q = \frac{qw}{\wp}$ for $q = 0, \ldots, \wp$.

7. For all local leaves $\mathcal{C}_L$ (in ascending order of domain keys k_L^D):

   Set local intermediate interval boundary $\tilde{r}_q^{\hat{q}} = k_L^D$ (on processor $\hat{q}$) where $q \in \{0, \ldots, \wp\}$ is the smallest integer with $w_g^{\hat{q}} \leq w_b^q$ and update estimate $w_g^{\hat{q}} = w_g^{\hat{q}} + w_L$.

8. Set (global) interval boundaries $r_q = \max_{\tilde{q}} \tilde{r}_q^{\tilde{q}}$ for all $q \in \{0, \ldots, \wp\}$ by reducing the set of all (local) intermediate boundaries $\{\tilde{r}_q^{\tilde{q}}\}$ over all processors $\tilde{q}$, force $r_0 = 0$ and $r_\wp = \mathsf{k}_{\max}$.

The quality of the load balancing scheme is essentially determined by the local load estimate w_L. With respect to the finest level J we can estimate the computational work associated with a particular patch $\omega_{i,k}$ for instance by the number of degrees of freedom assigned to the patch. Such an estimate is very cheap to compute, however, its quality is rather poor. We can obtain a very accurate estimate of the local computational work by considering the number of integration points employed on a particular patch. Such an estimate however cannot be attained with reasonable effort in general. The next best local load estimate on a patch ω_i is the number of the integration cells $ic_{i,\alpha} \in \omega_i$ which is still rather expensive to compute. If the local approximation spaces V_i employ only polynomials we can estimate the number of integration cells (at least away from the boundary) by the number of neighbors $\omega_j \in C_i$. Thus, if we assume that the integration of all nonzero entries of the stiffness matrix can be computed with a similar amount of work, we can use the number of nonzeros per block-row as our load estimate, i.e.,

$$w_L = w_{i,J} = \dim(V_{i,J}^{p_{i,J}}) \sum_{\omega_k \in C_{i,J}} \dim(V_{k,J}^{p_{k,J}}).$$

Hence, to balance the load with this work load estimate, we need to compute all neighborhoods $C_{i,J}$ (in parallel) on the finest level J (in the un-balanced tree). Thus, the computation of an accurate load estimate for parallel simulations is already a challenging task.

Remark 6.2. The computational cost associated with the estimation of the current load can often be reduced. In a time-dependent setting or in adaptive refinement we usually have a pretty good load estimate from a previous time step or a coarser level without extra computations. This estimate can either be used directly to partition the data or it can be updated with a few operations only. Furthermore, we typically have to re-distribute only a small amount of data in these situations.

Remark 6.3. With the load balancing scheme given in Algorithm 6.1 we balance the load with respect to the finest level J only. For our PUM discretization this is sufficient, since the largest amount of work is due to the finest level and we already assume that we coarsen the number of patches at a constant rate from level to level to obtain an optimal complexity multilevel iteration. However, sometimes it might be necessary to balance the load with respect to all levels simultaneously. Then, we need to modify the load balancing scheme in such a way that we consider all nodes of the tree rather than only the leaves. Note, however, that the design of an appropriate load estimate is somewhat more involved in such cases.

Remark 6.4. In the case of a PDE with (piecewise) constant coefficients we can employ a caching technique in some regions of the domain. Hence, not all integrals associated with the stiffness matrix are computed explicitly, many entries are computed only once and re-used in the assembly. Thus, when this caching technique is employed we need to update our load estimate to account for this change in computational work. Yet, we must be aware that the load has to balanced also with respect to the solution phase, i.e., the scalability of a single matrix-vector product should be retained.

6.3.3 Parallel Cover Construction

Now that the computational domain is partitioned in an appropriate fashion among the processors we turn to the algorithmic changes for our parallel implementation, e.g. the computation of the communication pattern. The first task in our PUM is the multilevel cover construction which is essentially a post-order tree operation. Due to our tree decomposition which assigns complete sub-trees to processors most work can be done completely in parallel. When we reach elements of the common global tree we need to gather the respective tree cells from remote processors.

Then, all processors can complete the cover construction on the common global tree. The parallel version of the multilevel cover construction algorithm (compare Algorithm 5.1) reads as:

Algorithm 6.2 (Parallel Multilevel Cover Construction)**.**

1. Given the domain $\Omega \subset \mathbb{R}^D$ and a bounding box $R_\Omega = \prod_{i=1}^D [l_\Omega^i, u_\Omega^i] \supset \overline{\Omega}$.
2. Given the interval boundaries $\{r_q \,|\, q = 0, \ldots, \wp\}$ and the local part $\mathcal{X}_{N,\hat{q}}$ of the initial point set $\mathcal{X}_N = \{x_j \,|\, x_j \in \overline{\Omega}, j = 1, \ldots, \tilde{N}\}$, i.e. $\mathsf{k}_j^D \in [r_{\hat{q}}, r_{\hat{q}+1})$ for all $x_j \in \mathcal{X}_{N,\hat{q}}$.[6]
3. Initialize the common global D-binary tree (quadtree, octree) according to the $\wp$ intervals $[r_q, r_{q+1})$.
4. Build parallel D-binary sub-trees over local sub-tree roots, such that per leaf L at most one $x_i \in \mathcal{X}_{N,\hat{q}}$ lies within the associated cell $\mathcal{C}_L := \prod_{i=1}^D [l_L^i, u_L^i]$.
5. Set J to the finest refinement level of the global tree.
6. For all local sub-tree root cells $\mathcal{C}_L = \prod_{d=1}^D [l_L^d, u_L^d] = R_\Omega$:
   (a) If current tree cell $\mathcal{C}_L$ is an INNER tree node and $\mathcal{C}_L \cap \Omega \neq \emptyset$:
      i. Descend tree for all successors $\mathcal{C}_S$ of $\mathcal{C}_L$. (→ 6(a))
      ii. Set patch
      $$\omega_L := \prod_{d=1}^D [x_L^d - h_L^d, x_L^d + h_L^d] \supset \mathcal{C}_L$$
      where $x_L := \frac{1}{2^D} \sum x_S$ is the center of its successor points x_S and $h_L^i := 2 \max_S h_S^i$ is twice the maximum radius of its successors h_S^i.
      iii. Set active levels to $[k_L^{\min}, k_L^{\max}]$ with
      $$k_L^{\min} = k_L^{\max} = \min_{\mathcal{C}_S \subset \mathcal{C}_L} k_S^{\min} - 1$$
      and update for all successors $k_{S'}^{\min} = \min_{\mathcal{C}_S \subset \mathcal{C}_L} k_S^{\min}$.
      iv. Set polynomial degrees $p_{L,k} = p_{\min}$ for all $k \in [k_L^{\min}, k_L^{\max}]$ where
      $$p_{\min} := \min_{\mathcal{C}_S \subset \mathcal{C}_L} \min_{q \in [k_S^{\min}, k_S^{\max}]} p_{q,S}.$$
   (b) Else if $\mathcal{C}_L \cap \Omega \neq \emptyset$:
      i. Set patch
      $$\omega_L := \prod_{d=1}^D [x_L^d - h_L^d, x_L^d + h_L^d] \supset \mathcal{C}_L, \quad \text{and} \quad h_L^d := \frac{\alpha}{2}(u_L^d - l_L^d) \tag{6.6}$$
      where
      $$x_L^d := l_L^d + \frac{1}{2}(u_L^d - l_L^d) \quad \text{with} \quad \alpha > 1. \tag{6.7}$$

[6] An initial partition can easily be constructed by choosing uniform interval boundaries $\{r_q\}$ and partitioning the initial point set $\tilde{P}$ according to the domain keys on the finest possible tree level.

ii. Set active levels to $[k_L^{\min}, k_L^{\max}]$ with $k_L^{\min} = k_L^{\max} = J$.

iii. Set polynomial degrees $p_{L,k} = p$ to some given value p for all $k \in [k_L^{\min}, k_L^{\max}]$.

7. For $k = 0, \ldots, J-1$:

   Set $C_{\Omega,\hat{q}}^k = \{\omega_L \,|\, k_L^{\min} \leq k \leq k_L^{\max}\}$.

8. Broadcast patches ω_L associated with local sub-tree roots $\mathcal{C}_L$ to all processors.

9. For the common global root cell $\mathcal{C}_L = \prod_{i=1}^d [l_L^i, u_L^i] = R_\Omega$:

   (a) If current tree cell $\mathcal{C}_L$ is not the root of any complete processor sub-tree, and an INNER tree node with $\mathcal{C}_L \cap \Omega \neq \emptyset$:

   i. Descend tree for all successors $\mathcal{C}_S$ of $\mathcal{C}_L$. (→ 9(a))

   ii. Set patch

   $$\omega_L := \prod_{d=1}^{D} [x_L^d - h_L^d, x_L^d + h_L^d] \supset \mathcal{C}_L$$

   where $x_L := \frac{1}{2^D} \sum x_S$ is the center of its successor points x_S and $h_L^i := 2 \max_S h_S^i$ is twice the maximum radius of its successors h_S^i.

   iii. Set active levels to $[k_L^{\min}, k_L^{\max}]$ with

   $$k_L^{\min} = k_L^{\max} = \min_{\mathcal{C}_S \subset \mathcal{C}_L} k_S^{\min} - 1$$

   and update for all successors $k_{S'}^{\min} = \min_{\mathcal{C}_S \subset \mathcal{C}_L} k_S^{\min}$.

   iv. Set polynomial degrees $p_{L,k} = p_{\min}$ for all $k \in [k_L^{\min}, k_L^{\max}]$ where

   $$p_{\min} := \min_{\mathcal{C}_S \subset \mathcal{C}_L} \min_{q \in [k_S^{\min}, k_S^{\max}]} p_{q,S}.$$

10. For $k = 0, \ldots, J-1$:

    Set $C_{\Omega,\hat{q}}^k = \{\omega_L \,|\, k_L^{\min} \leq k \leq k_L^{\max} \text{ and } \mathrm{k}_L^D \in [r_{\hat{q}}, r_{\hat{q}+1})\}$.

Note that the main difference between this parallel cover construction algorithm and Algorithm 5.1 is the use of different entry points for insert operations into the (global) tree. With Algorithm 5.1 we always insert points starting at the (global) root of the tree whereas in parallel each processor will essentially insert points into one of its local sub-tree roots only. Therefore, Algorithm 6.2 will yield the same sequence of covers C_Ω^k as Algorithm 5.1 only if the initial common global tree of step 3 (which is induced by the interval boundaries of step 2) is reasonable, compare section 6.3.1. Otherwise there can be slight differences in the constructed covers using different processor numbers $\wp$ for small initial point sets $\mathcal{X}_N$.

6.3.4 Parallel Discretization

After the sequence of covers C_Ω^k with $k = 0, \ldots, J$ is constructed in a distributed fashion we can formally define the respective PPUM spaces

$$V_k^{\mathrm{PU}} = \sum_{i=1}^{N_k} \varphi_{i,k} V_{i,k} = \sum_{i=1}^{N_k} \varphi_{i,k} \mathcal{P}^{p_{i,k}} + \sum_{i=1}^{N_k} \varphi_{i,k} \mathcal{E}_{i,k}$$

by local operations only. On each processor $\hat{q}$ we define the sub-space

$$V_{k,\hat{q}}^{\mathrm{PU}} = \sum_{\omega_{i,k} \in C_{\Omega,\hat{q}}^k} \varphi_{i,k} V_{i,k} = \sum_{\omega_{i,k} \in C_{\Omega,\hat{q}}^k} \varphi_{i,k} \mathcal{P}^{p_{i,k}} + \sum_{\omega_{i,k} \in C_{\Omega,\hat{q}}^k} \varphi_{i,k} \mathcal{E}_{i,k}$$

associated with the local part $C_{\Omega,\hat{q}}^k$ of the cover C_Ω^k. This partitioning of the PPUM space V_k^{PU} is then used to partition the stiffness matrix A_k on level k in block-rows; i.e. via the test space. To this end, we simply restrict the assembly of the stiffness matrix (and the transfer operators) on each of the $\wp$ processors to the block-rows associated with its assigned patches $\omega_{i,k}$. A processor $\hat{q}$ computes all block-entries

$$(A_k)_{i,j} = (A_{k(i,n),(j,m)}), \text{ with } A_{k(i,n),(j,m)} = a(\varphi_{j,k}\vartheta_{j,k}^m, \varphi_{i,k}\vartheta_{i,k}^n) \in \mathbb{R}, \tag{6.8}$$

where $\varphi_{i,k}$ is the PU function associated with one of its assigned patches $\omega_{i,k}$, i.e., the domain key $\mathsf{k}_{i,k}^D = \mathsf{k}_i^D$ associated with the patch $\omega_{i,k}$ is element of $[r_{\hat{q}}, r_{\hat{q}+1})$.

However, the evaluation of a PU function $\varphi_{i,k}$ on a processor $\hat{q}$ requires the availability of its neighboring patches $\omega_{j,k} \in C_{i,k}$. Although most neighbors $\omega_{j,k}$ of a patch $\omega_{i,k}$ are stored on the same processor due the locality of the employed data partition, the patch $\omega_{i,k}$ may well overlap patches which are stored on a remote processor. Hence, a processor may need copies of certain patches from a remote processor for the computation of the neighborhoods

$$C_{i,k,\tilde{k}} = \{\omega_{j,\tilde{k}} \in C_\Omega^{\tilde{k}} \,|\, \omega_{i,k} \cap \omega_{j,\tilde{k}} \neq \emptyset\}.$$

Recall that the neighborhoods $C_{i,k} = C_{i,k,k}$ also determine the sparsity pattern of the stiffness matrix A_k on level $k = 0, \dots, J$. The interlevel neighborhoods $C_{i,k,\tilde{k}}$ with $k \neq \tilde{k}$ determine the sparsity pattern of the global transfer operators of §5.2.1. The neighborhoods $C_{i,k,\tilde{k}}$ are also the only information needed for the construction of the decomposition of the respective integration domains, compare §6.1.

The computation of a single block-entry $(A_k)_{i,j}$ involves $\varphi_{i,k}$ and $\varphi_{j,k}$. Hence, it seems that we not only need remote patches $\omega_{j,k}$ but also all their neighbors $\omega_{l,k} \in C_{j,k}$ for the evaluation of the integrands involved in the block-row corresponding to the local patch $\omega_{i,k}$. This would significantly increase the communication volume and storage overhead due to parallelization. But since all function evaluations of $\varphi_{j,k}$ are restricted to the support of $\varphi_{i,k}$—recall that the integration domain for the block entry is $\Omega \cap \omega_{i,k} \cap \omega_{j,k}$—every neighboring patch $\omega_{l,k} \in C_{j,k}$ that contributes a nonzero weight $W_{l,k}$ to the PU function $\varphi_{j,k}$ (on the integration domain) must also be a neighbor of $\omega_{i,k}$. Hence we only need those neighbors $\omega_{l,k} \in C_{j,k}$ of $\omega_{j,k} \in C_i$ which satisfy

$$\omega_{l,k} \cap \omega_{i,k} \neq \emptyset \quad \Rightarrow \quad \omega_{l,k} \in C_{i,k}.$$

Hence, it is sufficient to store copies of remote patches $\omega_{j,k}$ which are direct neighbors of a local patch $\omega_{i,k}$. There is no need to store neighbors of neighbors for the assembly of the stiffness matrix.

But how does a processor $\hat{q}$ detect which neighbors $\omega_{j,k}$ exist on a remote processor $q \neq \hat{q}$ for one of its local patches $\omega_{i,k}$? In fact, a processor $\hat{q}$ cannot determine which patches $\omega_{j,k}$ to request from a remote processor $q \neq \hat{q}$. But a processor can certainly determine which of its local patches $\omega_{i,k}$ overlap the remote sub-trees with the help of the leaves of the common global tree. Hence, a processor can compute which of its local patches $\omega_{i,k}$ a remote processor may need to complete its neighbor search. Thus, it seems that we need to perform only a parallel communication step where a processor sends its local patches $\omega_{i,k}$ which overlap the remote sub-trees prior to the computation of the neighborhoods $C_{i,k}$.

However this procedure may yield a very large yet unnecessary communication volume if applied based on the current common global tree patches. The reason for this behavior is that our multilevel cover construction is designed for the construction of a sequence of valid covers C_Ω^k which essentially means that the absolute size of the overlap regions grow with decreasing levels. Therefore, we can encounter the situation that

$$\omega_{i,k} \cap \omega_{j,\tilde{k}} \neq \emptyset$$

where $\omega_{j,\tilde{k}}$ is an element of the common global tree yet all of the successors $\omega_{l,k} \subset \omega_{j,\tilde{k}}$ of $\omega_{j,\tilde{k}}$ on level k satisfy

$$\omega_{i,k} \cap \omega_{l,k} = \emptyset.$$

Hence, the patch $\omega_{i,k}$ must not be communicated for the neighbor search on level k. The elements $\omega_{j,\tilde{k}}$ of the common global tree can be much larger than the union of their successor patches $\omega_{l,k}$ on levels $k > \tilde{k}$. Therefore, the communication volume is substantially overestimated which can compromise the overall optimal complexity of our implementation.

To cure this we need to store a separate copy of the common global tree which consists of patches with minimal extension. To this end, we need to modify steps 6(a)ii and 9(a)ii of Algorithm 6.2 slightly and define these patches as the minimal bounding box of its successors. After we have computed these minimal patches we compute our cover sequence C_Ω^k with $k = 0 \ldots, J$ as before with Algorithm 6.2.

Any processor $\hat{q}$ can now compute which of its patches $\omega_{i,k}$ is very likely to overlap a remote patch $\omega_{j,k}$ on processor q and must be communicated to enable processor q to compute the neighborhoods $C_{j,k}$ with a minimal number of operations and communication. Once the neighbors are exchanged between the processors the assembly of the stiffness matrix, the transfer operators and the right-hand side vector can be carried out locally without any communication.

The solution of the resulting linear system with our multilevel solver of course requires some parallel communication in each iteration, i.e. in each matrix-vector product and each application of the smoother. The communication pattern of these operations are given by the sparsity structure of the respective matrices. The sparsity structure of any PPUM matrix however is defined by the respective neighborhoods $C_{i,k,\tilde{k}}$ which are known, thus the necessary data exchange can be easily implemented.

Thus, it remains to consider the parallelization of the error estimation and the adaptive refinement (compare step 4 of Algorithm 5.5). The error estimation essentially involves two matrix-vector products for the evaluation of the right-hand sides, compare §5.4.1, which require communication. Furthermore, the computation of the activity classifier requires the exchange of additional neighbor information. Again, all required communication patterns and communication volumes are known via the neighborhoods $C_{i,k}$. Only the refinement step requires further attention since we generate new patches and modify the local approximation spaces $V_{i,J}$ on some patches $\omega_{i,J}$ *locally*. Let us first consider the modification of a patch $\omega_{i,J}$ on processor $\hat{q}$. If this patch has a remote neighbor $\omega_{j,J} \in C_{i,J}$ on processor $q \neq \hat{q}$ then a copy of $\omega_{i,J}$ was previously sent to processor q for the neighbor search. This copy must now be updated; i.e., we must send the modified patch again to processor q. Note however that we not only need to check for the neighbors $\omega_{j,J} \in C_{i,J}$ but also the neighbors $\omega_{j,J-1} \in C_{i,J,J-1}$ or the neighbors $\omega_{j,J-1} \in C_{i,J,J-1}^H$ involved in the transfer operators.

Let us now consider the generation of new patches in the refinement step. If the refinement lead to the construction of a new level we obviously need to compute the neighborhoods with respect to the newly generated level. Here, we determine the remote neighbors of a newly generated patch simply by checking the neighborhood of its ancestor. If the ancestor does have a remote neighbor on processor q it is very likely that its successor patches will have neighboring patches on processor q as well. More importantly, if the ancestor does not have a remote neighbor

on processor q then it is guaranteed that its successors do not have remote neighbors on processor q since the ancestor completely covers the union of the supports of its successors. Thus, we compute the communication pattern on the new level directly from the communication pattern on level J. Similarly we can update the local neighborhood $C_{i,J}$ if no new level was generated.

Finally, we must check our local load estimates to determine whether we should re-balance the load among the processors to obtain an optimal scaling of our computations.

Chapter 7
Validation

In this section we are concerned with the numerical validation of the presented theoretical properties of our PPUM. To this end, we consider the scalar reaction-diffusion problem

$$\begin{aligned} -\Delta u + cu &= f && \text{in } \Omega \subset \mathbb{R}^D, \\ \frac{\partial u}{\partial n} &= g_N && \text{on } \Gamma_N \subset \overline{\Omega}, \\ u &= g_D && \text{on } \Gamma_D = \partial\Omega \setminus \Gamma_N, \end{aligned} \tag{7.1}$$

and the equations of linear elasticity

$$\begin{aligned} -\mathbf{div}\,\boldsymbol{\sigma}(u) &= f && \text{in } \Omega \subset \mathbb{R}^D, \\ \boldsymbol{\sigma}(u) \cdot n &= g_N && \text{on } \Gamma_N \subset \overline{\Omega}, \\ u &= g_D && \text{on } \Gamma_D = \partial\Omega \setminus \Gamma_N, \end{aligned} \tag{7.2}$$

where the symmetric stress tensor $\boldsymbol{\sigma}(u)$ and the symmetric strain tensor $\boldsymbol{\epsilon}(u)$ are defined as

$$\boldsymbol{\sigma}(u) := \mathcal{C} \cdot \boldsymbol{\epsilon}(u), \quad \text{and} \quad \boldsymbol{\epsilon}(u) := \frac{1}{2}\Big(\nabla u + (\nabla u)^T\Big)$$

and $\mathcal{C} = \mathcal{C}(E, \nu)$ denote material parameters.

First, we focus on the approximation properties of our PPUM when applied to the Galerkin discretization of (7.1) and (7.2). Here we distinguish three major cases which determine the convergence behavior of our PPUM:

1. a uniform global refinement of the approximation space $V^{\mathrm{PU}} = \sum_{i=1}^{N} \varphi_i V_i$,
2. an adaptive local refinement of V^{PU}; i.e., of the the cover patches $\omega_i = \mathrm{supp}^{\circ}(\varphi_i)$ or the local approximation spaces V_i,
3. and a refinement of the PPUM space V^{PU} with (hierarchical) enrichment.

Recall that uniform global refinement (without enrichment) attains its optimal convergence behavior only if the sought solution is regular whereas an adaptive refinement and hierarchical enrichment lead to an optimal convergence behavior also for singular solutions.

Moreover, we study the effect of boundary conditions and their implementation on the approximation properties of our PPUM. To this end, we compare the results obtained for a (definite) Neumann problem with those computed for the respective Dirichlet problem using the

non-conforming approach due to Nitsche of §3.2.2 and the conforming boundary treatment of §4.2.2.

The second major topic of this section is the efficiency of the presented multilevel solver. Here, we study the dependence of the convergence behavior of our iterative schemes with respect to the type of boundary conditions (Neumann or Dirichlet) and their implementation (conforming or non-conforming). Furthermore, we are concerned with the influence of the employed local approximation spaces (higher order or enriched) on the speed of convergence.

7.1 Approximation Properties

To determine the approximation properties of our PPUM numerically, we introduce some shorthand notation for various norms of the error $u - u^{\mathrm{PU}}$, i.e., we define

$$e_{L^\infty} := \frac{\|u - u^{\mathrm{PU}}\|_{L^\infty}}{\|u\|_{L^\infty}}, \quad e_{L^2} := \frac{\|u - u^{\mathrm{PU}}\|_{L^2}}{\|u\|_{L^2}}, \quad \text{and} \quad e_{H^1} := \frac{\|u - u^{\mathrm{PU}}\|_{H^1}}{\|u\|_{H^1}}. \tag{7.3}$$

For each of these error norms we compute the respective algebraic convergence rate ρ as

$$\rho := -\frac{\log\left(\frac{\|u-u_l^{\mathrm{PU}}\|}{\|u-u_{l-1}^{\mathrm{PU}}\|}\right)}{\log(\frac{\mathsf{dof}_l}{\mathsf{dof}_{l-1}})}, \qquad \text{where} \quad \mathsf{dof}_m := \dim(V_m^{\mathrm{PU}}) = \sum_{i=1}^{N_m} \dim(V_{i,m}), \tag{7.4}$$

by considering the error norms of two consecutive levels $l-1$ and l. Hence the optimal rate ρ_{H^1} of an uniformly h-refined sequence of spaces with $p_{i,k} = p$ for all $i = 1, \dots, N_k$ and $k = 0, \dots, J$ for a sufficiently smooth solution u is $\rho_{H^1} = \frac{p}{D}$ where D denotes the spatial dimension of $\Omega \subset \mathbb{R}^D$. This corresponds to the classical h^γ notation with $\gamma = \rho D = p$.

According to §6.1 the reliable approximation of the error norms (7.3) requires the use of an appropriate numerical integration scheme. Furthermore, this integration scheme should be employed on all levels $k = 0, \dots, J$ to ensure the comparability of the computed results and the evaluation of the rates (7.4) from the relative errors (7.3). Therefore, we always employ an h-refined version of the numerical integration scheme of §6.1 constructed on the finest level J for the evaluation of (7.3).

If the analytical solution is u is not available, we assess the convergence behavior of our PPUM via a reference solution u_{ref}. This reference solution is obtained by a global hp-refinement of the finest approximation level J. In these cases we employ an h-refined version of the numerical integration scheme constructed on the reference level $J+1$ for the approximation of the respective relative errors (7.3) and convergence rates (7.4).

In those experiments where we use the conforming boundary treatment presented in §4.2.2 we do not give the total number of degrees of freedom dof of (7.4) but rather the splitting

$$\sum_{i=1}^{N_m} \dim(V_{i,m}) = \mathsf{dof}_m = \mathsf{dof}_{m,K} + \mathsf{dof}_{m,I} = \sum_{i=1}^{N_m} \dim(V_{i,m,K}) + \sum_{i=1}^{N_m} \dim(V_{i,m,I})$$

where $\mathsf{dof}_{m,I}$ denotes the number of degrees of freedom employed in the approximation of the essential boundary conditions on Γ_D and $\mathsf{dof}_{m,K} = \mathsf{dof}_m - \mathsf{dof}_{m,I}$ is the dimension of the conforming subspace $V_{m,K}^{\mathrm{PU}} \subset V_m^{\mathrm{PU}}$ on level m. Observe though that the convergence rates (7.4) are computed with respect to the total number of degrees of freedom dof_m.

Table 7.1. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.1 with Neumann boundary conditions.*

J	dof	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	12	4	4.594_{-1}	–	4.008_{-1}	–	8.391_{-1}	–
2	48	16	2.750_{-1}	0.37	1.543_{-1}	0.69	6.137_{-1}	0.23
3	192	64	1.707_{-1}	0.34	4.854_{-2}	0.83	3.649_{-1}	0.38
4	768	256	6.705_{-2}	0.67	1.314_{-2}	0.94	1.907_{-1}	0.47
5	3072	1024	2.167_{-2}	0.81	3.405_{-3}	0.97	9.674_{-2}	0.49
6	12288	4096	6.509_{-3}	0.87	8.667_{-4}	0.99	4.875_{-2}	0.49
7	49152	16384	1.882_{-3}	0.89	2.182_{-4}	0.99	2.432_{-2}	0.50
8	196608	65536	5.320_{-4}	0.91	5.470_{-5}	1.00	1.209_{-2}	0.50
9	786432	262144	1.480_{-4}	0.92	1.369_{-5}	1.00	6.164_{-3}	0.49
10	3145728	1048576	4.070_{-5}	0.93	3.424_{-6}	1.00	3.162_{-3}	0.48
11	12582912	4194304	1.109_{-5}	0.94	8.559_{-7}	1.00	1.473_{-3}	0.55
12	50331648	16777216	3.004_{-6}	0.94	2.125_{-7}	1.00	7.625_{-4}	0.47

7.1.1 Uniform Refinement

In the following examples we consider a uniform global refinement of our PPUM space

$$V^{\mathrm{PU}} := \sum_{i=1}^{N} \varphi_i V_i = \sum_{i=1}^{N} \varphi_i \mathcal{P}^{p_i}$$

without enrichment in two and three space dimensions. As local approximation spaces V_i we employ polynomials $\mathcal{P}^{p_i}$ of total degree p_i with tensor products of univariate Legendre polynomials as basis functions $\mathcal{P}^{p_i} = \mathrm{span}\langle \psi_i^t \rangle$.

Example 7.1 (Scalar problem with regular solution)**.** In our first example, we consider the reaction-diffusion problem (7.1) with $c = 1$ and Neumann boundary conditions on the domain $(0,1)^2$ where we choose the data f and g_N such that the analytical solution is given by

$$u(x_1, x_2) = \arctan\Big(100\big(\frac{x_1 + x_2}{\sqrt{2}} - 0.8\big)\big(x_1 - x_1^2\big)\big(x_2 - x_2^2\big)\Big). \tag{7.5}$$

Here, we use linear polynomials as local approximation space, i.e. $p_i = 1$. Thus, we anticipate to find a reduction of the error in the L^2-norm e_{L^2} by 0.25 due to the refinement of level l to level $l+1$. This behavior corresponds to a rate (7.4) in the L^2-norm of $\rho_{L^2} = 1$ (i.e. $\gamma_{L^2} = 2$). With respect to the H^1-norm we expect to find $\rho_{H^1} = 0.5$ (i.e. $\gamma_{H^1} = 1$).

The results of this numerical experiment are summarized in Table 7.1 and Figure 7.1. From the measured relative errors (7.3) and the respective convergence rates (7.4) we clearly observe the anticipated convergence behavior with $\rho_{L^2} = 1$ and $\rho_{H^1} = 0.5$. Furthermore, this optimal behavior is attained on all levels $l \geq 4$, compare Figure 7.1. Note that the considered model problem can be regarded as the simplest test case for our PPUM since it involves natural boundary conditions only and always yields a symmetric positive definite stiffness matrix. Thus, there are no explicit constraints acting on the PPUM function space V^{PU} which could adversely affect its approximation properties.

Let us now consider a slightly more challenging model problem. Here we are concerned with the approximation of (7.1) with $c = 0$ and Dirichlet boundary conditions which either requires the use of a non-conforming approach like Nitsche's method of §3.2.2 or the conforming boundary treatment of §4.2.2 to obtain a positive definite stiffness matrix. This however means that certain components of our PPUM space are explicitly constrained which may have an adverse effect on the absolute value of the obtained errors (7.3) (or in principle even on the convergence rates (7.4)).

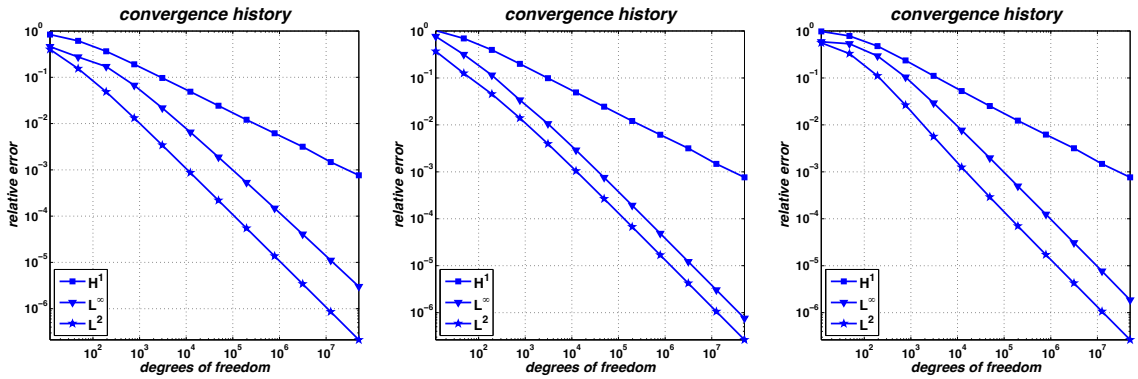


Figure 7.1. *Convergence history for Example 7.1 in the L^∞-norm, the L^2-norm, and the H^1-norm (left: Neumann boundary conditions; center: Dirichlet boundary conditions via Nitsche's method; right: Dirichlet boundary conditions via conforming local approximation spaces).*

Table 7.2. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.1 with Dirichlet boundary conditions implemented via Nitsche's method.*

J	dof	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	12	4	7.660_{-1}	–	3.699_{-1}	–	1.024_{0}	–
2	48	16	3.156_{-1}	0.64	1.258_{-1}	0.78	6.958_{-1}	0.28
3	192	64	1.145_{-1}	0.73	4.540_{-2}	0.74	3.957_{-1}	0.41
4	768	256	3.381_{-2}	0.88	1.397_{-2}	0.85	1.997_{-1}	0.49
5	3072	1024	1.055_{-2}	0.84	3.937_{-3}	0.91	9.884_{-2}	0.51
6	12288	4096	2.893_{-3}	0.93	1.042_{-3}	0.96	4.920_{-2}	0.50
7	49152	16384	7.555_{-4}	0.97	2.663_{-4}	0.98	2.442_{-2}	0.51
8	196608	65536	1.929_{-4}	0.98	6.713_{-5}	0.99	1.211_{-2}	0.51
9	786432	262144	4.868_{-5}	0.99	1.684_{-5}	1.00	6.169_{-3}	0.49
10	3145728	1048576	1.221_{-5}	1.00	4.215_{-6}	1.00	3.163_{-3}	0.48
11	12582912	4194304	3.049_{-6}	1.00	1.054_{-6}	1.00	1.473_{-3}	0.55
12	50331648	16777216	7.573_{-7}	1.00	2.636_{-7}	1.00	7.626_{-4}	0.48

We choose the data f and g_D such that the analytical solution u is again given by (7.5) to allow for a direct comparison of the results. In Table 7.2 we give the measured relative errors and the respective convergence rates for a PPUM discretization of (7.1) using Nitsche's method with the minimal regularization parameter, see Figure 7.1. Observe that we attain the optimal convergence rates $\rho_{L^2} = 1$ and $\rho_{H^1} = 0.5$ also in this experiment. Comparing the absolute values of the measured errors of Tables 7.2 and 7.1 we find an almost perfect agreement, see also Figure 7.1. Thus, the approximation properties of our PPUM are not compromised by Nitsche's method.

Next, we consider the use of the conforming technique of §4.2.2 for the implementation of the essential boundary conditions. The respective results are collected in Table 7.3. Again, we find the optimal convergence behavior with $\rho_{L^2} = 1$ and $\rho_{H^1} = 0.5$ also in this case. Moreover, the absolute values of the measured relative errors (7.3) agree almost perfectly with those of Tables 7.2 and 7.1, see also Figure 7.1. Here, the approximation properties of our PPUM seem to be completely independent of the considered boundary conditions and the employed implementation of essential boundary data.

As the last experimental configuration of this example we consider the scalar model problem (7.1) with $c = 0$ on the domain $(-1,1)^3$ and Dirichlet boundary conditions on

$$\Gamma_D := \{(x_1, x_2, x_3) \in [-1,1]^3 \,|\, x_d = -1 \text{ for at least one } d = 1,2,3\}$$

Table 7.3. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.1 with Dirichlet boundary conditions implemented via the algebraic conforming boundary treatment of §4.2.2.*

J	dof_K	dof_I	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	0	12	4	5.893_{-1}	–	5.605_{-1}	–	9.799_{-1}	–
2	20	28	16	5.353_{-1}	0.07	3.302_{-1}	0.38	7.865_{-1}	0.16
3	132	60	64	2.940_{-1}	0.43	1.110_{-1}	0.79	4.766_{-1}	0.36
4	644	124	256	1.041_{-1}	0.75	2.631_{-2}	1.04	2.365_{-1}	0.51
5	2820	252	1024	2.922_{-2}	0.92	5.639_{-3}	1.11	1.109_{-1}	0.55
6	11780	508	4096	7.691_{-3}	0.96	1.248_{-3}	1.09	5.256_{-2}	0.54
7	48132	1020	16384	1.974_{-3}	0.98	2.907_{-4}	1.05	2.529_{-2}	0.53
8	194564	2044	65536	4.956_{-4}	1.00	7.006_{-5}	1.03	1.234_{-2}	0.52
9	782340	4092	262144	1.244_{-4}	1.00	1.719_{-5}	1.01	6.225_{-3}	0.49
10	3137540	8188	1048576	3.082_{-5}	1.01	4.259_{-6}	1.01	3.177_{-3}	0.49
11	12566532	16380	4194304	7.669_{-6}	1.00	1.059_{-6}	1.00	1.477_{-3}	0.55
12	50298884	32764	16777216	1.862_{-6}	1.02	2.643_{-7}	1.00	7.635_{-4}	0.48

Table 7.4. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.1 in three dimensions using the algebraic conforming boundary treatment of §4.2.2 for the implementation of essential boundary conditions.*

J	dof_K	dof_I	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	12	20	8	7.290_{0}	–	1.691_{0}	–	1.063_{0}	–
2	168	88	64	2.251_{0}	0.56	9.595_{-1}	0.27	9.675_{-1}	0.05
3	1680	368	512	2.023_{0}	0.05	4.912_{-1}	0.32	7.128_{-1}	0.15
4	14880	1504	4096	7.524_{-1}	0.48	1.897_{-1}	0.46	4.415_{-1}	0.23
5	124992	6080	32768	2.499_{-1}	0.53	5.574_{-2}	0.59	2.393_{-1}	0.29
6	1024128	24448	262144	7.430_{-2}	0.58	1.433_{-2}	0.65	1.173_{-1}	0.34
7	8290560	98048	2097152	2.042_{-2}	0.62	3.572_{-3}	0.67	5.952_{-2}	0.33

and Neumann boundary conditions on $\Gamma_N := \partial\Omega \setminus \Gamma_D$. We realize the essential boundary conditions via the construction of conforming local approximation spaces $V_{i,K} \subset V_i = \mathcal{P}^1$. Here, we choose the data f, g_D and g_N such that the analytical solution is given by

$$u(x) = \sin\Big(2\pi\Big(\sum_{d=1}^{D}\Big(\Big(\frac{3}{4}+\frac{d}{3}\Big)x_d\Big)\Big)\Big). \tag{7.6}$$

In three dimensions the optimal rates ρ_{L^2} and ρ_{H^1} that correspond to the optimal classical convergence behavior with $\gamma_{L^2} = 2$ and $\gamma_{H^1} = 1$ are $\rho_{L^2} = \frac{2}{3}$ and $\rho_{H^1} = \frac{1}{3}$. From the measured errors given in Table 7.4 we see that the optimal convergence behavior is attained also in this three dimensional experiment with mixed boundary conditions.

Example 7.2 (Higher order discretization)**.** In our second example we focus on the approximation properties of higher order PPUM discretizations. To this end, we first consider the approximation of (7.1) with $c = 0$ on the domain $(-1,1)^2$ and Dirichlet boundary conditions on

$$\Gamma_D := \{(x_1,x_2) \in [-1,1]^2 \,|\, x_d = -1 \text{ for at least one } d = 1,2\}$$

and Neumann boundary conditions on $\Gamma_N := \partial\Omega \setminus \Gamma_D$ using cubic polynomials as local approximation spaces $V_i = \mathcal{P}^3$. Thus, we expect to find the convergence rates $\rho_{L^2} = 2$ and $\rho_{H^1} = 1.5$ for the approximation of the analytical solution (7.6) independent of the implementation of the essential boundary conditions. Again, we compare the results obtained by Nitsche's method with the minimal regularization parameter, summarized in Table 7.5, and those attained with conforming local approximation spaces, see Table 7.6. Observe that we find the anticipated optimal conver-

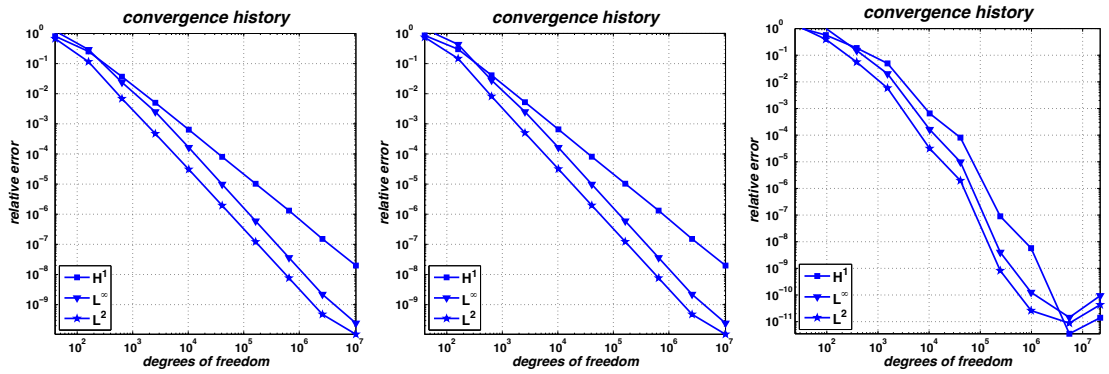


Figure 7.2. *Convergence history for Example 7.2 in the L^∞-norm, the L^2-norm, and the H^1-norm using cubic polynomials and Nitsche's method (left), or the conforming approach (center). Convergence history for Example 7.2 in the L^∞-norm, the L^2-norm, and the H^1-norm using an alternating h-refinement and hp-refinement (right).*

Table 7.5. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.2 with $p = 3$ using Nitsche's method.*

J	dof	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	40	4	1.213_{0}	–	6.755_{-1}	–	8.124_{-1}	–
2	160	16	2.920_{-1}	1.03	1.151_{-1}	1.28	2.526_{-1}	0.84
3	640	64	2.392_{-2}	1.80	6.879_{-3}	2.03	3.672_{-2}	1.39
4	2560	256	2.516_{-3}	1.62	4.692_{-4}	1.94	5.003_{-3}	1.44
5	10240	1024	1.633_{-4}	1.97	3.075_{-5}	1.97	6.429_{-4}	1.48
6	40960	4096	9.843_{-6}	2.03	1.943_{-6}	1.99	8.042_{-5}	1.50
7	163840	16384	5.932_{-7}	2.03	1.215_{-7}	2.00	1.028_{-5}	1.48
8	655360	65536	3.605_{-8}	2.02	7.593_{-9}	2.00	1.320_{-6}	1.48
9	2621440	262144	2.188_{-9}	2.02	4.736_{-10}	2.00	1.514_{-7}	1.56
10	10485760	1048576	2.418_{-10}	1.59	1.037_{-10}	1.10	1.983_{-8}	1.47

gence behavior regardless of the approach employed for the implementation of essential boundary conditions. For both techniques we measure the convergence rates $\rho_{L^2} = 2$ and $\rho_{H^1} = 1.5$ and the absolute values of the relative errors (7.3) agree very well also for higher order discretizations, compare Figure 7.2. The reduction in the measured convergence rates ρ_{L^∞} and ρ_{L^2} in the last refinement step are round-off effects (see below).

Finally, we consider a sequence of PPUM spaces V_k^{PU} with $k = 1, \ldots, J$ resulting from alternating a global uniform h-refinement and a global uniform hp-refinement step for the approximation of (7.1) with the analytical solution (7.6). Up to level $k = 4$ we employ quadratic polynomials as local approximation spaces, then we continue to apply a global hp-refinement step followed by a global h-refinement step. Hence, we expect to find the algebraic convergence rates $\rho_{L^2} = \frac{p+1}{2}$ and $\rho_{H^1} = \frac{p}{2}$ on all odd levels $k > 4$ and a kind of exponential convergence on all even levels $k \geq 4$. The measured relative errors and convergence rates are collected in Table 7.7. From the numbers displayed there we learn that we attain the anticipated convergence behavior on all levels $k = 1, \ldots, 9$. In the last refinement step from level 9 to level 10 with quintic polynomials as local approximation spaces the measured convergence breaks down. We attribute this to the fact that the relative error in the L^∞-norm is already on the scale of machine precision which renders the numerical integration employed in the approximation of the L^2-error and the H^1-error rather challenging.

Table 7.6. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.2 with p = 3 and the algebraic conforming boundary approach of §4.2.2 for the implementation of essential boundary conditions.*

J	dof_K	dof_I	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	25	15	4	1.415_{0}	–	7.486_{-1}	–	8.607_{-1}	–
2	129	31	16	4.310_{-1}	0.86	1.467_{-1}	1.18	3.003_{-1}	0.76
3	577	63	64	2.804_{-2}	1.97	8.166_{-3}	2.08	4.082_{-2}	1.44
4	2433	127	256	2.516_{-3}	1.74	5.003_{-4}	2.01	5.227_{-3}	1.48
5	9985	255	1024	1.633_{-4}	1.97	3.163_{-5}	1.99	6.561_{-4}	1.50
6	40449	511	4096	9.846_{-6}	2.03	1.970_{-6}	2.00	8.123_{-5}	1.51
7	162817	1023	16384	5.934_{-7}	2.03	1.224_{-7}	2.00	1.033_{-5}	1.49
8	653313	2047	65536	3.606_{-8}	2.02	7.619_{-9}	2.00	1.323_{-6}	1.48
9	2617345	4095	262144	2.189_{-9}	2.02	4.744_{-10}	2.00	1.516_{-7}	1.56
10	10477569	8191	1048576	2.413_{-10}	1.59	1.037_{-10}	1.10	1.984_{-8}	1.47

Table 7.7. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.2 with global hp-refinement and the algebraic conforming boundary approach of §4.2.2 for the implementation of essential boundary conditions.*

J	dof_K	dof_I	N	p	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	13	11	4	2	2.714_{0}	–	1.409_{0}	–	1.185_{0}	–
2	73	23	16	2	1.024_{0}	0.70	3.950_{-1}	0.92	5.735_{-1}	0.52
3	337	47	64	2	1.512_{-1}	1.38	5.544_{-2}	1.42	1.872_{-1}	0.81
4	1441	95	256	2	2.014_{-2}	1.45	5.858_{-3}	1.62	4.971_{-2}	0.96
5	9985	255	1024	3	1.633_{-4}	2.54	3.163_{-5}	2.75	6.561_{-4}	2.28
6	40449	511	4096	3	9.846_{-6}	2.03	1.970_{-6}	2.00	8.123_{-5}	1.51
7	244481	1279	16384	4	4.006_{-9}	4.36	8.234_{-10}	4.34	8.993_{-8}	3.80
8	980481	2559	65536	4	1.254_{-10}	2.50	2.610_{-11}	2.49	5.766_{-9}	1.98
9	5498881	6143	262144	5	1.439_{-11}	1.26	8.785_{-12}	0.63	3.525_{-12}	4.30
10	22007809	12287	1048576	5	9.436_{-11}	−1.36	4.329_{-11}	−1.15	1.404_{-11}	−1.00

Example 7.3 (System of equations with regular solution)**.** Let us now turn to the discretization of systems of PDEs by our PPUM. To this end, we consider the equations of linear elasticity (7.2) with the material parameters $E = 1000$ and $\nu = 0.3$ on the domain $\Omega := (-1,1)^D$ with $D = 2,3$ and employ linear polynomials as local approximation spaces $V_i = \mathcal{P}^1$. The boundary conditions are given by

$$u \cdot n = 0 \text{ on } \Gamma_D, \quad (\sigma(u) \cdot n) \cdot t = 0 \text{ on } \Gamma_D, \quad \text{and} \quad \sigma(u) \cdot n = x \cdot n + 1 \text{ on } \Gamma_N := \partial\Omega \setminus \Gamma_D$$

with the Dirichlet boundary $\Gamma_D := \{x \in (-1,1)^D \,|\, x_d = -1 \text{ for at least one } d = 1,\dots,D\}$. The essential boundary conditions are implemented by the conforming approach of §4.2.2. In this example we use a reference solution to assess the performance of our PPUM. To this end, we employ a uniform global hp-refinement step of the finest computational level J. In two dimensions the reference solution is computed with roughly 200 million degrees of freedom and in three space dimension with 8 million degrees of freedom. The relative errors (7.3) with respect to the reference solution are approximated by an numerical integration scheme based on the decomposition of §6.1 constructed for the reference PPUM space.

The measured errors and the respective convergence rates are summarized in Tables 7.8 and 7.9. We anticipate to find the convergence rates $\rho_{L^2} = 1$ and $\rho_{H^1} = 0.5$ in two dimensions and $\rho_{L^2} = \frac{2}{3}$ and $\rho_{H^1} = \frac{1}{3}$ in three dimensions as in Example 7.1. The results given clearly show this optimal convergence behavior also for the discretization of systems of PDEs.

The results of Examples 7.1, 7.2 and 7.3 where we considered the approximation of smooth solutions on simple D-rectangular domains show that we attain the optimal convergence behav-

Table 7.8. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.3 in two dimensions using the algebraic conforming boundary treatment of §4.2.2 for the implementation of essential boundary conditions.*

J	dof_K	dof_I	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
4	1472	64	256	1.052_{-2}	–	5.682_{-3}	–	5.769_{-2}	–
5	6016	128	1024	3.525_{-3}	0.79	1.408_{-3}	1.01	2.827_{-2}	0.51
6	24320	256	4096	1.100_{-3}	0.84	3.442_{-4}	1.02	1.396_{-2}	0.51
7	97792	512	16384	3.291_{-4}	0.87	8.450_{-5}	1.01	6.940_{-3}	0.50
8	392192	1024	65536	9.556_{-5}	0.89	2.089_{-5}	1.01	3.459_{-3}	0.50
9	1570816	2048	262144	2.725_{-5}	0.91	5.190_{-6}	1.00	1.719_{-3}	0.50
10	6287360	4096	1048576	7.627_{-6}	0.92	1.293_{-6}	1.00	8.540_{-4}	0.50
11	25157632	8192	4194304	2.106_{-6}	0.93	3.221_{-7}	1.00	4.059_{-4}	0.54

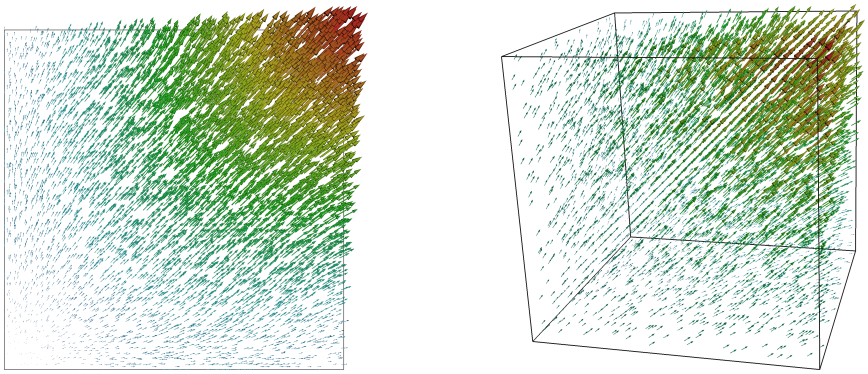


Figure 7.3. *Vector field representation of the computed solution two dimensions (left) and in three dimensions (right).*

ior of our PPUM according to Theorem 3.1. Here, the uniform h-version converges with the rates $\rho_{L^2} = \frac{p+1}{D}$ and $\rho_{H^1} = \frac{p}{D}$ for a discretization with polynomials of total degree p in D space dimensions independent of the type and implementation of the boundary conditions.

Let us now consider the approximation of singular solutions. Here, the uniform h-version (without enrichment) cannot achieve its optimal convergence rate due to the limited regularity of the solution. Nonetheless, the study of the performance of a uniformly refined PPUM approximation for problems with singularities due to geometry and boundary conditions is highly instructive to understand the differences in the treatment of essential boundary conditions by Nitsche's method and via the use of conforming local approximation spaces.

Example 7.4 (Problems with singular solution)**.** Recall that for convex domains our PU satisfies not only the flat top condition in the domain Ω but also on the boundary $\partial\Omega$. In the case of a non-convex domain however the flat top condition on the boundary may not be satisfied, compare Figure 4.3. Thus, the conforming approach of §4.2.2 constructs a splitting

$$V^{\mathrm{PU}} = V_K^{\mathrm{PU}} \oplus V_I^{\mathrm{PU}}$$

of our PPUM space V^{PU} which is sub-optimal only. The dimension of the conforming subspace V_K^{PU} used for the approximation of the PDE in the domain Ω maybe smaller than the dimension of the kernel of the trace operator applied to V^{PU}. Thus, we might use more degrees of freedom for the approximation of the boundary conditions than necessary and less degrees of freedom in the domain than possible. This however does not compromise the convergence rate of our PPUM yet the absolute value of the attained errors may be affected.

Table 7.9. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.3 in three dimensions using the algebraic conforming boundary treatment of §4.2.2 for the implementation of essential boundary conditions.*

J	dof_K	dof_I	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	60	36	8	2.103_{-1}	–	1.179_{-1}	–	2.996_{-1}	–
2	624	144	64	9.027_{-2}	0.41	5.218_{-2}	0.39	1.855_{-1}	0.23
3	5568	576	512	4.334_{-2}	0.35	1.838_{-2}	0.50	9.968_{-2}	0.30
4	46848	2304	4096	1.646_{-2}	0.47	5.467_{-3}	0.58	5.417_{-2}	0.29
5	384000	9216	32768	1.432_{-2}	0.07	1.355_{-3}	0.67	3.334_{-2}	0.23

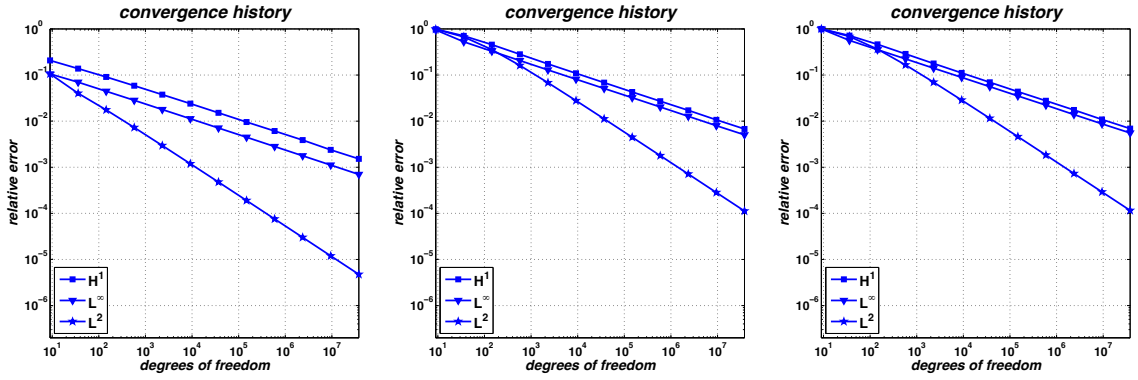


Figure 7.4. *Convergence history for Example 7.4 in the L^∞-norm, the L^2-norm, and the H^1-norm (left: Nitsche's method with minimal regularization; center: conforming local approximation spaces; right: Nitsche's method with large regularization).*

The standard reference domain for the study of singular solutions due to boundary conditions in two dimensions is the L-shaped domain $\Omega := (-1,1)^2 \setminus [0,1)^2$. On Ω we consider the model problem (7.1) with $c = 0$ and homogeneous Dirichlet boundary conditions on

$$\Gamma_D := \{(x_1, x_2) \in \overline{\Omega} \,|\, x_1 = 0, \text{ or } x_2 = 0\} \tag{7.7}$$

and Neumann boundary data g_N on $\Gamma_N := \partial\Omega \setminus \Gamma_D$ such that the solution u is given by

$$u(r,\theta) = r^{\frac{2}{3}} \sin\left(\frac{2\theta - \pi}{3}\right) \tag{7.8}$$

where $(r(x), \theta(x))$ denote polar coordinates. The solution (7.8) is singular at the re-entrant corner $(0,0)$ and is in H^s with $s < 1 + \frac{2}{3}$ only. Thus, we expect to find the convergence rates $\rho_{L^2} = \frac{2}{3}$ and $\rho_{H^1} = \frac{1}{3}$ for a uniformly refined sequence of PPUM spaces V_k^{PU} with $k = 0, \ldots, J$.

Note that each cover C_Ω^k of the sequence constructed by Algorithm 5.1 based on the minimal bounding box $[-1,1]^2 \supset \Omega$ contains three patches ω_i that cover the re-entrant corner at $(0,0)$ (and part of both edges running into the corner), compare Figure 4.3. Thus, a PPUM discretization according to Algorithm 4.2 will construct a direct splitting of the respective local approximation spaces V_i defined on these patches ω_i into $V_{i,K}$ and $V_{i,I}$ where $V_{i,K}$ is the kernel of the localized trace operator applied to V_i. This kernel however is empty for all three patches ω_i since we currently limit ourselves to the use of polynomials as local approximation spaces V_i (independent of the employed polynomial degree). Hence, all three local approximation spaces V_i which overlap the re-entrant corner are used completely for the approximation of the (vanishing) boundary

Table 7.10. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.4 with Dirichlet boundary conditions on Γ_D of (7.7) implemented via Nitsche's method.*

J	dof	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	9	3	1.058_{-1}	–	1.060_{-1}	–	2.076_{-1}	–
2	36	12	7.014_{-2}	0.30	4.015_{-2}	0.70	1.378_{-1}	0.30
3	144	48	4.456_{-2}	0.33	1.753_{-2}	0.60	9.095_{-2}	0.30
4	576	192	2.836_{-2}	0.33	7.277_{-3}	0.63	5.883_{-2}	0.31
5	2304	768	1.789_{-2}	0.33	2.959_{-3}	0.65	3.769_{-2}	0.32
6	9216	3072	1.128_{-2}	0.33	1.190_{-3}	0.66	2.400_{-2}	0.33
7	36864	12288	7.104_{-3}	0.33	4.762_{-4}	0.66	1.519_{-2}	0.33
8	147456	49152	4.475_{-3}	0.33	1.899_{-4}	0.66	9.584_{-3}	0.33
9	589824	196608	2.818_{-3}	0.33	7.558_{-5}	0.66	6.102_{-3}	0.33
10	2359296	786432	1.774_{-3}	0.33	3.005_{-5}	0.67	3.894_{-3}	0.32
11	9437184	3145728	1.116_{-3}	0.33	1.194_{-5}	0.67	2.386_{-3}	0.35
12	37748736	12582912	7.005_{-4}	0.34	4.737_{-6}	0.67	1.523_{-3}	0.32

Table 7.11. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.4 with Dirichlet boundary conditions on Γ_D of (7.7) implemented via the algebraic conforming boundary treatment of §4.2.2 for the implementation of essential boundary conditions.*

J	dof_K	dof_I	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	0	9	3	1.000_{0}	–	1.000_{0}	–	1	–
2	23	13	12	5.654_{-1}	0.41	6.872_{-1}	0.27	7.194_{-1}	0.24
3	123	21	48	3.552_{-1}	0.34	3.651_{-1}	0.46	4.645_{-1}	0.32
4	539	37	192	2.243_{-1}	0.33	1.657_{-1}	0.57	2.872_{-1}	0.35
5	2235	69	768	1.413_{-1}	0.33	7.000_{-2}	0.62	1.780_{-1}	0.35
6	9083	133	3072	8.898_{-2}	0.33	2.862_{-2}	0.65	1.111_{-1}	0.34
7	36603	261	12288	5.607_{-2}	0.33	1.153_{-2}	0.66	6.968_{-2}	0.34
8	146939	517	49152	3.528_{-2}	0.33	4.613_{-3}	0.66	4.378_{-2}	0.34
9	588795	1029	196608	2.224_{-2}	0.33	1.839_{-3}	0.66	2.766_{-2}	0.33
10	2357243	2053	786432	1.392_{-2}	0.34	7.315_{-4}	0.66	1.750_{-2}	0.33
11	9433083	4101	3145728	8.746_{-3}	0.34	2.907_{-4}	0.67	1.089_{-2}	0.34
12	37740539	8197	12582912	5.538_{-3}	0.33	1.155_{-4}	0.67	6.899_{-3}	0.33

conditions in the conforming approach. With Nitsche's method however the boundary conditions act on the product functions $\varphi_i \vartheta_i^m$ rather than on the local approximation functions ϑ_i^m and this non-conforming technique balances the error in the domain with the error on the boundary (weighted by the regularization parameter). Therefore, we expect that the results obtained by Nitsche's method and a conforming discretization will not agree with respect to the absolute values of the attained errors (7.3) — unlike in the previous examples.

Let us first consider a non-conforming discretization via Nitsche's methods with the *minimal* regularization parameter. The errors (7.3) and the respective convergence rates ρ are given in Table 7.10, compare Figure 7.4. These measurements clearly show the anticipated convergence behavior with the rates $\rho_{L^2} = \frac{2}{3}$ and $\rho_{H^1} = \frac{1}{3}$. On level $J = 12$ we find the absolute values of the relative errors $e_{L^2} = 4.737_{-6}$ and $e_{H^1} = 1.523_{-3}$. Comparing these results with those obtained by a conforming discretization, see Table 7.11, we clearly see the effect of the much more restrictive conforming boundary treatment. Here, the discretization on level $J = 12$ yields the relative errors $e_{L^2} = 1.155_{-4}$ and $e_{H^1} = 6.899_{-3}$ only. Yet the asymptotic convergence rates of the conforming approach also agree with the optimal attainable values of $\rho_{L^2} = \frac{2}{3}$ and $\rho_{H^1} = \frac{1}{3}$. With respect to the L^2-norm this optimal behavior however is achieved on levels $k \geq 5$ only for the conforming approach whereas with Nitsche's method (employing the minimal regularization parameter) the optimal rates are obtained on levels $k \geq 2$ already, compare also Figures 7.4 and 7.5.

Note however that these results only imply that we may obtain a smaller absolute error;

Table 7.12. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.4 with Dirichlet boundary conditions on Γ_D of (7.7) implemented via Nitsche's method with large regularization.*

J	dof	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	9	3	9.523_{-1}	–	9.884_{-1}	–	9.918_{-1}	–
2	36	12	5.277_{-1}	0.43	6.734_{-1}	0.28	7.078_{-1}	0.24
3	144	48	3.287_{-1}	0.34	3.567_{-1}	0.46	4.563_{-1}	0.32
4	576	192	2.060_{-1}	0.34	1.615_{-1}	0.57	2.820_{-1}	0.35
5	2304	768	1.294_{-1}	0.34	6.817_{-2}	0.62	1.747_{-1}	0.35
6	9216	3072	8.135_{-2}	0.33	2.787_{-2}	0.65	1.091_{-1}	0.34
7	36864	12288	5.123_{-2}	0.33	1.122_{-2}	0.66	6.840_{-2}	0.34
8	147456	49152	3.224_{-2}	0.33	4.490_{-3}	0.66	4.298_{-2}	0.34
9	589824	196608	2.032_{-2}	0.33	1.789_{-3}	0.66	2.714_{-2}	0.33
10	2359296	786432	1.271_{-2}	0.34	7.119_{-4}	0.66	1.717_{-2}	0.33
11	9437184	3145728	7.991_{-3}	0.34	2.830_{-4}	0.67	1.069_{-2}	0.34
12	37748736	12582912	5.062_{-3}	0.33	1.124_{-4}	0.67	6.772_{-3}	0.33

Table 7.13. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.5 with the algebraic conforming boundary approach of §4.2.2 for the implementation of essential boundary conditions.*

J	dof_K	dof_I	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
5	1601	490	697	4.885_{-1}	0.09	2.742_{-1}	0.17	5.267_{-1}	0.08
6	7063	926	2663	3.236_{-1}	0.31	1.131_{-1}	0.66	3.372_{-1}	0.33
7	29324	1714	10346	2.113_{-1}	0.31	4.998_{-2}	0.60	2.211_{-1}	0.31
8	119016	3282	40766	1.364_{-1}	0.32	1.966_{-2}	0.68	1.345_{-1}	0.36
9	479181	6279	161820	8.755_{-2}	0.32	7.164_{-3}	0.73	7.872_{-2}	0.39
10	1921984	12203	644729	5.480_{-2}	0.34	2.587_{-3}	0.74	4.579_{-2}	0.39
11	7697471	24256	2573909	3.273_{-2}	0.37	8.616_{-4}	0.79	2.552_{-2}	0.42

i.e., the asymptotic convergence behavior is reached earlier, with Nitsche's method compared with the conforming boundary treatment. But, this is only the case if we employ the optimal regularization parameter β in Nitsche's method. Recall that our conforming construction can be interpreted as the limit case when the regularization parameter $\beta \to \infty$ in Nitsche's method. To confirm this interpretation numerically we also give the results obtained by Nitsche's method using a very large regularization parameter β in Table 7.12. Observe that now the measured errors and convergence rates agree almost perfectly with those of Table 7.11, see Figures 7.4 and 7.5.

Example 7.5. As the final experimental setup we consider a slight generalization of the L-shaped domain problem, see Figure 7.6. The computational domain has a re-entrant corner with an interior angle $> \frac{3\pi}{2}$ and an elliptical hole. Note that the boundary segments running into the re-entrant corner are intentionally not aligned with the coordinate axes. Here, we employ homogeneous Dirichlet boundary conditions and a constant right-hand side $f = 10$. The conforming approach will eliminate all local approximation spaces V_i which overlap a curved boundary segment to construct the conforming subspace $V_K^{\text{PU}} \subset V^{\text{PU}}$, compare Figure 7.6.

7.1.2 Adaptive Refinement

In the following we now consider the adaptive refinement of our PPUM space V^{PU} in two and three space dimensions based on the error estimator η (5.56) of §5.4.1. To assess the performance of our adaptation approach, compare §5.4, we consider the estimated relative error in the energy-norm

$$e_A^* := \frac{\epsilon_q}{\sqrt{a(u,u)}} = \sqrt{\frac{\sum_{i=1}^N \epsilon_{i,q}^2}{a(u,u)}}, \tag{7.9}$$

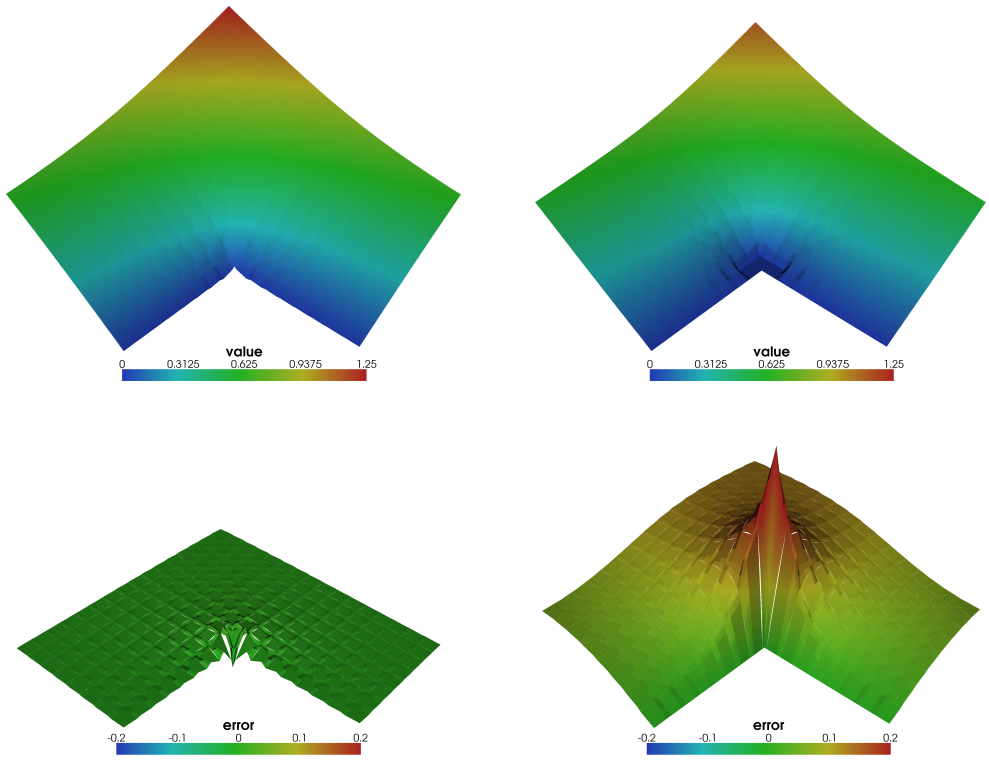


Figure 7.5. *Surface plots of the computed solution on level $k = 4$ (top row) and the respective error (bottom row) obtained by Nitsche's method with minimal regularization (left) and conforming subspaces (right).*

the respective convergence rate ρ_A^*, and the efficiency index

$$\kappa_A^* := \frac{\eta}{\sqrt{a(u - u^{\mathrm{PU}}, u - u^{\mathrm{PU}})}}. \tag{7.10}$$

Note that the bilinear form in Nitsche's method changes from level to level due to the regularization parameter and (7.9) and (7.10) actually employ this level-dependent bilinear form when applied to the results of a non-conforming discretization.

In all experiments we use the parameters $q = 2$, $\sigma_b = 0.8$, $\sigma_{\mathtt{null}} = 1$, $\sigma_{\mathtt{h}} = 3$, and $\sigma_{\mathtt{p}} = 0.6$ in the computation of the error estimator and the refinement scheme, see §5.4.1 and §5.4.2.

Example 7.6. In our first experiment we consider the reference situation of Example 7.4 on the L-shaped domain with the analytical solution (7.8) and realize the Dirichlet boundary conditions on the boundary segments running into the re-entrant corner via Nitsche's method. Here, we use our hp-adaptive refinement scheme to construct a sequence of PPUM spaces V_k^{PU} with $k = 0, \dots, J$ based on the local error estimators η_i of (5.56).[1]

In the FEM it is well-known [122] that the best attainable convergence behavior with hp-refinement for the considered problem class is characterized by the error bound

$$\|u - u^{\mathrm{FE}}\|_{H^1(\Omega)} \leq C \exp(-b \, \mathrm{dof}^{1/3}). \tag{7.11}$$

[1] Note that within the non-conforming approach it is crucial to employ appropriate regularization parameters for the local subdomain problems, see [128] for details.

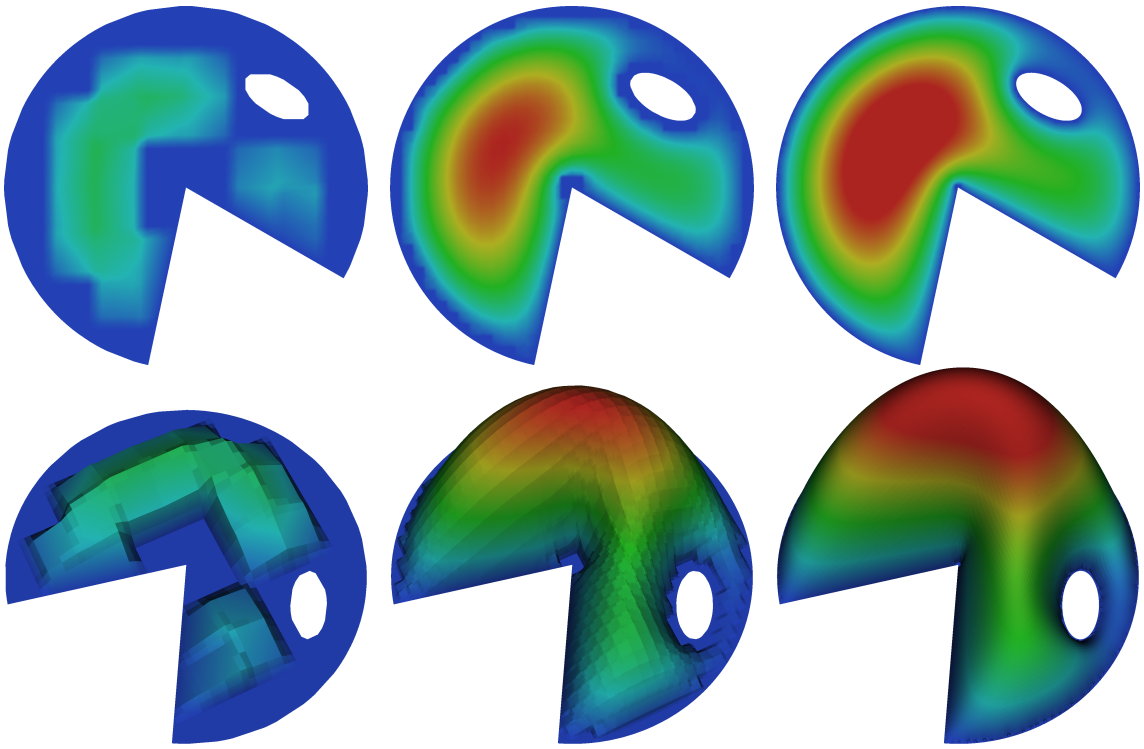


Figure 7.6. *Contour plots of the computed solution (top row) on levels $k = 3, 5, 7$ (left to right) and the respective surface plots (bottom row, rotated). All local approximation spaces intersecting a curved boundary are used for the boundary value approximation.*

We anticipate a similar behavior of our hp-adaptive PPUM. Therefore, we plot the measured relative errors and in the L^∞-norm, the L^2-norm, the H^1-norm, and the estimated relative error in the energy-norm against $\text{dof}^{1/3}$ on a logarithmic scale in Figure 7.7, see also Figures 7.8 and 7.9. The validity of an error bound of type (7.11) for the PPUM can be inferred from the displayed results. Up to level $k = 18$ we essentially find straight lines in Figure 7.7. The error reduction however decays slightly in further refinement steps. We attribute this loss in performance to numerical integration errors. Observe from Figure 7.8 that also the efficiency index of our error estimator falls below 1 on levels $k > 18$. We can also observe that on levels $k > 18$ our PPUM space V^{PU} employs polynomials of maximal degree $p \geq 7$. Thus the integrands arising in the computation of the local approximations to the error are of order 18 already which cannot be computed exactly with our sparse grid integration rule. Thus, we assume that on patches $\omega_{i,k}$ with very large polynomial degree $p_{i,k}$ the reliability of our error estimator $\eta_{i,k}$ is compromised due to integration errors. Therefore, the refinement is no longer steered appropriately and yields sub-optimal results for $k > 18$ only.

Let us now consider the hp-adaptive refinement of a PPUM space in the conforming treatment of our model problem. From the observations of the previous section we assume that the conforming approach yields slightly larger errors than the non-conforming approach since the conforming splitting at the re-entrant corner uses all polynomials at the re-entrant corner for the approximation of the (vanishing) boundary condition such that the approximation benefits only indirectly from the higher polynomial degree via a correction of the right-hand side. The measured results are depicted in Figure 7.10. From these plots we find not only a larger error

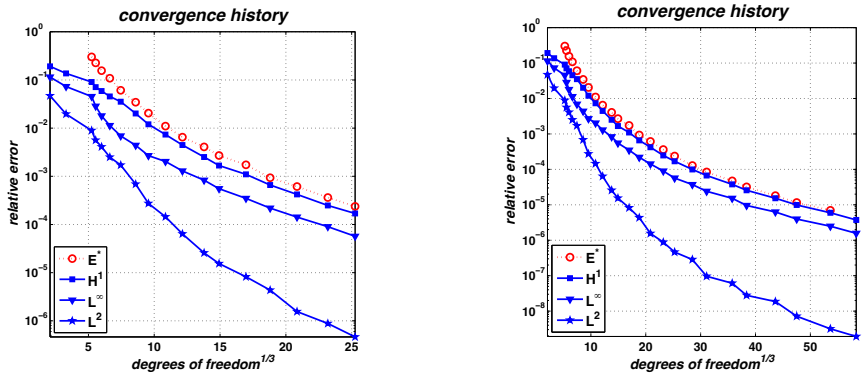


Figure 7.7. *Convergence history of the measured relative errors e (7.3) in the L^∞-norm, the L^2-norm, and the H^1-norm for Example 7.6 using Nitsche's method. The dotted red line with circular markers gives the estimated error (5.56) (denoted by E^* in the legend) used to steer the refinement process. On the left are the results up to level $J = 18$, on the right up to level 26.*

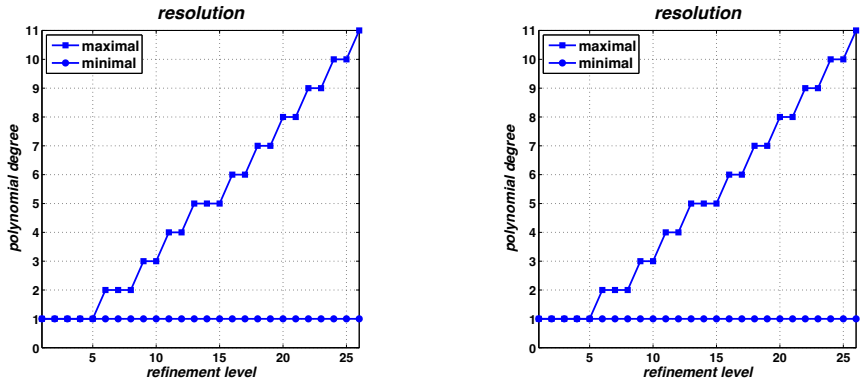


Figure 7.8. *Efficiency index κ^* (left) and range of polynomial degrees (right) for Example 7.6 using Nitsche's method.*

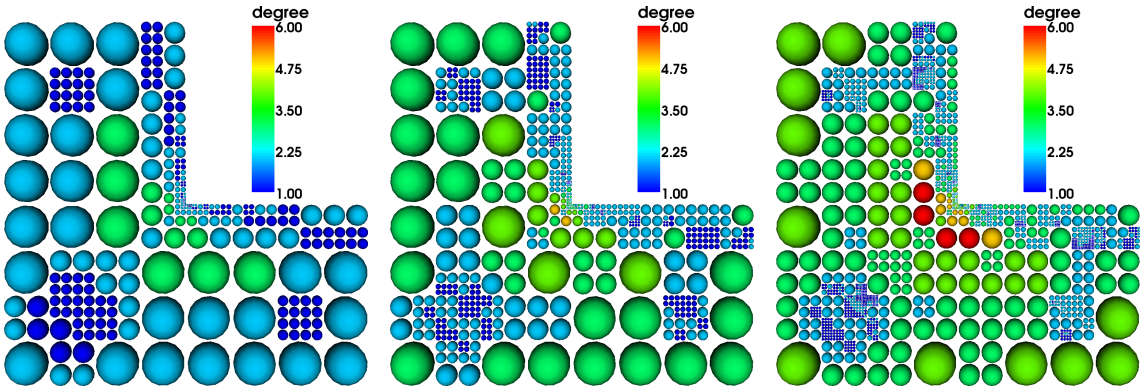


Figure 7.9. *Distribution of the polynomial degrees for levels $k = 10, 13, 16$ (left to right) depicted on the cell centers of the respective tree decomposition for Example 7.6 using Nitsche's method. The size of the particle at the cell center indicates the size of the respective cover patch ω_i, the color of the particle indicates the employed polynomial degree p_i.*

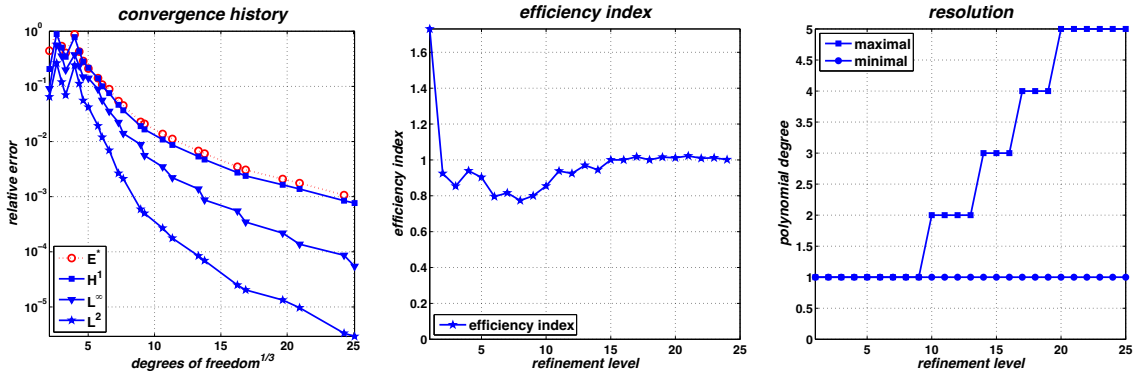


Figure 7.10. *Convergence history (left) of the measured relative errors e (7.3) in the L^∞-norm, the L^2-norm, and the H^1-norm for Example 7.6 using a conforming boundary treatment and the efficiency index κ^* (center) and range of polynomial degrees (right) resulting from an hp-adaptive refinement.*

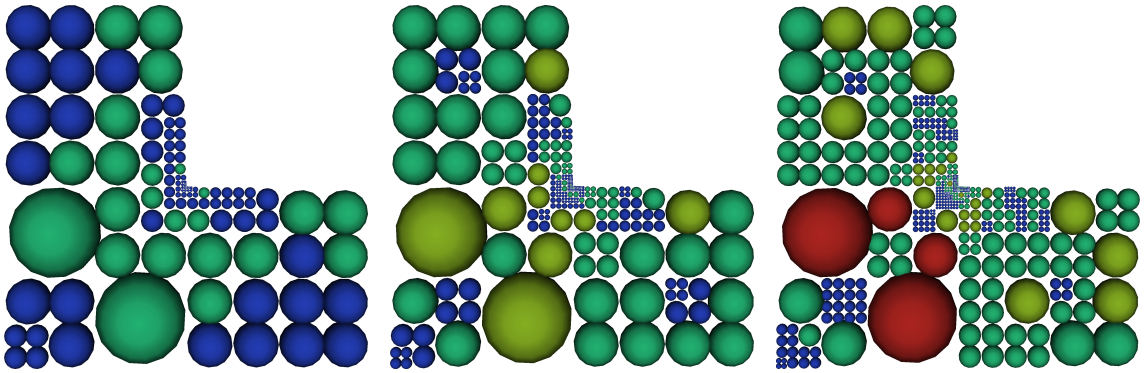


Figure 7.11. *Distribution of the polynomial degrees for levels $k = 10, 13, 16$ (left to right) depicted on the cell centers of the respective tree decomposition for Example 7.6 using a conforming discretization. The size of the particle at the cell center indicates the size of the respective cover patch ω_i, the color of the particle indicates the employed polynomial degree p_i.*

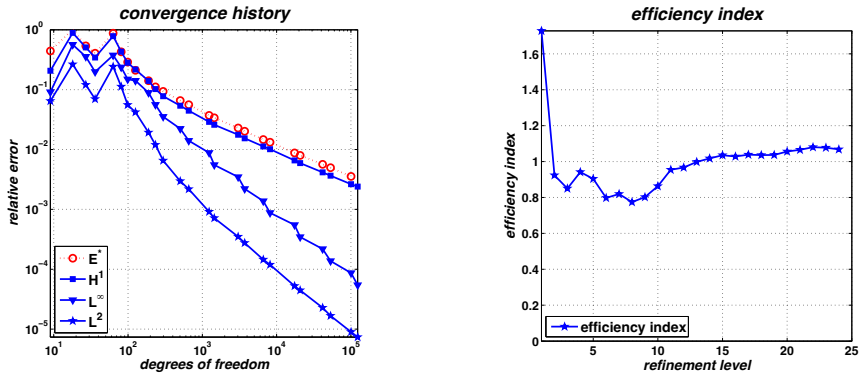


Figure 7.12. *Convergence history (left) of the measured relative errors e (7.3) in the L^∞-norm, the L^2-norm, and the H^1-norm for Example 7.6 using a conforming boundary treatment and the efficiency index κ^* (right) resulting from an h-adaptive refinement.*

Table 7.14. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.6 with h-adaptive refinement and conforming boundary treatment.*

J	dof_K	dof_I	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
1	0	9	3	9.123_{-2}	–	6.457_{-2}	–	2.069_{-1}	–
2	2	16	6	5.650_{-1}	−2.63	2.646_{-1}	−2.03	8.878_{-1}	−2.10
3	11	16	9	3.560_{-1}	1.14	1.200_{-1}	1.95	5.080_{-1}	1.38
4	20	16	12	1.986_{-1}	2.03	6.975_{-2}	1.89	3.440_{-1}	1.35
5	31	32	21	3.755_{-1}	−1.14	2.428_{-1}	−2.23	7.824_{-1}	−1.47
6	45	36	27	2.366_{-1}	1.84	1.125_{-1}	3.06	4.304_{-1}	2.38
7	59	40	33	1.489_{-1}	2.31	5.560_{-2}	3.51	2.814_{-1}	2.12
8	78	48	42	1.413_{-1}	0.22	4.201_{-2}	1.16	2.170_{-1}	1.08
9	121	68	63	8.890_{-2}	1.14	1.921_{-2}	1.93	1.399_{-1}	1.08
10	158	76	78	5.601_{-2}	2.16	1.197_{-2}	2.21	1.023_{-1}	1.47
11	209	88	99	3.528_{-2}	1.94	6.540_{-3}	2.54	7.768_{-2}	1.16
12	388	116	168	2.224_{-2}	0.87	2.966_{-3}	1.49	5.392_{-2}	0.69
13	521	136	219	1.401_{-2}	1.74	2.177_{-3}	1.17	4.452_{-2}	0.72
14	1025	208	411	8.826_{-3}	0.73	9.138_{-4}	1.38	2.896_{-2}	0.68
15	1237	212	483	5.523_{-3}	2.90	7.156_{-4}	1.51	2.592_{-2}	0.69
16	2713	320	1011	3.479_{-3}	0.63	3.498_{-4}	0.97	1.763_{-2}	0.52
17	3373	344	1239	2.206_{-3}	2.24	2.751_{-4}	1.18	1.540_{-2}	0.67
18	6147	468	2205	1.378_{-3}	0.82	1.449_{-4}	1.11	1.124_{-2}	0.55
19	7626	528	2718	8.743_{-4}	2.18	1.191_{-4}	0.94	1.013_{-2}	0.50
20	16613	820	5811	5.495_{-4}	0.61	5.245_{-5}	1.08	6.580_{-3}	0.57
21	19877	868	6915	3.462_{-4}	2.66	4.454_{-5}	0.94	5.929_{-3}	0.60
22	40344	1272	13872	2.181_{-4}	0.66	2.275_{-5}	0.97	4.154_{-3}	0.51
23	51782	1408	17730	1.374_{-4}	1.88	1.670_{-5}	1.26	3.659_{-3}	0.52
24	98691	1884	33525	8.654_{-5}	0.73	8.957_{-6}	0.98	2.634_{-3}	0.52
25	121072	2120	41064	5.451_{-5}	2.28	7.361_{-6}	0.97	2.403_{-3}	0.45

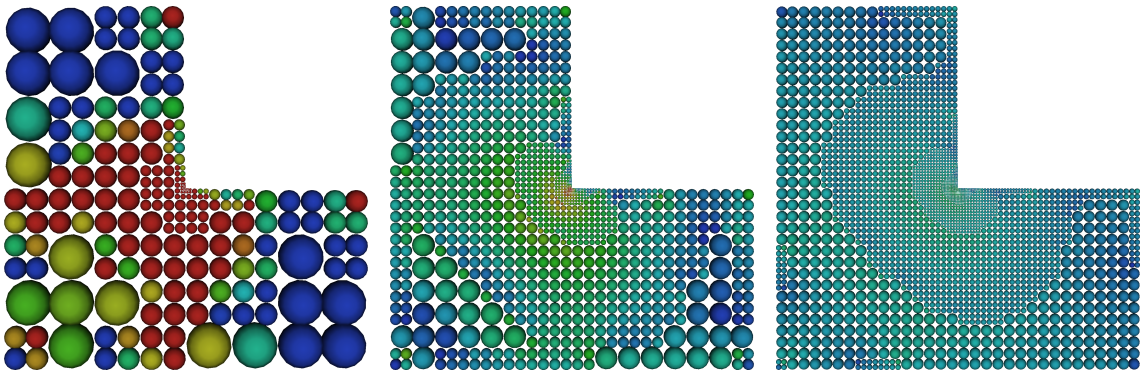


Figure 7.13. *Distribution of the error for levels $k = 13, 16, 19$ (left to right) depicted on the cell centers of the respective tree decomposition for Example 7.6 using an h-adaptive conforming discretization. The size of the particle at the cell center indicates the size of the respective cover patch ω_i.*

compared to the non-conforming approach but also the convergence is slightly slower. Yet, the quality of the error estimator is high since we find a very stable efficiency index of about 1. One of the reasons for this loss in performance is the different error distribution in the conforming approach, compare Figure 7.5. Here, the error seems to be rougher such that less p-refinement is carried out in the conforming approach, compare Figures 7.10 and 7.8, and the Figures 7.11 and 7.9. Thus, we assume that the optimal convergence behavior (7.11) is reached later than in the non-conforming approach. Moreover, it is not clear that the refinement parameters given above which were identified for the non-conforming approach are optimal for the conforming approach

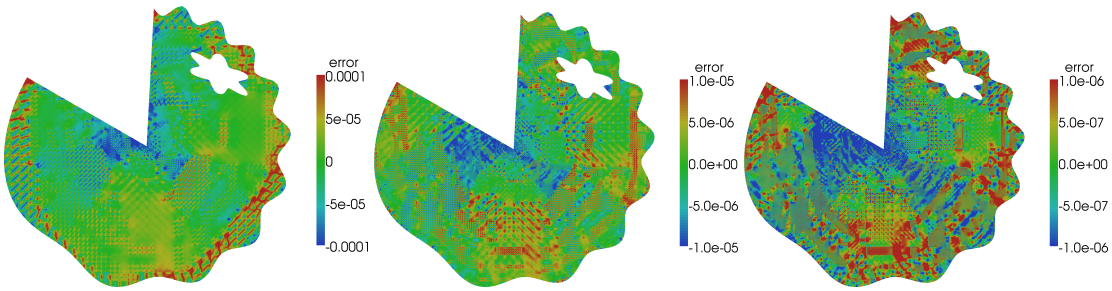


Figure 7.14. *Distribution of the error for levels $k = 15, 20, 25$ (left to right) for Example 7.6 using an hp-adaptive conforming discretization.*

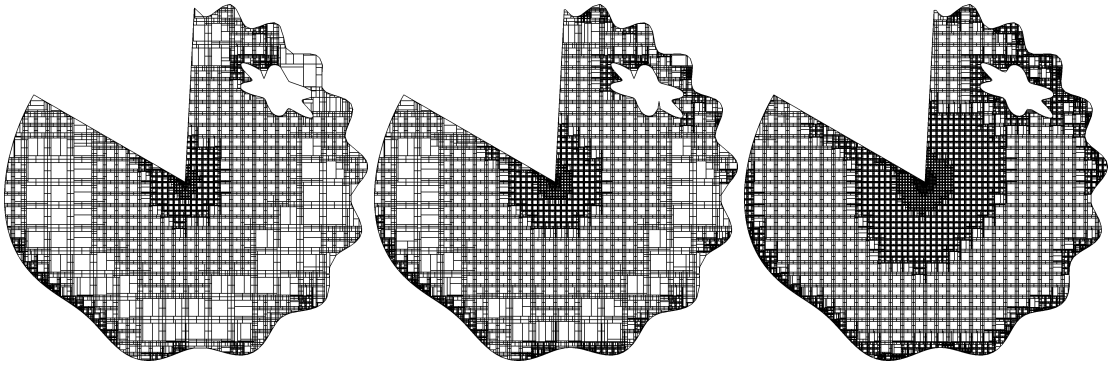


Figure 7.15. *Wireframe representation of the integration cells constructed for the h-adaptive conforming discretization of Example 7.6.*

as well.

An h-adaptive refinement however should yield its optimal convergence behavior much faster since the quality of the error estimator is the only influence on the refinement and convergence behavior. From the plots depicted in Figure 7.12 and the convergence rates ρ given in Table 7.14 the optimal convergence of the h-adaptive PPUM can in fact be observed. We find very stable rates around $\rho_{L^2} = 1$ and $\rho_{H^1} = \frac{1}{2}$ as expected. In Figure 7.13 we give some snapshots of the pointwise error at the cell centers of our cover patches on the levels $k = 13, 16, 19$. From these plots we can clearly observe the fast reduction of the error and the strong refinement towards the re-entrant corner.

Finally, we consider our model problem (7.1) with $c = 0$ on a non-convex domain with interior angle $\beta = \frac{3.3\pi}{2}$, a non-convex hole, and an oscillatory outer boundary, see Figures 7.14 and 7.15. We consider Dirichlet boundary conditions and choose the data f and g such that the analytical solution is given by

$$u(r, \theta) = r^{\pi/\beta} \sin\left(\frac{\pi\theta}{\beta}\right)$$

with respect to an appropriately rotated polar coordinate system. Since the boundary of the domain is approximated on each level independently we cannot expect a very smooth convergence behavior in this example. Moreover, the boundary of the domain is rather complicated and our conforming approach will use all local approximation spaces at the curved boundaries for the approximation of the boundary data. From the results of the previous experiment we infer that the hp-adaptive refinement strategy will most probably not yield exponential convergence for the

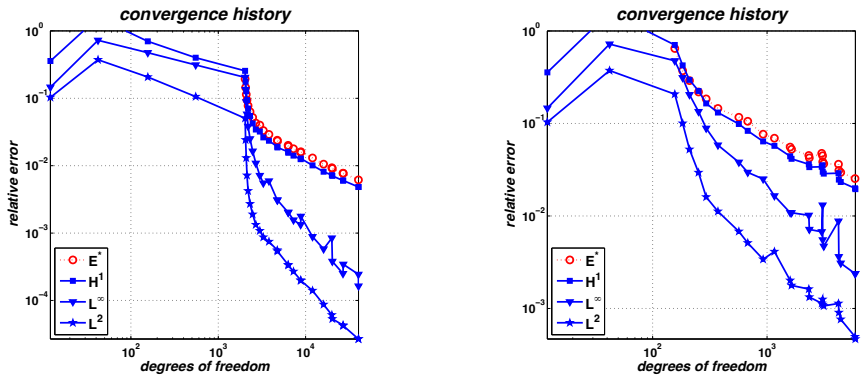


Figure 7.16. *Convergence history of the measured relative errors e (7.3) in the L^∞-norm, the L^2-norm, and the H^1-norm for Example 7.6 using a conforming boundary treatment and hp-adaptive refinement (left), or an h-adaptive refinement (right) only.*

Table 7.15. *Estimated errors e^* and respective convergence rates ρ^* for Example 7.7.*

J	dof	N	$[p]$	L_C	$e^*_{L^2}$	$\rho^*_{L^2}$	$e^*_{H^1}$	$\rho^*_{H^1}$	e^*_A	ρ^*_A
3	1792	448	[1,1]	0	6.673_{-3}	0.55	2.315_{-1}	0.24	3.475_{-1}	0.30
4	14336	3584	[1,1]	0	1.881_{-3}	0.61	1.280_{-1}	0.28	1.721_{-1}	0.34
5	58996	14749	[1,1]	1	1.174_{-3}	0.33	9.464_{-2}	0.21	1.072_{-1}	0.33
6	131366	28091	[1,2]	1	3.085_{-4}	1.67	4.446_{-2}	0.94	5.292_{-2}	0.88
7	276180	47166	[1,3]	2	8.861_{-5}	1.68	2.059_{-2}	1.04	2.390_{-2}	1.07
8	659599	109291	[1,4]	2	7.210_{-5}	0.24	1.622_{-2}	0.27	2.038_{-2}	0.18
9	1468658	220969	[1,5]	3	2.430_{-5}	1.36	9.816_{-3}	0.63	1.324_{-2}	0.54

conforming approach. We assume that an insufficient amount of p-refinement is carried out so that in the worst case our hp-refinement actually behaves like an h-adaptive PPUM. This assertion is confirmed by the measured results given in Figure 7.16 (note that we employ a small number of uniform h-refinements before the adaptation process starts). Our hp-refinement scheme here essentially yields almost the same refinement pattern as the h-refinement strategy with only a minimal number of p-refinements (up to level 26 the maximal polynomial degree is $p = 2$).

Example 7.7. Next we consider the three-dimensional analogue of the L-shaped domain with homogeneous Dirichlet boundary conditions realized by Nitsche's method and a constant right-hand side $f = 1$. In this example we measure absolute errors only and we estimate the errors in the L^2-norm and the H^1-norm by $e^*_{L^2}$ and $e^*_{H^1}$ with the help of the local approximations to the error $e^{\mathrm{PU}}_{i,q}$. The estimated errors e^* and the respective convergence rates ρ^* are given in Table 7.15. We can observe an increase in the rates ρ^* for larger refinement levels J indicating an exponential convergence as in the previous example.

Example 7.8. In our last example of this section, we consider the adaptive refinement of an enriched PPUM space. To this end, we apply our PPUM with a minimal amount of enrichment functions to a reference problem from linear elastic fracture mechanics

$$\begin{aligned} -\,\mathbf{div}\,\sigma(u) &= f && \text{in } \Omega = (-1,1)^2, \\ \sigma(u)\cdot n &= g_N && \text{on } \Gamma_N \subset \partial\Omega \cup C, \\ u &= g_D && \text{on } \Gamma_D = \partial\Omega \setminus \Gamma_N, \end{aligned} \tag{7.12}$$

with material parameters $E = 1$ and $\nu = 0.3$. The Dirichlet boundary is given by $\Gamma_D := \{(x,y) \in \partial\Omega \,|\, y = -1\}$ and we use $g_D = (0,0)$. On the upper boundary $\{(x,y) \in \partial\Omega \,|\, y = 1\} \subset \Gamma_N$ we use

Table 7.16. *Estimated errors e^* and respective convergence rates ρ^* for Example 7.8.*

J	dof	N	$[p]$	$L_{\max}$	$e^*_{L^2}$	$\rho^*_{L^2}$	$e^*_{H^1}$	$\rho^*_{H^1}$	e^*_A	ρ^*_A
3	472	64	[1,1]	0	1.981_{-2}	0.89	4.530_{-1}	0.23	5.164_{-1}	0.23
4	1132	166	[1,1]	1	1.159_{-2}	0.61	3.805_{-1}	0.20	4.120_{-1}	0.26
5	2464	346	[1,2]	2	4.782_{-3}	1.14	2.777_{-1}	0.41	2.852_{-1}	0.47
6	4600	568	[1,3]	2	1.851_{-3}	1.52	1.846_{-1}	0.65	1.860_{-1}	0.68
7	7068	778	[1,3]	2	8.175_{-4}	1.90	1.237_{-1}	0.93	1.238_{-1}	0.95
8	10144	1066	[1,3]	3	4.676_{-4}	1.55	8.485_{-2}	1.04	8.494_{-2}	1.04
9	13364	1330	[1,4]	3	2.914_{-4}	1.72	6.044_{-2}	1.23	6.020_{-2}	1.25
10	17800	1708	[1,4]	3	1.879_{-4}	1.53	4.332_{-2}	1.16	4.310_{-2}	1.17
11	23380	2140	[1,4]	4	1.036_{-4}	2.18	3.014_{-2}	1.33	3.001_{-2}	1.33
12	29716	2608	[1,4]	4	6.432_{-5}	1.99	2.138_{-2}	1.43	2.124_{-2}	1.44
13	37412	3160	[1,5]	4	3.680_{-5}	2.42	1.518_{-2}	1.49	1.504_{-2}	1.50
14	46536	3922	[1,5]	4	2.431_{-5}	1.90	1.061_{-2}	1.64	1.055_{-2}	1.63
15	56476	4582	[1,6]	5	1.605_{-5}	2.14	7.583_{-3}	1.73	7.525_{-3}	1.74
16	69256	5542	[1,6]	5	1.037_{-5}	2.14	5.405_{-3}	1.66	5.357_{-3}	1.67
17	87896	6856	[1,6]	5	5.370_{-6}	2.76	3.709_{-3}	1.58	3.682_{-3}	1.57
18	106188	7726	[1,6]	5	3.421_{-6}	2.39	2.640_{-3}	1.80	2.616_{-3}	1.81
19	129972	9424	[1,7]	6	2.065_{-6}	2.50	1.866_{-3}	1.72	1.849_{-3}	1.72
20	160800	11194	[1,7]	6	1.396_{-6}	1.84	1.281_{-3}	1.77	1.272_{-3}	1.76
21	193736	12826	[1,7]	6	9.238_{-7}	2.22	9.190_{-4}	1.78	9.114_{-4}	1.79
22	239784	15850	[1,7]	6	6.079_{-7}	1.96	6.501_{-4}	1.62	6.445_{-4}	1.62
23	310348	19648	[1,8]	7	3.949_{-7}	1.67	4.514_{-4}	1.41	4.478_{-4}	1.41

$g_N = (0,1)$. On the remaining parts of the Neumann boundary Γ_N we use $g_N = (0,0)$ where the crack $C \subset \Gamma_N$ is defined as

$$C := \{(x,y) \in \Omega \,|\, x \in (-0.5, 0.5) \text{ and } y = 0\}.$$

The singular behavior of the solution at the crack tips is modeled by the enrichment functions (5.23) and its discontinuity across the crack line is realized by (5.22). Note that we do not use the hierarchical enrichment approach of §5.3 but employ enrichment only directly at the tips. Thus, the enriched PPUM spaces do not resolve the singular behavior of the solution asymptotically and an adaptive refinement process is necessary to recover the optimal convergence behavior.

First, we consider a non-conforming discretization of (7.12) and adaptive hp-refinement. The estimated errors e^* and the respective convergence rates ρ^* are collected in Table 7.16. Here, we give estimated absolute errors only and use the strain energy seminorm $\|\frac{1}{2}(\nabla u + \nabla^T u)\|_{L^2(\Omega)}$ for the definition of $e^*_{H^1}$. Furthermore, we also indicate the maximal level difference $L_{\max}$ attained on the respective refinement level. From the displayed numbers we can clearly observe the anticipated increase in the algebraic convergence rates which indicates an exponential convergence.

We give snapshots of the distribution of the employed polynomial degrees on the (scaled) deformed configuration in Figure 7.17 and for the distribution of the von Mises stress in Figure 7.18. From these plots, we clearly see the hp-adaptive resolution of the singularities near the crack tips as well as the additional singularities in the Dirichlet-Neumann corners. Note also that across the crack line we obtain a (pure) p-refinement in regions where the displacement field is locally (above/below the crack line) smooth.

7.1.3 Hierarchical Enrichment

Let us now consider the refinement of our PPUM spaces with hierarchical enrichment. Recall that this approach may recover the optimal convergence behavior of the uniform h-version also for singular solutions — provided that the employed enrichment functions resolve the singularities of the solution. Thus, we may also interpret the enrichment approach as an solution-adapted refinement technique, yet it is an algebraic adaptation rather than a geometric adaptation.

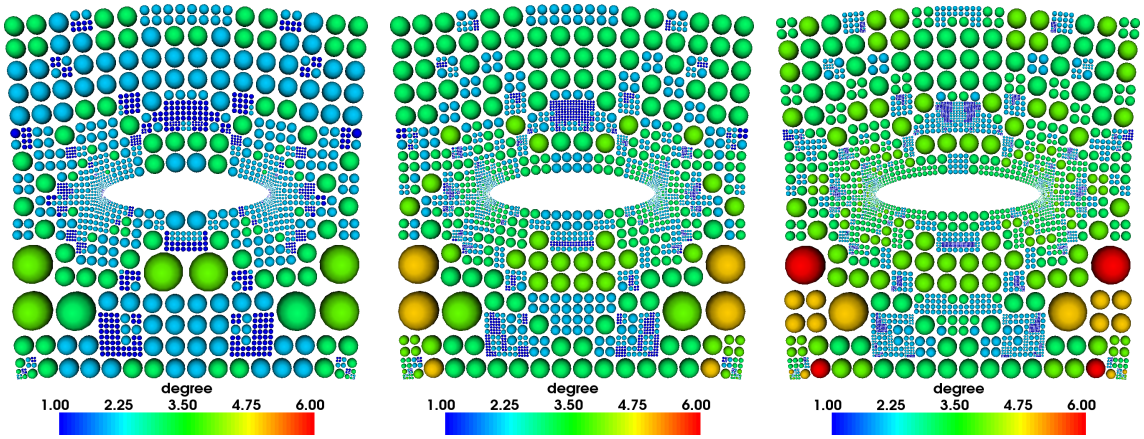


Figure 7.17. *Distribution of the polynomial degrees for levels $k = 10, 13, 16$ (left to right) depicted on the cell centers of the respective (deformed) tree decomposition for Example 7.8. The size of the particle at the cell center indicates the size of the respective support patch ω_i, the color of the particle indicates the employed polynomial degree p_i.*

Example 7.9. In the first example we consider the model problem (7.1) with $c = 0$ and Dirichlet boundary conditions on the domain $\Omega = (-1, 1)^2$. Here we choose the data f and g_D such that the analytical solution is given by

$$u(x_1, x_2) = \sqrt{r}(\sin(\frac{\theta}{2}) + \cos(\frac{\theta}{2}))(1 + \sin(\theta)) + (x_1^2 - 1) + (x_2^2 - 1) + 1 \tag{7.13}$$

where $(r(x), \theta(x))$ denote polar coordinates, see Figure 7.19. This solution is obviously designed to mimic the behavior of a solution to a crack problem with $C = \{(x_1, x_2) \,|\, x_1 \in (-1, 0) \text{ and } x_2 = 0\}$ so that the enrichment spaces (5.22) and (5.23) are optimal for the approximation of (7.13).

Here, we employ three initial refinement steps with minimal enrichment and employ the hierarchical enrichment with local preconditioning on all later refinement levels $k > 3$, as smooth local approximation spaces we use linear polynomials. Thus, we consider the enrichment zone

$$E_{\text{tip}} := (-0.25, 0.25)^2 \subset \Omega = (-1, 1)^2 \tag{7.14}$$

on all levels $k = 0, \dots, J$ and we expect to find a linear convergence behavior with respect to the H^1-norm globally, i.e. $\rho_{H^1} = \frac{1}{2}$. According to Theorem 5.1 we can obtain a better convergence behavior within the enrichment zone E_{tip}. To confirm this result we measure the errors (7.3) and respective convergence rates (7.4) not only globally on Ω but also with respect to the subdomains

$$E_1 := E_{\text{tip}} = \Big(-\frac{1}{4}, \frac{1}{4}\Big)^2, \quad E_2 := \Big(-\frac{1}{8}, \frac{1}{8}\Big)^2, \quad E_2 := \Big(-\frac{1}{16}, \frac{1}{16}\Big)^2. \tag{7.15}$$

Note that in this example we use a very aggressive enrichment strategy since we employ the H^1-norm for the construction of the preconditioner $S_{\mathcal{E}\setminus\mathcal{P}}$ where we set $\epsilon = 10^{-12}$.

The measured values are given in Table 7.17, see also Figures 7.20 and 7.21. We can clearly see a perfect agreement with the theoretical convergence behavior. We find the rates $\rho_{L^2} = 1$ and $\rho_{H^1} = \frac{1}{2}$ for the global convergence on Ω. Within the enrichment zone E_{tip} we obtain convergence rates of $1.3 \geq \rho_{H^1} \geq 0.75$ and $1.78 \geq \rho_{L^2} \geq 1$. Thus, the super-convergence established in Theorem 5.1 is practically observed and (at least for the considered enrichment spaces) substantial. We attain an almost quadratic convergence behavior in the H^1-norm near the singularity of the solution.

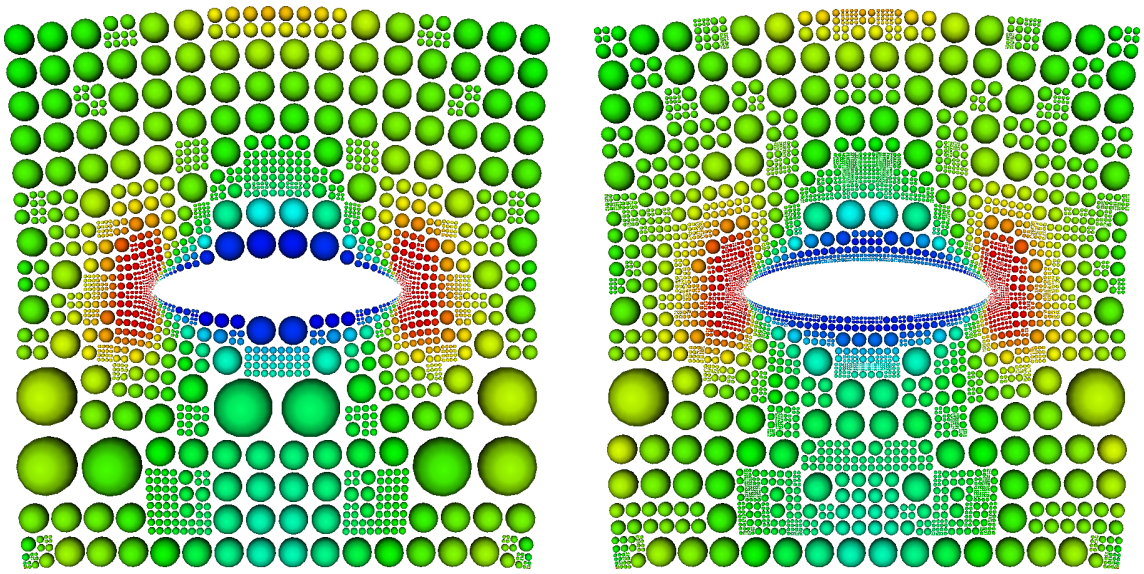


Figure 7.18. *Distribution of the von Mises stress for Example 7.8 on the cell centers of the respective deformed tree decomposition for levels $J = 10$ (left) and $J = 19$ (right). The size of the particle at the cell center indicates the size of the respective support patch ω_i, the color of the particle indicates the von Mises stress at the cell center.*

Example 7.10. In our second example we consider the linear elastic fracture mechanics model problem

$$\begin{aligned} -\operatorname{\mathbf{div}} \sigma(u) &= f && \text{in } \Omega \subset \mathbb{R}^D, \\ \sigma(u) \cdot n &= g_N && \text{on } \Gamma_N, \\ u \cdot n &= g_D && \text{on } \Gamma_D = \partial\Omega \setminus \Gamma_N, \end{aligned} \tag{7.16}$$

on the domain $\Omega := (-1,1)^2$. The considered material parameters are $E = 1000$ and $\nu = 0.3$. The Dirichlet boundary is defined as

$$\Gamma_D := \{(x,y) \in \partial\Omega \,|\, y = -1\}$$

and we assume homogeneous Dirichlet data $g_D = 0$ which we implement by Nitsche's method. The Neumann boundary $\Gamma_N := \partial\Omega \setminus \Gamma_D \cup C$ contains the traction-free crack C of Example 7.9. We apply inhomogeneous Neumann boundary conditions $g_N = (1,0)$ on the segment $\{(x,y) \in \Gamma_N \,|\, y = 1\} \subset \Gamma_N$ only. Again we use the same enrichment zone E_{tip} given in (7.14) and the enrichment spaces (5.22), (5.23), and (5.26).

Note that we employ a reference solution $u_{\text{ref}}^{\text{PU}} \in V_{\text{ref}}^{\text{PU}}$ in this example to assess the performance of our PPUM. This reference solution was computed by the enriched PPUM using a global hp-refinement of the PPUM space V_J^{PU} the finest level $J = 9$; i.e., the reference space $V_{\text{ref}}^{\text{PU}}$ employs roughly 10^6 patches, the local polynomial spaces $\mathcal{P}^2$, the enrichment spaces (5.22), (5.23), and (5.26), and has roughly 10^7 degrees of freedom dof_{ref}. Thus, the relative errors (7.3) and respective convergence rates (7.4) given in Table 7.18, see also Figures 7.22 and 7.24, refer to this reference solution $u_{\text{ref}}^{\text{PU}}$. In Figure 7.10 we give a contour plot of the von Mises stress of the reference solution on the deformed domain. A contour plot with respect to the enrichment zone E_{tip} of the dimension $d_{i,J_{\text{ref}}}^{\mathcal{D}} = \dim(\mathcal{D}_{i,J_{\text{ref}}})$ of the local (scalar) pure enrichment spaces employed in $V_{\text{ref}}^{\text{PU}}$ is depicted in Figure 7.26. Observe that close to the singularity all original additive enrichment functions are present ($d_{i,J_{\text{ref}}}^{\mathcal{D}} = \dim(W_{\text{tip}}) = 4$). Further away from the singularity we

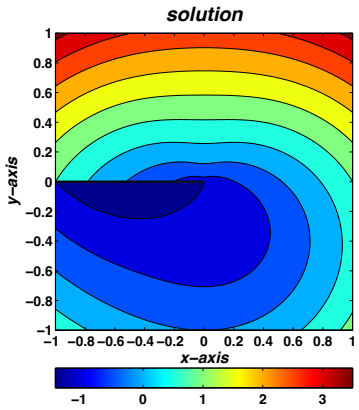


Figure 7.19. *Contour plot of the solution* (7.13).

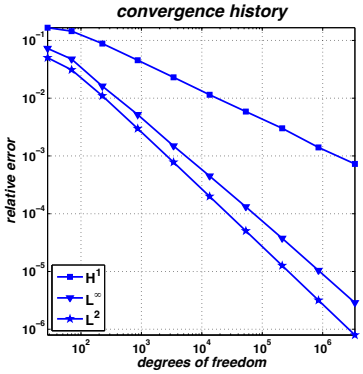


Figure 7.20. *Convergence history of the measured relative errors e* (7.3) *with respect to the complete domain* Ω *in the* L^∞*-norm, the* L^2*-norm, and the* H^1*-norm for Example* 7.9.

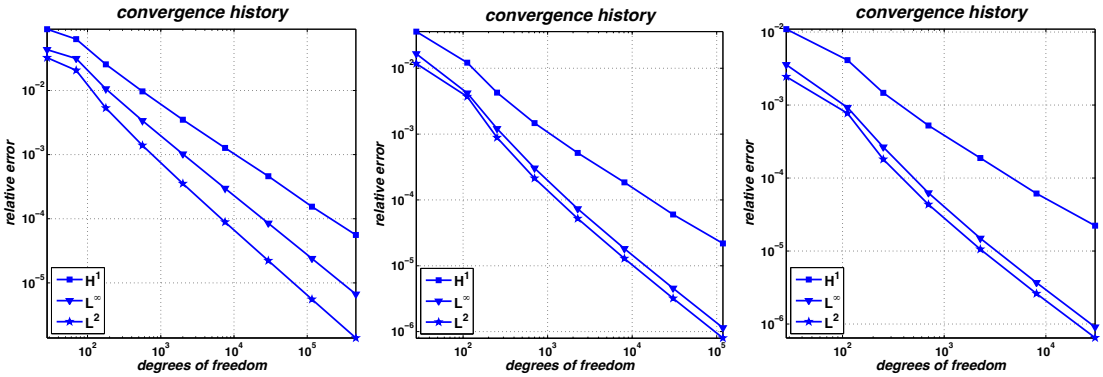


Figure 7.21. *Convergence history of the measured relative errors e* (7.3) *with respect to the subdomains* E_1 *(left),* E_2 *(center), and* E_3 *(right) given in* (7.15) *with respect to the* L^∞*-norm, the* L^2*-norm, and the* H^1*-norm for Example* 7.9.

find $d^{\mathcal{D}}_{i,J_{\text{ref}}} = 3$; i.e. one enrichment function was automatically removed by our preconditioner. On the crack line C we employ both enrichments (5.22) and (5.23); i.e., we use (5.26), so that $c^{\mathcal{E}}_{i,J_{\text{ref}}} = \dim(W_{\text{tip}}) + \dim(H^C_\pm \mathcal{P}^2) = 4 + 6 = 10$. Close to the singularity all these enrichment functions are present and contribute to the approximation since we find $d^{\mathcal{D}}_{i,J_{\text{ref}}} = 10$. Moving away from the singularity along the crack the dimension $d^{\mathcal{D}}_{i,J_{\text{ref}}}$ decreases and several enrichment functions were removed by our preconditioner to obtain a stable basis of $V^{\text{PU}}_{\text{ref}}$.

Note that there are some oscillations in the local dimension $d^{\mathcal{D}}_{i,J_{\text{ref}}}$ due to the interplay of the cut-off parameter ϵ in the construction of the local preconditioner and quadrature errors in the assembly of the local operators. These oscillations however are insubstantial and have no effect on the stability, the approximation properties, or the regularity of the approximation.

Again, we compute the relative errors and corresponding convergence rates with respect to the domain Ω and the subdomains E_1, E_2, and E_3 of (7.15). In this example we expect to find the

Table 7.17. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.9 using Nitsche's method.*

J	dof	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
with respect to Ω								
1	28	4	7.262_{-2}	–	5.011_{-2}	–	1.663_{-1}	–
2	70	16	4.741_{-2}	0.47	3.096_{-2}	0.53	1.449_{-1}	0.15
3	226	64	1.614_{-2}	0.92	1.098_{-2}	0.88	8.826_{-2}	0.42
4	874	256	5.192_{-3}	0.84	2.974_{-3}	0.97	4.544_{-2}	0.49
5	3418	1024	1.488_{-3}	0.92	7.779_{-4}	0.98	2.296_{-2}	0.50
6	13498	4096	4.491_{-4}	0.87	1.990_{-4}	0.99	1.148_{-2}	0.50
7	53626	16384	1.317_{-4}	0.89	5.034_{-5}	1.00	5.864_{-3}	0.49
8	213736	65536	3.748_{-5}	0.91	1.266_{-5}	1.00	3.010_{-3}	0.48
9	853410	262144	1.042_{-5}	0.92	3.173_{-6}	1.00	1.405_{-3}	0.55
10	3410716	1048576	2.878_{-6}	0.93	7.946_{-7}	1.00	7.269_{-4}	0.48
with respect to E_1								
2	28	4	4.329_{-2}	–	3.245_{-2}	–	9.040_{-2}	–
3	70	16	3.145_{-2}	0.35	2.061_{-2}	0.50	6.313_{-2}	0.39
4	184	36	1.058_{-2}	1.13	5.325_{-3}	1.40	2.559_{-2}	0.93
5	580	100	3.385_{-3}	0.99	1.390_{-3}	1.17	9.671_{-3}	0.85
6	2044	324	1.018_{-3}	0.95	3.534_{-4}	1.09	3.495_{-3}	0.81
7	7660	1156	2.974_{-4}	0.93	8.867_{-5}	1.05	1.270_{-3}	0.77
8	29626	4356	8.505_{-5}	0.93	2.217_{-5}	1.02	4.604_{-4}	0.75
9	116532	16900	2.394_{-5}	0.93	5.538_{-6}	1.01	1.541_{-4}	0.80
10	462382	66564	6.645_{-6}	0.93	1.384_{-6}	1.01	5.586_{-5}	0.74
with respect to E_2								
3	28	4	1.659_{-2}	–	1.178_{-2}	–	3.599_{-2}	–
4	118	16	4.200_{-3}	0.95	3.707_{-3}	0.80	1.216_{-2}	0.75
5	264	36	1.209_{-3}	1.55	8.846_{-4}	1.78	4.271_{-3}	1.30
6	724	100	3.040_{-4}	1.37	2.133_{-4}	1.41	1.472_{-3}	1.06
7	2316	324	7.327_{-5}	1.22	5.209_{-5}	1.21	5.182_{-4}	0.90
8	8188	1156	1.818_{-5}	1.10	1.287_{-5}	1.11	1.849_{-4}	0.82
9	30658	4356	4.555_{-6}	1.05	3.200_{-6}	1.05	6.004_{-5}	0.85
10	118588	16900	1.141_{-6}	1.02	7.980_{-7}	1.03	2.187_{-5}	0.75
with respect to E_3								
4	28	4	3.567_{-3}	–	2.447_{-3}	–	1.096_{-2}	–
5	118	16	9.241_{-4}	0.94	7.710_{-4}	0.80	4.124_{-3}	0.68
6	264	36	2.651_{-4}	1.55	1.803_{-4}	1.80	1.471_{-3}	1.28
7	724	100	6.244_{-5}	1.43	4.337_{-5}	1.41	5.247_{-4}	1.02
8	2316	324	1.495_{-5}	1.23	1.056_{-5}	1.21	1.882_{-4}	0.88
9	8188	1156	3.685_{-6}	1.11	2.606_{-6}	1.11	6.133_{-5}	0.89
10	30652	4356	9.081_{-7}	1.06	6.476_{-7}	1.05	2.227_{-5}	0.77

global convergence rates to be close to $\rho_{L^2} = 1$ and $\rho_{H^1} = \frac{1}{2}$ since to solution of (7.16) has two additional but weaker singularities at the corners $(-1,-1)$ and $(1,-1)$. Within the enrichment zone we anticipate to find rates $\rho_{L^2} \geq 1$ and $\rho_{H^1} \geq \frac{1}{2}$ as in Example 7.9.

Table 7.18 summarizes the computed results. From the displayed numbers we can clearly see the anticipated convergence behavior. For the complete domain we find the rates $\rho_{L^2} = 1$ and $\rho_{H^1} = \frac{1}{2}$ and inside the enrichment zone we obtain rates between 1 and 1.5 for the L^2-norm and 0.8 and 1.2 for the H^1-norm. These results correspond very well to those of Example 7.9. Again, we find an almost quadratic convergence with respect to the H^1-norm within the enrichment zone E_{tip}. Observe from Figure 7.23 where we give a contour plot of the error on level $k = 9$ with respect to the reference solution that the error is regular (with the expected jump across the crack line). All singular components (apart from the corner singularities at $(-1,-1)$ and $(1,-1)$ which are not considered by the enrichment) of the solution are resolved. In Figure 7.23 we depict the dimension $d_{i,k}^{\mathcal{D}} = \dim(\mathcal{D}_{i,k})$ of the local (scalar) pure enrichment spaces employed in V_k^{PU} on level $k = 9$. Observe that our preconditioner does not remove any enrichment functions for patches $\omega_{i,k} \cap C = \emptyset$. Only along the crack line we see a decay in the dimension $d_{i,k}^{\mathcal{D}}$ moving away from

Table 7.18. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.10 using Nitsche's method.*

J	dof	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
with respect to Ω								
1	56	4	2.401_{-1}	–	2.028_{-1}	–	2.519_{-1}	–
2	140	16	1.945_{-1}	0.23	1.870_{-1}	0.09	2.161_{-1}	0.17
3	452	64	9.564_{-2}	0.61	9.405_{-2}	0.59	1.131_{-1}	0.55
4	1748	256	2.786_{-2}	0.91	2.761_{-2}	0.91	4.041_{-2}	0.76
5	6836	1024	7.320_{-3}	0.98	7.288_{-3}	0.98	1.608_{-2}	0.68
6	26996	4096	1.853_{-3}	1.00	1.850_{-3}	1.00	7.355_{-3}	0.57
7	107252	16384	4.635_{-4}	1.00	4.640_{-4}	1.00	3.583_{-3}	0.52
8	427472	65536	1.772_{-4}	0.70	1.160_{-4}	1.00	1.783_{-3}	0.50
9	1706820	262144	1.040_{-4}	0.39	2.901_{-5}	1.00	8.575_{-4}	0.53
with respect to E_1								
2	28	4	1.493_{-1}	–	1.448_{-1}	–	1.620_{-1}	–
3	70	16	9.055_{-2}	0.55	7.939_{-2}	0.66	1.112_{-1}	0.41
4	184	36	2.713_{-2}	1.25	2.425_{-2}	1.23	3.867_{-2}	1.09
5	580	100	7.450_{-3}	1.13	6.553_{-3}	1.14	1.249_{-2}	0.98
6	2044	324	1.954_{-3}	1.06	1.694_{-3}	1.07	4.053_{-3}	0.89
7	7660	1156	5.056_{-4}	1.02	4.331_{-4}	1.03	1.350_{-3}	0.83
8	29626	4356	1.307_{-4}	1.00	1.109_{-4}	1.01	4.610_{-4}	0.79
9	116532	16900	3.381_{-5}	0.99	2.854_{-5}	0.99	1.581_{-4}	0.78
with respect to E_2								
3	56	4	7.690_{-2}	–	7.560_{-2}	–	8.106_{-2}	–
4	236	16	2.623_{-2}	0.75	2.281_{-2}	0.83	3.331_{-2}	0.62
5	528	36	6.887_{-3}	1.66	6.297_{-3}	1.60	1.114_{-2}	1.36
6	1448	100	1.752_{-3}	1.36	1.641_{-3}	1.33	3.598_{-3}	1.12
7	4632	324	4.435_{-4}	1.18	4.224_{-4}	1.17	1.194_{-3}	0.95
8	16376	1156	1.255_{-4}	1.00	1.091_{-4}	1.07	4.064_{-4}	0.85
9	61316	4356	4.028_{-5}	0.86	2.840_{-5}	1.02	1.384_{-4}	0.82
with respect to E_3								
4	56	4	2.006_{-2}	–	2.304_{-2}	–	1.947_{-2}	–
5	236	16	7.198_{-3}	0.71	6.343_{-3}	0.90	1.296_{-2}	0.28
6	528	36	1.763_{-3}	1.75	1.675_{-3}	1.65	4.912_{-3}	1.20
7	1448	100	4.770_{-4}	1.30	4.320_{-4}	1.34	1.724_{-3}	1.04
8	4632	324	1.520_{-4}	0.98	1.117_{-4}	1.16	6.045_{-4}	0.90
9	16376	1156	4.877_{-5}	0.90	2.913_{-5}	1.06	2.089_{-4}	0.84

the singularity at $(0,0)$.

We use the contour integral method (CIM) [140] to extract the stress intensity factors (SIF) $\mathsf{sif}_{\mathrm{I}}$ for mode I and $\mathsf{sif}_{\mathrm{II}}$ for mode II. For the computation of the SIFs of the reference solution we use a large number of different extraction domains $\{(x,y) \in \Omega \,|\, \max\{x,y\} < r\}$; i.e., extraction radii r, and compute the mean value of these SIFs (for $10^{-2} \leq r \leq 10^{-1}$) as our reference values

$$\mathsf{sif}^*_{\mathrm{avg,I}} = 5.091385419547622, \quad \mathsf{sif}^*_{\mathrm{avg,II}} = -0.330092517099450. \tag{7.17}$$

In Figure 7.27 we plot the relative error of the extracted SIFs to the mean value (for all depicted values of r). The depicted curves clearly show the path-independence of the extraction and we can infer that our reference values $\mathsf{sif}_{\mathrm{avg,I}}$ and $\mathsf{sif}_{\mathrm{avg,II}}$ are accurate with 6 digits of relative accuracy.

Since the convergence of the SIFs is related to the H^1-norm we expect to find an improved convergence behavior also for the SIFs. To validate this assertion we employ three different extraction radii

$$r_1 = \frac{2}{3}, \quad r_2 = \frac{1}{6}, \quad r_3 = \frac{1}{12\sqrt{2}}, \tag{7.18}$$

where the SIFs are computed outside of E_{tip} for r_1 and inside E_{tip} for r_2 and r_3. We define the

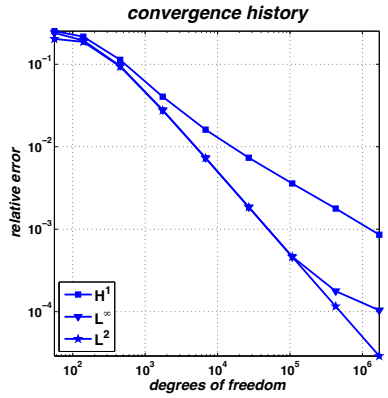


Figure 7.22. *Convergence history of the measured relative errors e (7.3) with respect to the complete domain Ω in the L^∞-norm, the L^2-norm, and the H^1-norm for Example 7.10.*

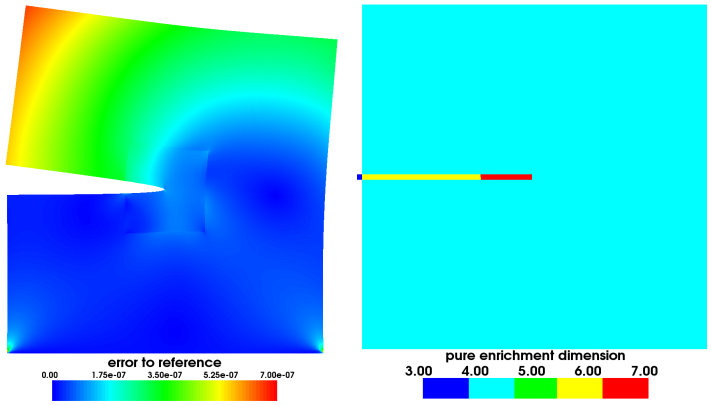


Figure 7.23. *Contour plot of the error (left) on level $k = 9$ with respect to the reference solution $u_{\text{ref}}^{\text{PU}}$ depicted on the (scaled) deformed domain with respect to the deformation $u_k^{\text{PU}} \in V_k^{\text{PU}}$. Contour plot of the dimension $d_{i,k}^{\mathcal{D}}$ of the pure enrichment spaces $\mathcal{D}_{i,k}$ (right) for patches $\omega_{i,k} \subset E_{\text{tip}}$ with $V_{i,k} = \mathcal{P}^1 \oplus \mathcal{D}_{i,k}$ on level $k = 9$.*

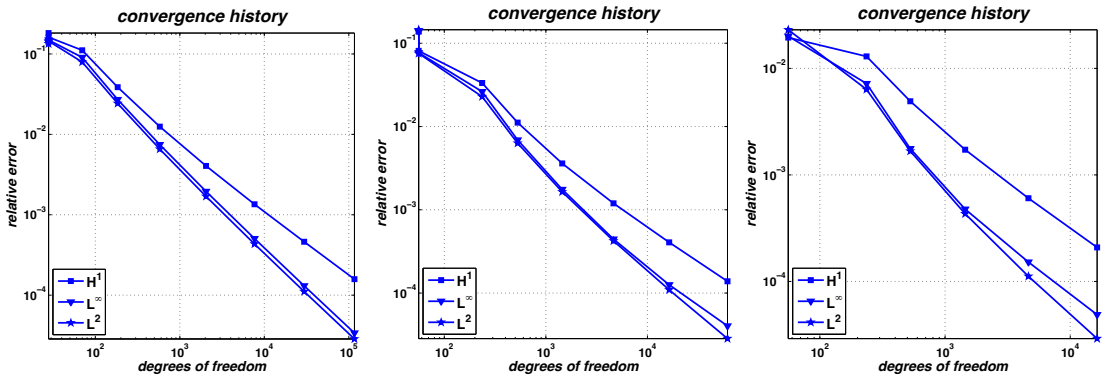


Figure 7.24. *Convergence history of the measured relative errors e (7.3) with respect to the subdomains E_1 (left), E_2 (center), and E_3 (right) given in (7.15) with respect to the L^∞-norm, the L^2-norm, and the H^1-norm for Example 7.10.*

relative errors

$$e := \frac{\mathsf{sif} - \mathsf{sif}^*_{\text{avg}}}{\mathsf{sif}^*_{\text{avg}}}$$

with respect to the reference values (7.17) and measure the respective convergence rates analogous to (7.4). In Table 7.19 we give the relative errors e_1, e_2, and e_3 obtained for the three extraction radii (7.18) with respect to mode I and mode II. From the displayed numbers we can clearly observe that the SIFs converge much faster in the enrichment zone, i.e., $\rho_{1,\text{I}}$ and $\rho_{1,\text{II}}$ are smaller than $\rho_{r,\text{I}}$ and $\rho_{r,\text{II}}$ with $r = 2, 3$ respectively, see also Figure 7.28. The relative errors are about one order smaller inside E_{tip}. Furthermore, we find an almost perfect agreement of the SIFs computed with respect to r_2 and r_3 with an average convergence rate of about 1.

Now we consider a conforming discretization of (7.16) with linear polynomials and an in-

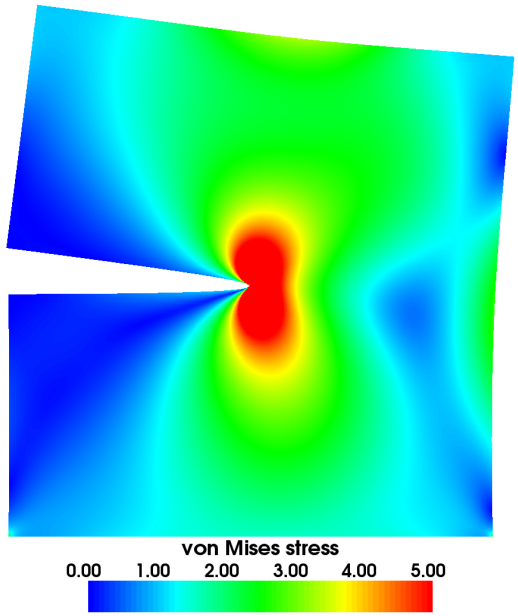


Figure 7.25. *Contour plot of the von Mises stress distribution depicted on the (scaled) deformed domain with respect to the reference solution* $u_{\text{ref}}^{\text{PU}} \in V_{\text{ref}}^{\text{PU}}$.

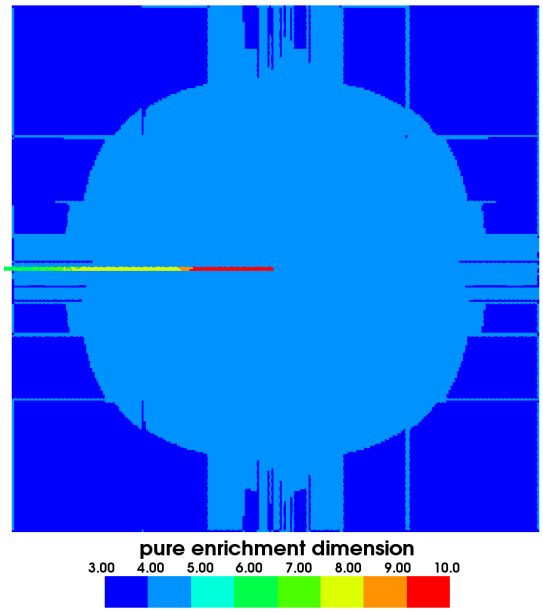


Figure 7.26. *Contour plot of the dimension* $d^{\mathcal{D}}_{i,J_{\text{ref}}}$ *of the pure enrichment spaces* $\mathcal{D}_{i,J_{\text{ref}}}$ *for patches* $\omega_{i,J_{\text{ref}}} \subset E_{\text{tip}}$ *with* $V_{i,J_{\text{ref}}} = \mathcal{P}^2 \oplus \mathcal{D}_{i,J_{\text{ref}}}$ *on the reference level* $J_{\text{ref}} = 10$.

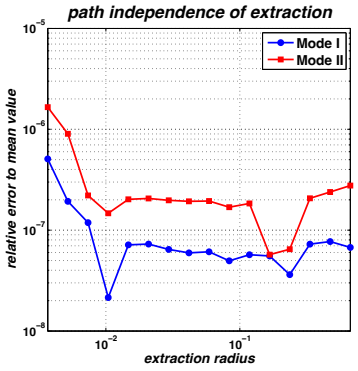


Figure 7.27. *Path independence of SIF extraction for reference solution.*

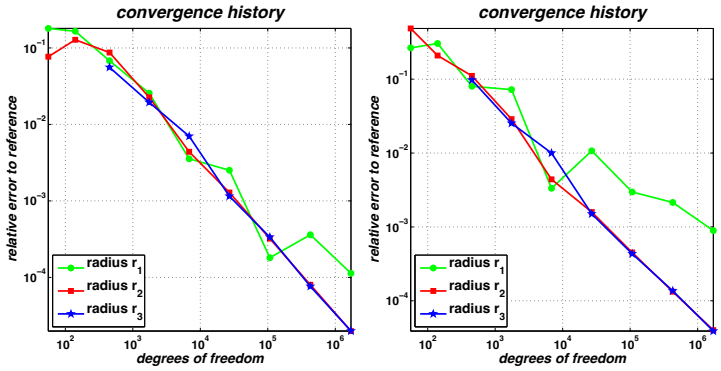


Figure 7.28. *Convergence history of stress intensity factors (left: mode I; right: mode II).*

clined crack $C = \overline{c_l c_u}$ with $c_l = (-1, -0.25)$ and $c_u = (0.0)$. Here we furthermore use a less aggressive enrichment strategy since we construct the preconditioner $S_{\mathcal{E}\setminus\mathcal{P}}$ of §5.3.2 based on the L^2-norm and use $\epsilon = 10^{-10}$ only. Thus, we expect to obtain less enrichment functions in this discretization than above (in fact already on level $k = 7$ some enrichment functions are eliminated). Yet, the convergence properties should still be comparable (up to an error of the order of ϵ).

The measured errors and the respective convergence rates are given in Table 7.20. From these numbers we can again observe that the optimal convergence behavior of the uniform h-

Table 7.19. *Relative errors e and respective convergence rates ρ for the stress intensity factors s_{I} and s_{II} for Example 7.10.*

J	$e_{1,\mathrm{I}}$	$\rho_{1,\mathrm{I}}$	$e_{1,\mathrm{II}}$	$\rho_{1,\mathrm{II}}$	$e_{2,\mathrm{I}}$	$\rho_{1,\mathrm{I}}$	$e_{2,\mathrm{II}}$	$\rho_{2,\mathrm{II}}$	$e_{3,\mathrm{I}}$	$\rho_{3,\mathrm{I}}$	$e_{3,\mathrm{II}}$	$\rho_{3,\mathrm{II}}$
1	1.80_{-1}	–	2.65_{-1}	–	7.68_{-2}	–	4.87_{-1}	–	–	–	–	–
2	1.65_{-1}	0.09	3.04_{-1}	−0.15	1.28_{-1}	−0.56	2.09_{-1}	0.92	–	–	–	–
3	6.79_{-2}	0.76	8.02_{-2}	1.14	8.71_{-2}	0.33	1.11_{-1}	0.54	5.57_{-2}	–	9.75_{-2}	–
4	2.56_{-2}	0.72	7.22_{-2}	0.08	2.26_{-2}	1.00	2.88_{-2}	1.00	1.94_{-2}	0.78	2.53_{-2}	1.00
5	3.56_{-3}	1.45	3.34_{-3}	2.25	4.39_{-3}	1.20	4.41_{-3}	1.38	7.01_{-3}	0.75	1.00_{-2}	0.68
6	2.52_{-3}	0.25	1.07_{-2}	−0.85	1.29_{-3}	0.89	1.59_{-3}	0.74	1.15_{-3}	1.32	1.50_{-3}	1.38
7	1.81_{-4}	1.91	2.97_{-3}	0.93	3.21_{-4}	1.01	4.51_{-4}	0.91	3.36_{-4}	0.89	4.32_{-4}	0.90
8	3.61_{-4}	−0.50	2.14_{-3}	0.24	8.01_{-5}	1.00	1.33_{-4}	0.88	7.64_{-5}	1.07	1.36_{-4}	0.83
9	1.13_{-4}	0.84	8.92_{-4}	0.63	1.99_{-5}	1.01	4.01_{-5}	0.87	1.99_{-5}	0.97	3.90_{-5}	0.90

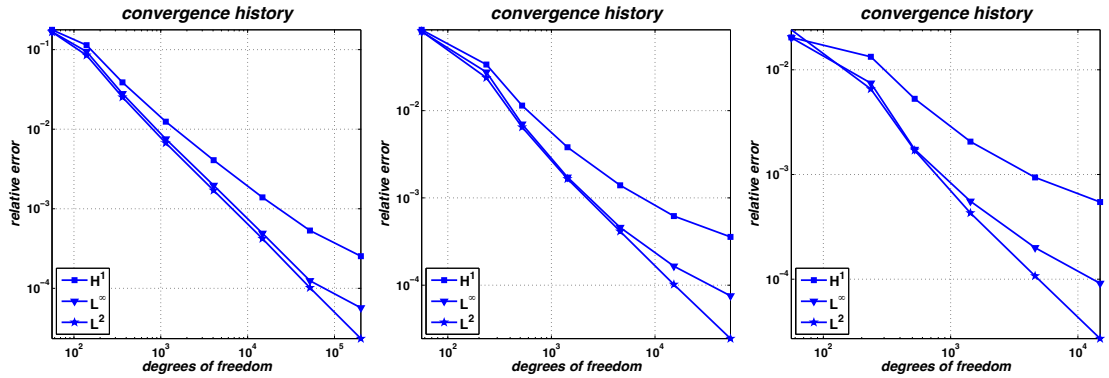


Figure 7.29. *Convergence history of the measured relative errors e (7.3) with respect to the subdomains E_1 (left), E_2 (center), and E_3 (right) given in (7.15) with respect to the L^∞-norm, the L^2-norm, and the H^1-norm for Example 7.10 using a conforming discretization.*

version is fully recovered globally, we again find $\rho_{L^2} = 1$ and $\rho_{H^1} = \frac{1}{2}$. Note that the total number of degrees of freedom $\mathrm{dof} = \mathrm{dof}_K + \mathrm{dof}_I$ though is significantly less in this experiment than with the very aggressive enrichment employed in the previous experiments. Here, we have $\mathrm{dof} = 1674372$ on level $k = 9$ whereas above we employed an enriched PPUM space V_9^{PU} with $\mathrm{dof} = 1706820$. Yet, this reduction in the number of degrees of freedom which is due to the elimination of more enrichment functions in the construction of $S_{\mathcal{E}\setminus\mathcal{P}}$ does not adversely affect the convergence behavior. In fact the attained errors on a particular level k are very much comparable so that the convergence rates ρ tend to be slightly larger on coarser levels in this experiment. On later refinement levels however we may encounter a slight decline of the convergence rates in the enrichment zone since here the impact of the elimination of the enrichment functions on the convergence properties is most prominent. This anticipated behavior can be observed from Table 7.20 and the plots depicted in Figure 7.29.

Example 7.11. In our last example use an h-adaptive refinement together with our hierarchical enrichment scheme and a conforming discretization with linear polynomials. Since the enrichment essentially eliminates the singularity of the solution at the crack tip we anticipate that the adaptive refinement will in fact be close to uniform outside of the enrichment zone and that there is almost no refinement within E_{tip}. Hence, we expect to find similar asymptotic convergence rates as in Example 7.10 with respect to the domain Ω. Within the enrichment zone, however, the combination of adaptive refinement and hierarchical enrichment can yield much higher convergence rates.

Table 7.20. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.10 using a conforming boundary treatment.*

J	dof_K	dof_I	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
with respect to Ω									
2	124	16	16	2.072_{-1}	–	2.025_{-1}	–	2.294_{-1}	–
3	408	32	64	9.899_{-2}	0.65	9.795_{-2}	0.63	1.162_{-1}	0.59
4	1660	64	256	2.842_{-2}	0.91	2.823_{-2}	0.91	4.068_{-2}	0.77
5	6660	128	1024	7.355_{-3}	0.99	7.335_{-3}	0.98	1.600_{-2}	0.68
6	26644	256	4096	1.819_{-3}	1.01	1.820_{-3}	1.01	7.296_{-3}	0.57
7	106080	512	16384	5.241_{-4}	0.90	4.396_{-4}	1.03	3.568_{-3}	0.52
8	419102	1024	65536	3.140_{-4}	0.37	1.026_{-4}	1.06	1.791_{-3}	0.50
9	1672324	2048	262144	1.834_{-4}	0.39	2.231_{-5}	1.10	8.838_{-4}	0.51
with respect to E_1									
2	56	0	4	1.672_{-1}	–	1.661_{-1}	–	1.774_{-1}	–
3	140	0	16	9.434_{-2}	0.62	8.475_{-2}	0.73	1.135_{-1}	0.49
4	362	0	36	2.794_{-2}	1.28	2.522_{-2}	1.28	3.876_{-2}	1.13
5	1142	0	100	7.533_{-3}	1.14	6.702_{-3}	1.15	1.242_{-2}	0.99
6	4064	0	324	1.965_{-3}	1.06	1.699_{-3}	1.08	4.075_{-3}	0.88
7	14804	0	1156	4.914_{-4}	1.07	4.207_{-4}	1.08	1.392_{-3}	0.83
8	52194	0	4356	1.245_{-4}	1.09	1.016_{-4}	1.13	5.327_{-4}	0.76
9	201192	0	16900	5.687_{-5}	0.58	2.337_{-5}	1.09	2.533_{-4}	0.55
with respect to E_2									
3	56	0	4	8.055_{-2}	–	8.122_{-2}	–	8.449_{-2}	–
4	236	0	16	2.764_{-2}	0.74	2.384_{-2}	0.85	3.375_{-2}	0.64
5	522	0	36	6.998_{-3}	1.73	6.467_{-3}	1.64	1.142_{-2}	1.37
6	1430	0	100	1.729_{-3}	1.39	1.653_{-3}	1.35	3.809_{-3}	1.09
7	4608	0	324	4.588_{-4}	1.13	4.138_{-4}	1.18	1.394_{-3}	0.86
8	15210	0	1156	1.650_{-4}	0.86	1.018_{-4}	1.17	6.195_{-4}	0.68
9	53472	0	4356	7.536_{-5}	0.62	2.444_{-5}	1.13	3.575_{-4}	0.44
with respect to E_3									
4	56	0	4	2.024_{-2}	–	2.409_{-2}	–	2.035_{-2}	–
5	236	0	16	7.448_{-3}	0.69	6.547_{-3}	0.91	1.331_{-2}	0.30
6	522	0	36	1.734_{-3}	1.84	1.700_{-3}	1.70	5.282_{-3}	1.16
7	1430	0	100	5.551_{-4}	1.13	4.287_{-4}	1.37	2.059_{-3}	0.93
8	4606	0	324	1.997_{-4}	0.87	1.072_{-4}	1.18	9.370_{-4}	0.67
9	14928	0	1156	9.119_{-5}	0.67	2.709_{-5}	1.17	5.452_{-4}	0.46

The measured errors and convergence rates are given in Table 7.21. From the displayed numbers we can observe the anticipated global convergence behavior with the rates $\rho_{L^2} \geq 1$ and $\rho_{H^1} \geq \frac{1}{2}$. The plots depicted in Figure 7.30 clearly show the expected refinement behavior. Outside of the enrichment zone we attain an almost uniform refinement with some additional refinement steps near the boundary of E_{tip} and the Dirichlet-Neumann corners. Observe the jump in the patch size at the boundary of E_{tip} ($L_{\max} \geq 3$) which accounts for the jump in the resolution of the local approximation spaces V_i. Outside of E_{tip} we employ linear polynomials only and within E_{tip} linear polynomials and the enrichment spaces (5.22) and (5.23). Inside the enrichment zone we find a highly localized refinement towards the crack tip which yields a very fast convergence with E_{tip}, see Table 7.21. Since the local approximation spaces in E_{tip} are much richer than just linear polynomials and themselves adapted to the asymptotic behavior of the solution the convergence rates in E_{tip} are much higher than for Ω. We measure rates ρ_{H^1} around 1.5 which corresponds to the optimal convergence of a uniform discretization with $p = 3$. Clearly, the super-convergence effect of our hierarchical enrichment approach is further enhanced by the adaptive h-refinement.

Table 7.21. *Relative errors e (7.3) and convergence rates ρ (7.4) for Example 7.11 using an h-adaptive conforming boundary treatment.*

J	dof_K	dof_I	N	e_{L^∞}	ρ_{L^∞}	e_{L^2}	ρ_{L^2}	e_{H^1}	ρ_{H^1}
with respect to Ω									
1	36	20	4	4.715_{-1}	0.19	4.445_{-1}	0.20	5.004_{-1}	0.17
2	124	16	16	2.078_{-1}	0.89	2.031_{-1}	0.85	2.301_{-1}	0.85
3	408	32	64	9.923_{-2}	0.65	9.819_{-2}	0.63	1.164_{-1}	0.59
4	1126	52	187	3.617_{-2}	1.02	3.524_{-2}	1.04	4.869_{-2}	0.89
5	2038	64	331	2.146_{-2}	0.90	2.092_{-2}	0.90	3.264_{-2}	0.69
6	3348	92	544	1.287_{-2}	1.04	1.243_{-2}	1.06	2.256_{-2}	0.75
7	4918	112	799	8.038_{-3}	1.24	7.899_{-3}	1.19	1.677_{-2}	0.78
8	7336	160	1204	5.638_{-3}	0.89	5.433_{-3}	0.94	1.347_{-2}	0.55
9	12536	192	2053	3.109_{-3}	1.12	2.989_{-3}	1.13	9.605_{-3}	0.64
10	19550	240	3211	1.937_{-3}	1.07	1.876_{-3}	1.06	7.401_{-3}	0.59
11	32168	348	5296	1.286_{-3}	0.82	1.236_{-3}	0.84	5.806_{-3}	0.49
with respect to E_{tip}									
2	56	0	4	1.679_{-1}	0.44	1.667_{-1}	0.45	1.780_{-1}	0.43
3	140	0	16	9.460_{-2}	0.63	8.500_{-2}	0.74	1.138_{-1}	0.49
4	182	0	24	3.505_{-2}	3.78	2.799_{-2}	4.23	5.365_{-2}	2.86
5	344	0	41	1.954_{-2}	0.92	1.787_{-2}	0.70	3.078_{-2}	0.87
6	458	0	52	1.154_{-2}	1.84	1.071_{-2}	1.79	2.072_{-2}	1.38
7	614	0	68	7.477_{-3}	1.48	6.870_{-3}	1.51	1.257_{-2}	1.70
8	704	0	83	5.301_{-3}	2.51	4.541_{-3}	3.03	9.978_{-3}	1.69
9	932	0	102	2.827_{-3}	2.24	2.474_{-3}	2.17	5.804_{-3}	1.93
10	1220	0	136	1.958_{-3}	1.36	1.590_{-3}	1.64	4.219_{-3}	1.18
11	1730	0	190	1.264_{-3}	1.25	1.037_{-3}	1.23	2.560_{-3}	1.43

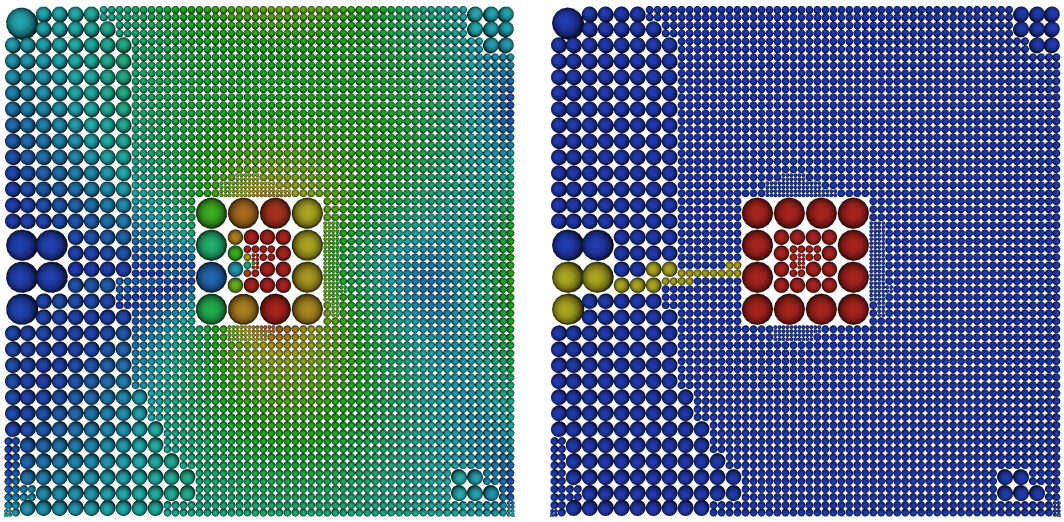


Figure 7.30. *Distribution of the von Mises stress (left) and the enrichment spaces (right) for Example 7.11 on the cell centers of the respective tree decomposition for level $J = 10$. The size of the particle at the cell center indicates the size of the respective support patch ω_i, the color of the particle indicates the von Mises stress at the cell center (left), or the type of enrichment space employed on ω_i (blue: $\mathcal{E}_i = \{0\}$, yellow: $\mathcal{E}_i = H^{\mathrm{C}}_{\pm}\mathcal{P}^1$, red: $\mathcal{E}_i \supset W_{\mathrm{tip}}$).*

7.2 Solver Efficiency

Let us now consider the solution of the large sparse linear system

$$A\tilde{u} = \hat{f} \tag{7.19}$$

arising from a PPUM Galerkin discretization of a PDE. Here, A denotes the PPUM stiffness matrix which is symmetric positive definite for the considered model problems (7.1) and (7.2) with appropriate boundary conditions, $\tilde{u} = (u_i^m)$ is the coefficient vector associated with a particular function $u^{\mathrm{PU}} \in V^{\mathrm{PU}}$, i.e.

$$u^{\mathrm{PU}} = \sum_{i=1}^{N} \varphi_i \sum_{m=1}^{\dim(V_i)} u_i^m \vartheta_i^m,$$

and $\hat{f} = (\langle l, \varphi_i \vartheta_i^m \rangle)$ refers to the load vector.

The standard solver employed in our PPUM is a conjugate gradient (CG) method with a multilevel $V(2,2)$-cycle using block-Gauss–Seidel smoothing as preconditioner and an initial guess obtained from the prolongation of the approximate solution obtained on a coarser level. Besides this nested iteration with a preconditioned CG solver we also consider our multilevel iteration presented in §5 as a stand-alone solver. In general we must consider stopping criteria based on the residual vector

$$\hat{r}_q := \hat{f} - A\tilde{u}_q \tag{7.20}$$

where the sub-script q denotes the current iteration step. In all our experiments we stop the iteration if $q = 30$ or if the l^2-norm of the residual vector $\|\hat{r}_q\|_{l^2} < 10^{-13}$. Furthermore, we are concerned with the iterative update

$$\tilde{h}_q := \tilde{u}_q - \tilde{u}_{q-1} \tag{7.21}$$

and the correction

$$\tilde{c}_q := \tilde{u}_q - \tilde{u}_0 \tag{7.22}$$

to measure the convergence behavior and performance of our linear solver. By itself the update $\tilde{h}_q$ allows us to determine the stagnation of the iteration only which in general however gives no indication if the iteration has truly converged. For an arbitrary initial guess the correction $\tilde{c}_q$ provides no additional information. Yet with the particular choice of the initial guess $\tilde{u}_0$ as the (prolonged) solution of a coarser approximation we can interpret the correction $\tilde{c}_q$ or its associated function c_q^{PU} as an approximation of the discretization error.

We assess the quality of the stand-alone multilevel iteration via the use of a vanishing right-hand side $\hat{f} = 0$ and a normalized random-valued initial guess $\tilde{u}_0$. Due to this choice for the load vector the iterates $\tilde{u}_q$ agree with the error $\tilde{e}_q := \tilde{u}_q - \tilde{u}_\infty$ where $\tilde{u}_\infty$ denotes the exact solution of (7.19), i.e. for $\hat{f} = 0$ there holds $\tilde{u}_\infty = 0$. Note that we in fact normalize the respective function u_0^{PU}. To this end, we define the notion of an L^2-norm of a coefficient vector $\tilde{u}$ by

$$\|\tilde{u}\|_{L^2}^2 := \tilde{u}^T M \tilde{u} = \|u^{\mathrm{PU}}\|_{L^2}^2$$

with the help of the mass matrix M so that $\|\tilde{u}\|_{L^2}^2$ is identical to the L^2-norm of the respective function u^{PU} and we enforce $\|\tilde{u}_0\|_{L^2} = 1$ for the random valued initial guess $\tilde{u}_0$. Analogously, we define the energy-norm

$$\|\tilde{u}_0\|_E^2 := \tilde{u}_0^T A \tilde{u}_0 = a(u_0^{\mathrm{PU}}, u_0^{\mathrm{PU}}).$$

The average convergence rate of the first q iterations with respect to error in the L^2-norm is defined as

$$\bar{\rho}_{L^2} := \left(\frac{\|\tilde{e}_q\|_{L^2}}{\|\tilde{e}_0\|_{L^2}} \right)^{\frac{1}{q}}.$$

Which can easily be determined in the case of $\hat{f} = 0$ since $\tilde{e}_q = \tilde{u}_q$, i.e.

$$\bar{\rho}_{L^2} = \left(\frac{\|\tilde{u}_q\|_{L^2}}{\|\tilde{u}_0\|_{L^2}} \right)^{\frac{1}{q}} = \left(\frac{\tilde{u}_q^T M \tilde{u}_q}{\tilde{u}_0^T M \tilde{u}_0} \right)^{\frac{1}{q}},$$

and with $\|\tilde{u}_0\|_{L^2} = 1$ there holds $\bar{\rho}^r_{L^2} = \|\tilde{u}_q\|_{L^2}$. Furthermore, we define the asymptotic convergence rate $\rho^*_{L^2}$ by considering the average of the last three iterations, i.e.

$$\rho^*_{L^2} := \left(\frac{\|\tilde{u}_q\|_{L^2}}{\|\tilde{u}_{q-3}\|_{L^2}} \right)^{\frac{1}{3}}.$$

Analogously we define the convergence rates with respect to the l^2-norm and the energy-norm of the iterate $\tilde{u}_q$, the correction $\tilde{c}_q$, and the iterative update $\tilde{h}_q$. For the residual vector $\hat{r}$ we consider the l^2-norm only.

Example 7.12. In our first example we consider the multilevel solution of the linear systems arising from the PPUM discretization of the scalar model problem (7.1) with $c = 1$ and Neumann boundary conditions on the domain $\Omega = (-1, 1)^2$ using a uniformly h-refined sequence of spaces V_l^{PU} with $l = 0, \ldots, J$ and linear local approximation spaces $V_{i,l} = \mathcal{P}^1$ for all $i = 1, \ldots, N_l$.

In Figure 7.31 we summarize the performance of our standard nested iteration solver. Depicted are the convergence history of the residual $\hat{r}$, of the correction $\tilde{c}$, and of the iterative update $\tilde{h}$. From these line plots we can clearly observe the level-independent convergence of the solver. For each level $l = 1, , \ldots, 11$ we need about 10 iterations to reduce the l^2-norm of the residual vector below machine accuracy.

From the rapid convergence of the correction $\tilde{c}$ and the update $\tilde{h}$ however we see that we can safely stop the iterative solver even after a single iteration to obtain an approximate solution within discretization accuracy. Observe also that the L^2-norm of the correction decreases by a factor of 0.25 from level to level which agrees perfectly with the reduction of the discretization error due to the uniform h-refinement with linear polynomials in two dimensions. With respect to the energy-norm we attain the anticipated reduction factor of 0.5.

Recall from section 4.4 that the nested iteration approach should converge with just a single iteration if the asymptotic convergence rate of the employed inner solver is less or equal to the reduction of the discretization error due to refinement. The measured asymptotic and average convergence rates for our default solver are depicted in Figure 7.31. The rates are clearly bounded below 0.06 which is much smaller than the reduction in the discretization error. Note also that the average convergence rates agree very well with the asymptotic rates which indicates that the convergence behavior is very stable in each iteration, compare Figure 7.31.

Let us now consider the efficiency of our multilevel iteration as a stand-alone solver. Here, we consider a vanishing right-hand side with random-valued initial guess so that we consider the convergence of iterate $\tilde{u}$ rather than iterative update $\tilde{h}$. The measured convergence rates are collected in Figure 7.32. We expect that the convergence behavior improves with multiple smoothing steps and that the convergence rates for the $V(1,1)$-cycle and the $V(3,3)$-cycle are somewhat larger than those of Figure 7.31 due to the use of an outer CG method in the standard solver. This anticipated behavior is clearly attained. We find the rates 0.27 for the $V(1,1)$-cycle, the $V(3,3)$-cycle yields a rate of 0.11, and the $V(5,5)$-cycle reduces the iteration errors by a factor of 0.07 in each iteration.

Thus, our multilevel iteration provides an optimally convergent stand-alone solver and its use as a preconditioner for a CG method further speeds up the convergence. Within a nested iteration approach a single application of the $V(s,s)$-cycle with $s > 1$ already provides an approximate solution within discretization accuracy.

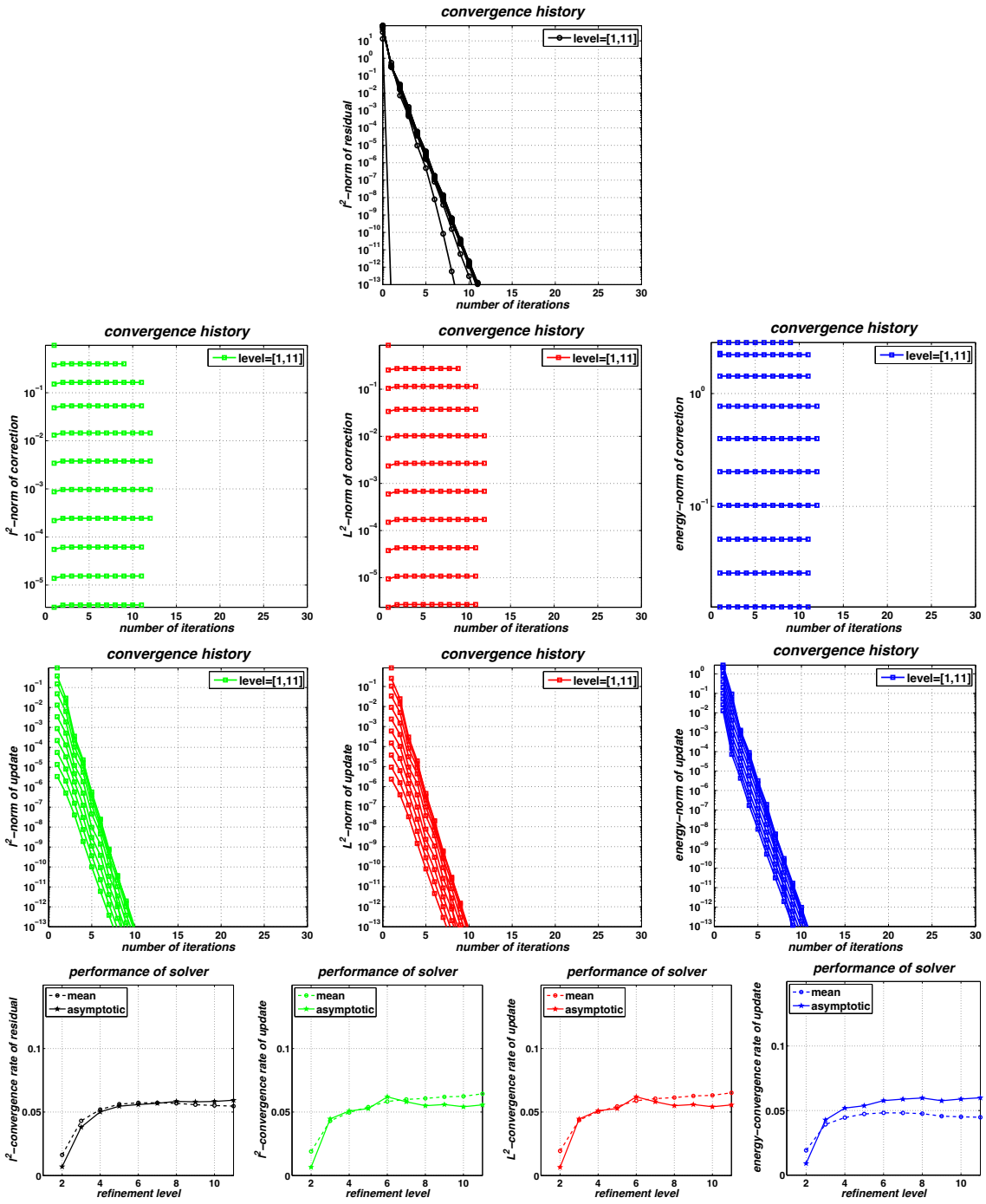


Figure 7.31. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 1$ and Neumann boundary conditions using uniformly h-refined PPUM spaces V_l^{PU} with linear local approximation spaces.*

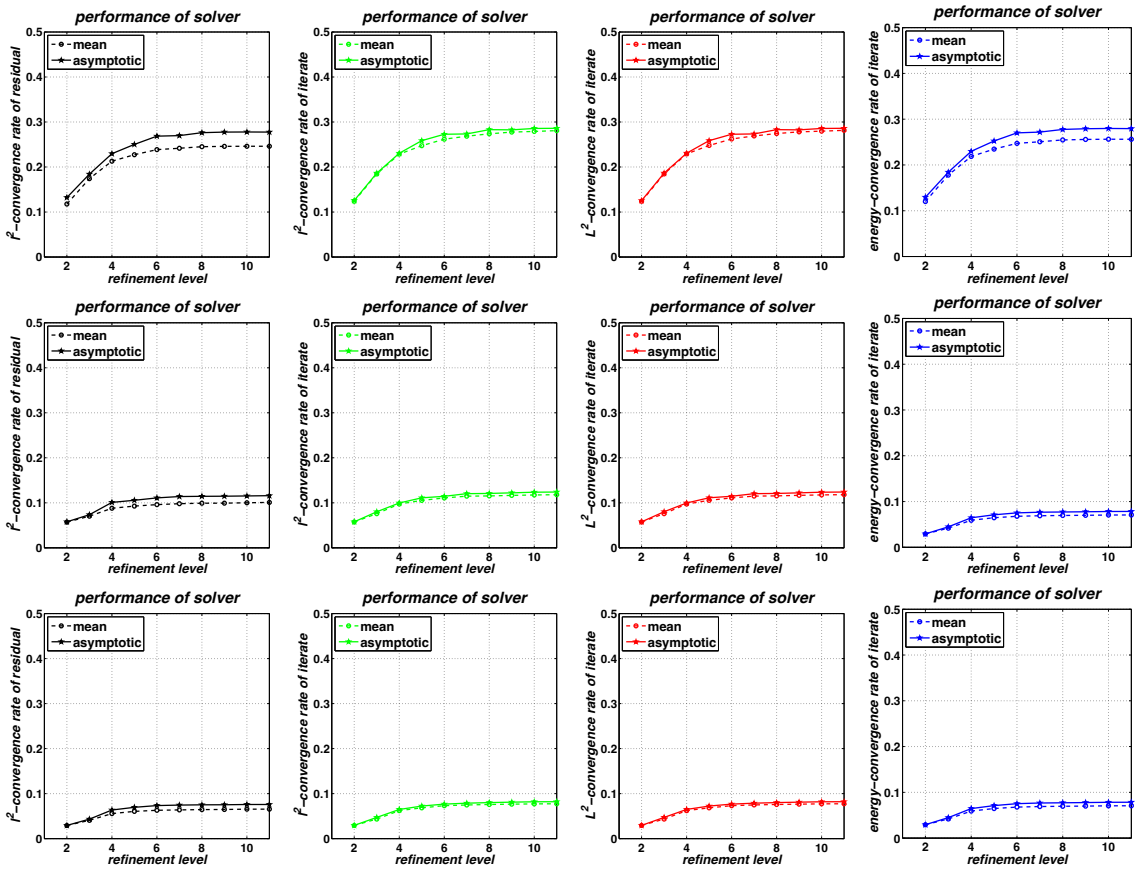


Figure 7.32. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 1$ and Neumann boundary conditions using uniformly h-refined PPUM spaces V_l^{PU} with linear local approximation spaces.*

Example 7.13. Let us now consider the efficient solution of the linear systems arising from the PPUM discretization of (7.1) with $c = 0$ and Dirichlet boundary conditions. Again, we use linear local approximation spaces and a uniformly h-refined sequence of PPUM spaces. Here, we compare the results of the non-conforming Nitsche method using the optimal regularization parameter with those of the conforming construction of §4.2.2.

We begin with the non-conforming discretization. The measured results are presented in Figures 7.33 and 7.34. We essentially obtain an identical convergence behavior of our multilevel solvers as in the previous experiment. The nested iteration attains an approximate solution within discretization accuracy with just a single iteration per level and its convergence rate is roughly 0.07. Also the results obtained for the stand-alone multilevel solver are very much comparable to those of Figure 7.32. Here, we find the rates 0.27, 0.11, and 0.07 for the $V(s,s)$-cycles with $s = 1,3,5$ respectively.

The results attained for the conforming approach are collected in Figures 7.35 and 7.36. Recall that in the conforming approach we consider the block-partitioned linear system

$$\begin{pmatrix} A_{K,K} & A_{K,I} \\ 0 & \hat{B}_{I,I} \end{pmatrix} \begin{pmatrix} \tilde{u}_K \\ \tilde{u}_I \end{pmatrix} = \begin{pmatrix} \hat{f}_K \\ \hat{g}_I \end{pmatrix}$$

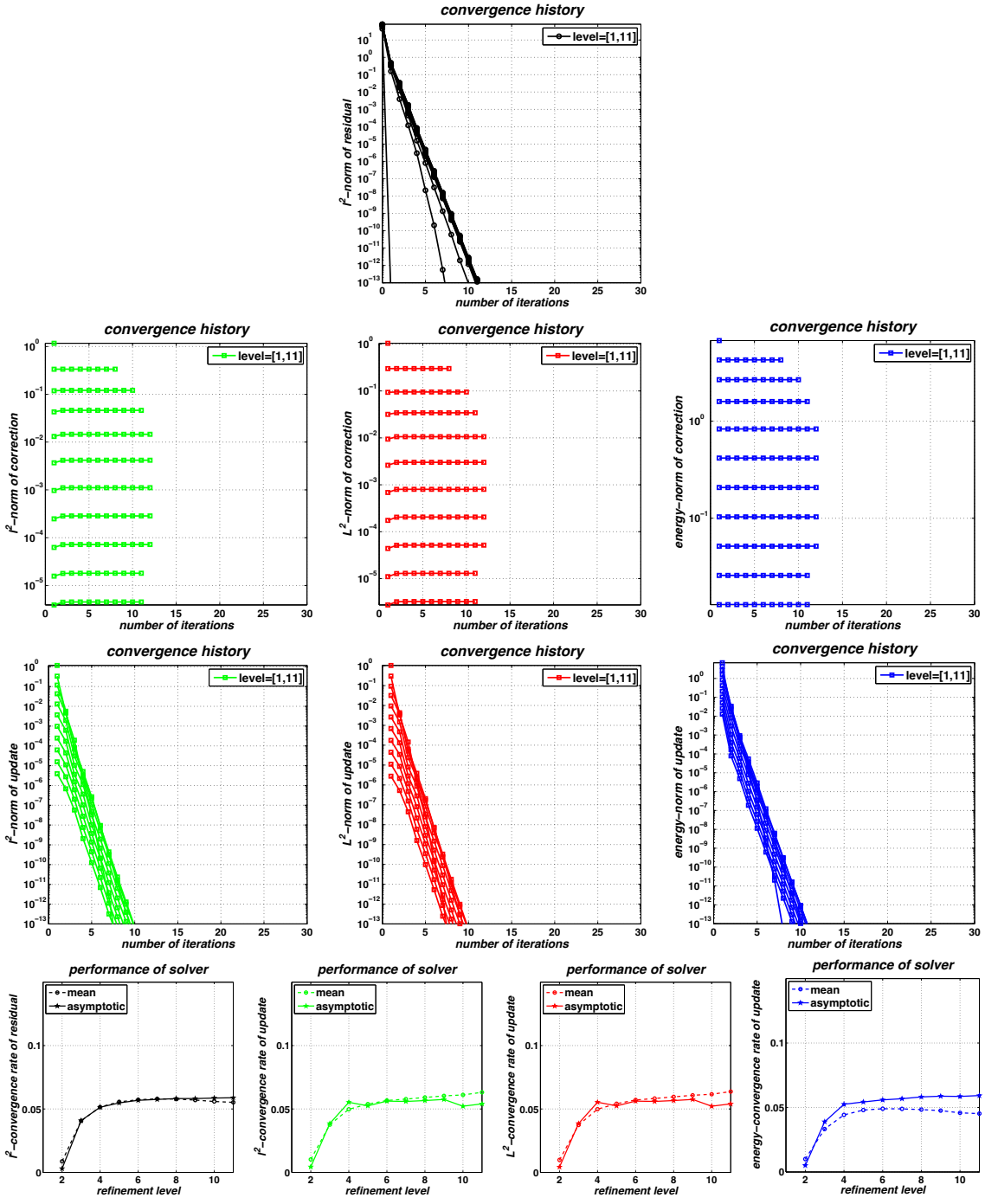


Figure 7.33. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 0$ and Dirichlet boundary conditions using Nitsche's method with the optimal regularization parameter and uniformly h-refined PPUM spaces V_l^{PU} with linear local approximation spaces.*

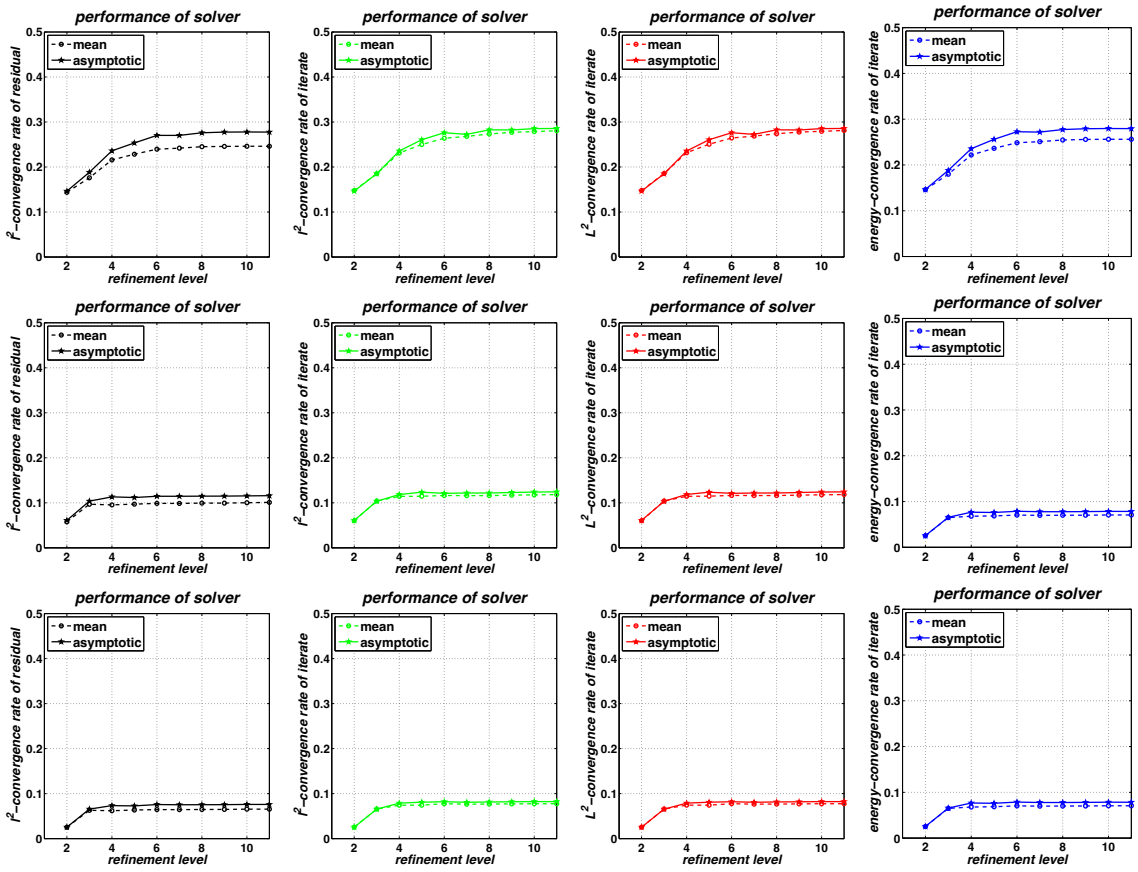


Figure 7.34. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 0$ and Dirichlet boundary conditions using Nitsche's method with the optimal regularization parameter using uniformly h-refined PPUM spaces V_l^{PU} with linear local approximation spaces.*

and apply our multilevel solver to the solution of

$$A_{K,K}\tilde{u}_K = \hat{f}_K - A_{K,I}\tilde{u}_I$$

with $\tilde{u}_I = \hat{B}_{I,I}^{-1}\hat{g}_I$ only. Again, all our multilevel iterative solvers converge with rates that are independent of the number of levels. However we must acknowledge a slight increase of the convergence rates for the conforming approach compared with those of the previous experiments. The preconditioned CG solver now reduces the iteration error roughly by a factor of 0.15 and for the $V(s,s)$-cycles we find the rates 0.6, 0.35, and 0.2 with $s = 1,3,5$ respectively. What causes this loss in performance?

In the Nitsche method our local-to-local restriction and prolongation operate on the *complete* local approximation spaces V_i. Thus, the transfer operators maintain the exactness for all polynomials $\pi \in \mathcal{P}^p$ if $\mathcal{P}^p \subset V_{i,m}$ for all $i = 1,\dots,N_m$ and all levels $m = 0,\dots,J$. Hence, the differences

$$V_{m+1}^{\mathrm{PU}} \setminus V_m^{\mathrm{PU}}$$

of our nonnested sequence of PPUM spaces are minimal on all levels. With the conforming ap-

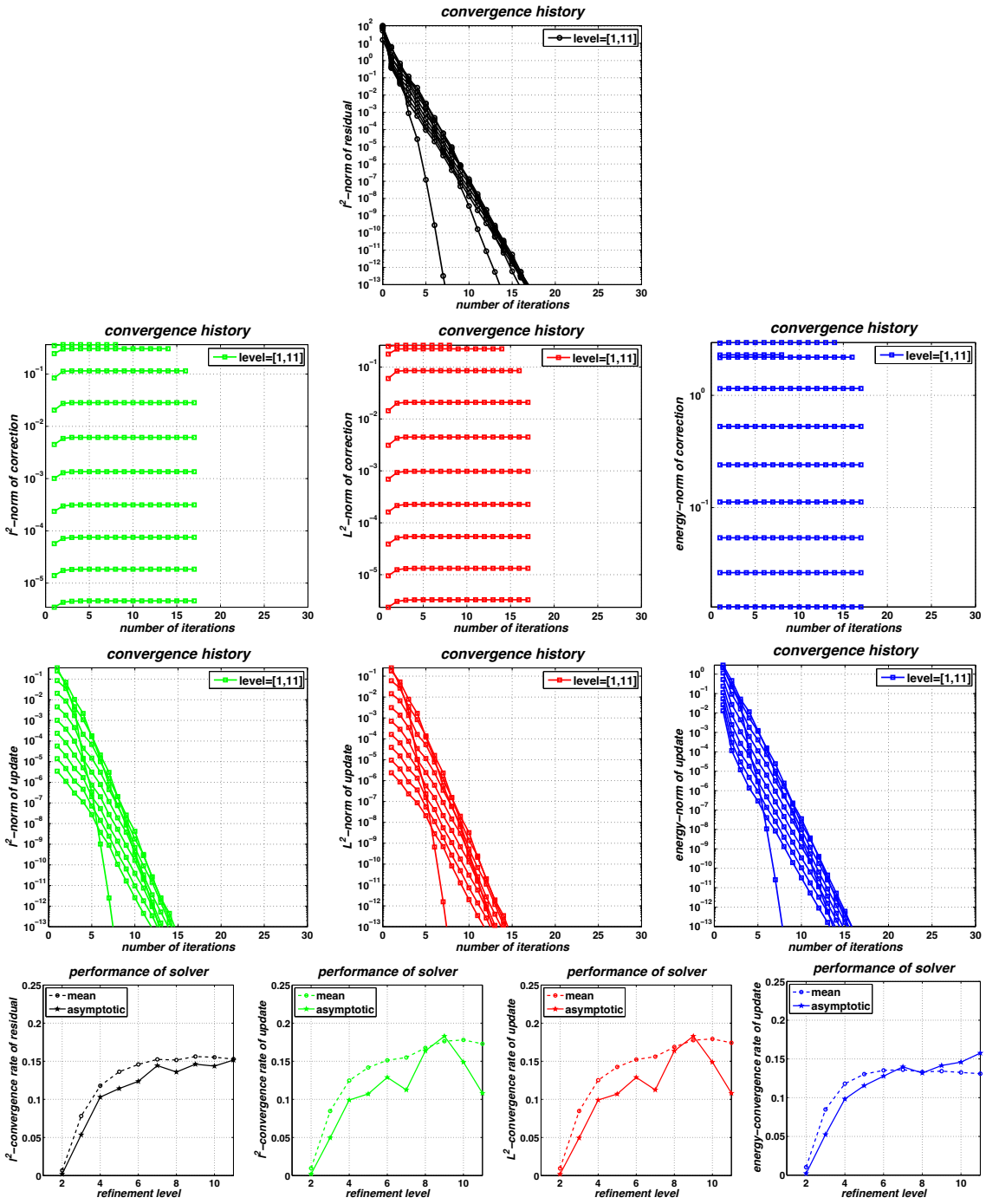


Figure 7.35. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 0$ and Dirichlet boundary conditions using uniformly h-refined PPUM spaces V_l^{PU} with the conforming subspaces of $\mathcal{P}^1$ as local approximation spaces.*

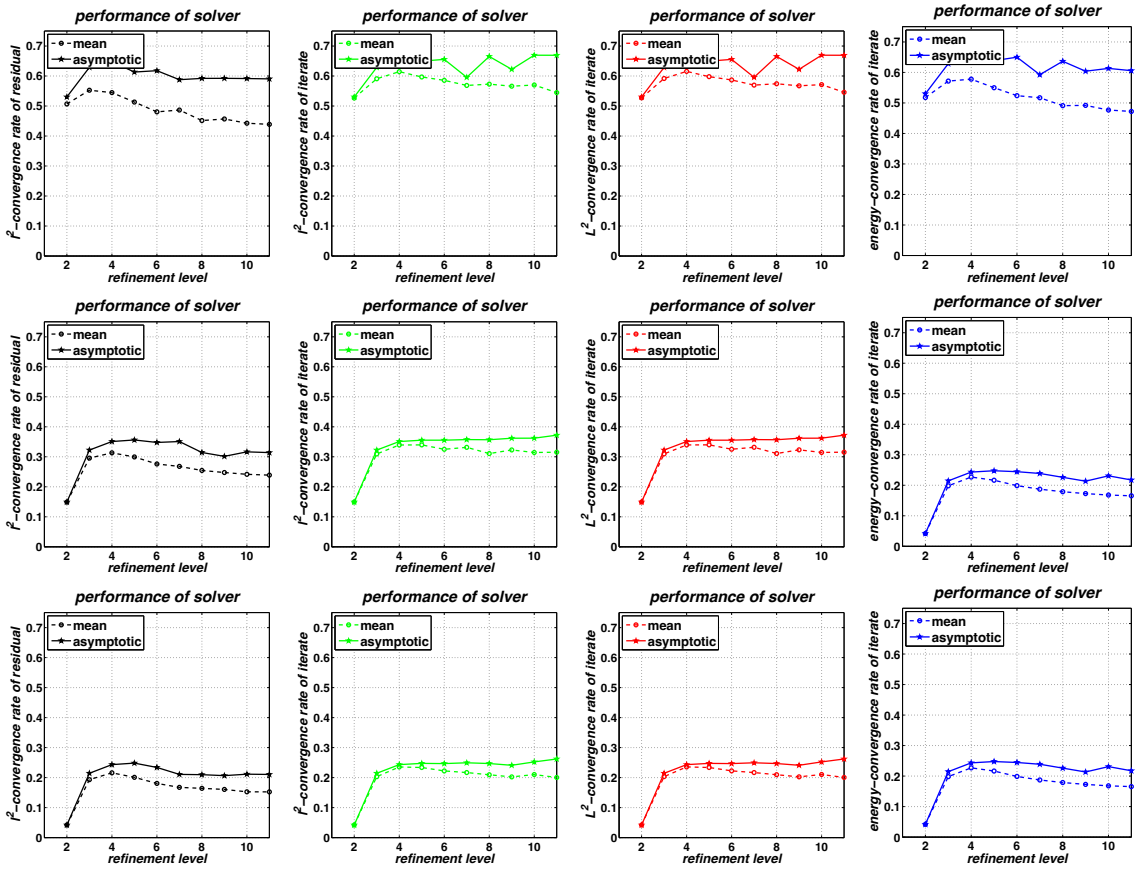


Figure 7.36. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 0$ and Dirichlet boundary conditions using uniformly h-refined PPUM spaces V_l^{PU} with the conforming subspaces of $\mathcal{P}^1$ as local approximation spaces.*

proach however we base our multilevel construction to the sequence of conforming subspaces

$$V_{m,K}^{\mathrm{PU}} = \sum_{i=1}^{N_m} \varphi_{i,m} V_{i,m,K}$$

only. Due to the elimination of the degrees of freedom associated with the boundary conditions $V_{m,I}^{\mathrm{PU}} := V_m^{\mathrm{PU}} \setminus V_{m,K}^{\mathrm{PU}}$ on each level $m = 0,\ldots,J$ we unfortunately increase the size of the difference spaces

$$V_{m+1,K}^{\mathrm{PU}} \setminus V_{m,K}^{\mathrm{PU}}$$

and thereby the nonnestedness compared with the non-conforming approach. Therefore, we assume that the quality of our coarse grid correction step in the conforming approach is reduced near the boundary. To confirm this assertion we consider the projection

$$E_m \tilde{u}_m := (\mathbb{I}_m - I_{m-1}^{l} M_{m-1}^{-1} I_m^{m-1} M_m)\tilde{u}_m \tag{7.23}$$

where M_m denotes the mass matrix on level m. The operator E_m identifies those fine scale components of a coefficient vector $\tilde{u}_m$ on level m that cannot be represented on the coarser level $m-1$

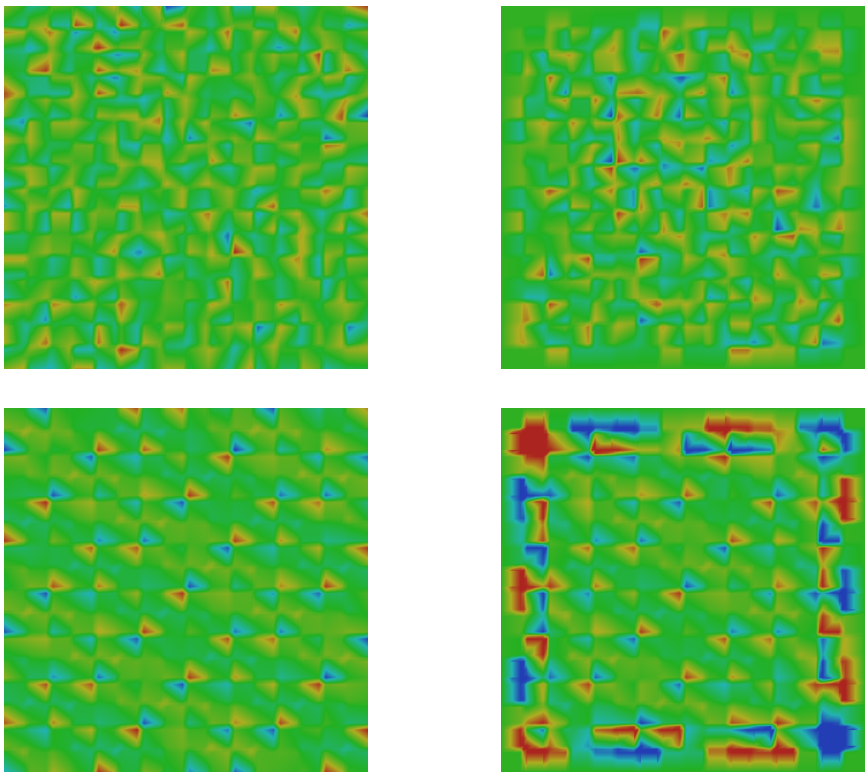


Figure 7.37. *Contour plots of the coarse grid projection error $E_l\tilde{u}_l$ of (7.23) with $l = 4$ for a random valued vector $\tilde{u}_l$ (top row) and a smooth coefficient vector $\tilde{u}_l$ (bottom row) (left: Nitsche method; right: conforming).*

using the restriction I_m^{m-1} and the prolongation I_{m-1}^m. Thus, (7.23) is well-suited to determine deficiencies of the coarse grid correction step of our multilevel iteration. In Figure 7.37 we give the contour plots of the projection error $E_l\tilde{u}_l$ for the Nitsche approach and the conforming discretization for a random valued coefficient vector $\tilde{u}_l$ and a coefficient vector $\tilde{u}_l$ corresponding to a smooth function. The errors $E_l\tilde{u}_l$ for the random valued vector $\tilde{u}_l$ serve as a quality measure for the coarse grid correction in the first iterations of our multilevel solver whereas the errors attained for the smooth function identify defects of the coarse grid correction at later iterations. With respect to the random valued test vector $\tilde{u}_l$ we find no qualitative difference in the projection error between the conforming and non-conforming approaches. Thus, the initial convergence behavior of the $V(s,s)$-cycles for the conforming and non-conforming discretization should agree. From Figure 7.38 we can observe this behavior. Yet, we can clearly see that the convergence rate for the conforming discretization increases after the first few iterations. Note that we essentially find a kink in the line plots. Observe though that the asymptotic rate is still bounded independent of the number of levels. Hence, we infer that we experience a jump in the quality of the coarse grid correction once the iterate is smooth, compare Figure 7.37. In fact we find a strong deviation in the projection errors for a smooth coefficient vector $\tilde{u}$. In the conforming approach we find a large projection error near the boundary (exactly one patch from the boundary) whereas in the non-conforming approach the error is uniformly distributed. This is due to the fact that a coarse patch $\omega_{j,l-1}$ with $\omega_{j,l-1} \cap \partial\Omega \neq \emptyset$ cannot capture all degrees of freedom of the finer patches $\omega_{i,l} \subset \omega_{j,l-1}$ due to the elimination of the boundary degrees of freedom from $V_{j,l-1}$.

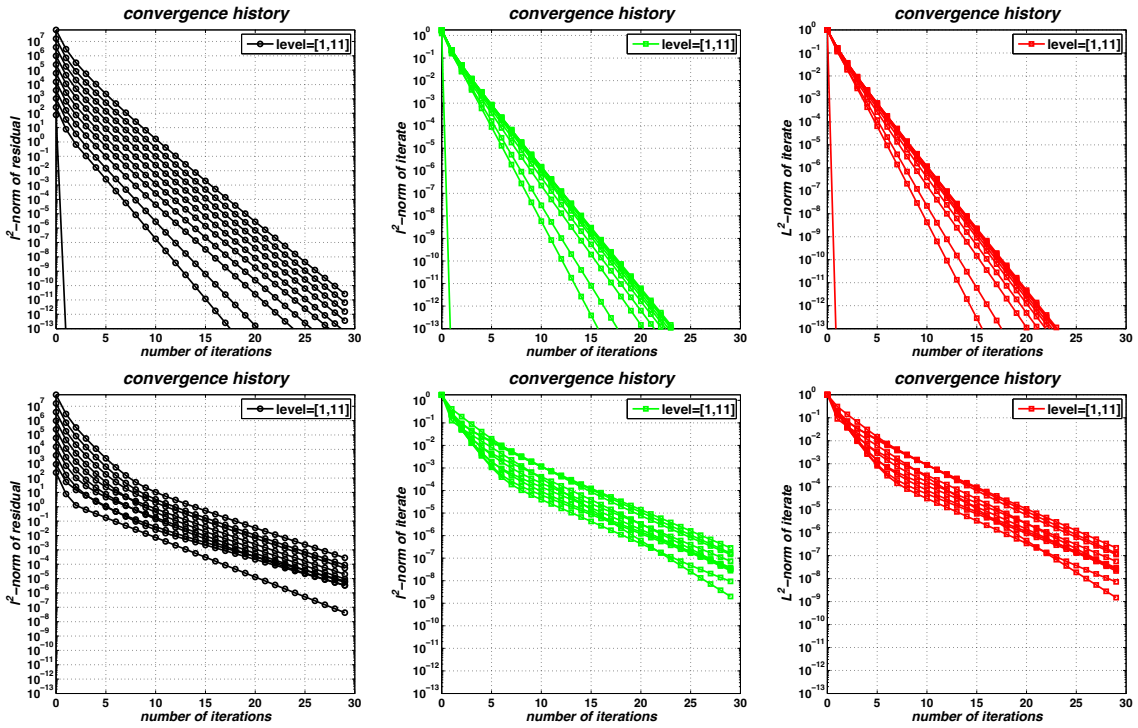


Figure 7.38. *Convergence history of the $V(1,1)$-cycle with block-Gauss–Seidel smoothing for the non-conforming discretization (top row) and the conforming approach (bottom row) using linear polynomials.*

The worst case scenario arises with the choice of $V_{i,m} = \mathcal{P}^1$ for all $i = 1, \dots, N_m$ and all levels $m = 0, \dots, J$. Then our conforming splitting of the local approximation spaces yields

$$V_{i,m,K} = \{0\} \quad \text{and} \quad V_{i,m,I} = V_{i,m} = \mathcal{P}^1$$

at the corners of the domain. Thus all degrees of freedom of a corner patch are fixed by the approximation of the boundary conditions so that the supports of the basis functions of the conforming spaces $V_{m,K}^{\mathrm{PU}}$ do not cover the complete domain Ω on any level $m = 0, \dots, J$ - the corner patches $\omega_{C,m}$ are missing. Therefore, we obtain patches $\omega_{i,m+1} \subset \omega_{C,m}$ with $\omega_{i,m+1} \cap \partial\Omega = \emptyset$ such that $V_{i,m+1,K} = V_{i,m+1} = \mathcal{P}^1$ which cannot be captured by $V_{C,m,K} = \{0\}$ on the coarser level l. These degrees of freedom are not corrected at all and are present on the finer level $m+1$ only which means that the respective error components must be handled completely by the smoother on level $m+1$.

In principal this deficiency is encountered for any choice of local approximation spaces $V_{i,m}$. The coarser spaces $V_{j,m,K}$ are smaller than the collection of the respective finer spaces $V_{i,m+1,K}$ so that the coarse grid correction does not provide an update of all fine scale degrees of freedom. This interpretation leads to the assertion that the conforming approach will always yield a somewhat reduced multilevel convergence rate. Yet, this is *not* the case. From the discussion given above we essentially learn that the restriction operator I_{m+1}^m projects the larger spaces $V_{i,m+1,K}$ onto smaller spaces $V_{j,m,K}$. This however may also yield a *smoother* approximation on the coarser level m and so the restriction serves as an additional filter for fine scale error components. Following this line of argument the quality of the coarse grid correction can actually improve in the conforming approach. From the contour plots of the respective projection errors depicted in Figure 7.39 for

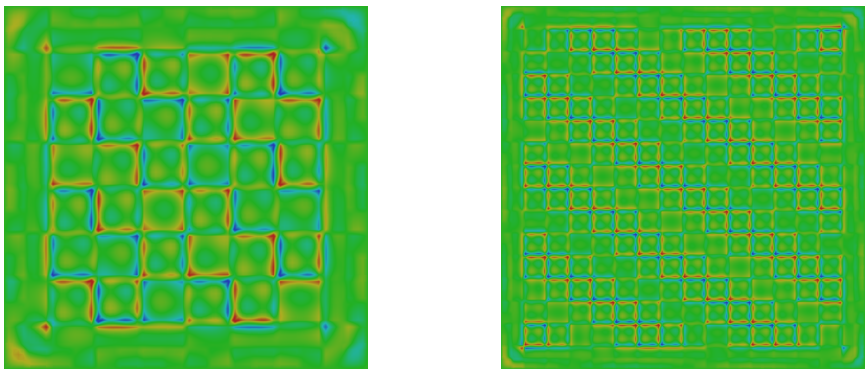


Figure 7.39. *Contour plots of the coarse grid projection error $E_l\tilde{u}_l$ of (7.23) with $l = 4$ (left) and $l = 5$ (right) using cubic polynomials.*

cubic local approximation spaces we can clearly observe that the error is much smaller close to the boundary. Thus, we expect the multilevel solver for the conforming approach with cubic spaces $V_{i,m}$ to converge somewhat faster than for the non-conforming Nitsche approach.

The measured results attained for a Nitsche discretization and a conforming discretization with cubic polynomials are collected in the Figures 7.40 and 7.41, and the Figures 7.42 and 7.43, respectively. Comparing these results we see that the multilevel solver for the conforming approach in fact yields better convergence rates than for the Nitsche method. For the non-conforming discretization we find the rates 0.45, 0.08, and 0.015 for the $V(s,s)$-cycles with $s = 1, 3, 5$ respectively and the preconditioned CG method reduces the iteration error by a factor of 0.075. The convergence rates attained for the conforming approach are even better. We measure the rates 0.25, 0.02, and 0.007 for the $V(s,s)$-cycles with $s = 1, 3, 5$ respectively and our default solver converges with a rate of 0.05.

In summary, the presented results clearly confirm that our multilevel PPUM solvers yield a level-independent convergence behavior independent of the type of boundary conditions. In the case of Dirichlet boundary conditions we may obtain faster convergence of the solver for the Nitsche approach or for the conforming boundary treatment. We assume that the geometry of the boundary, the employed local approximation spaces, and even the solution itself may have an impact on the specifics of the convergence behavior, i.e. on the constants.

Note that the arguments of the above discussion are valid in a more general situation. To this end, we consider a sequence of uniformly hp-refined PPUM spaces (actually we alternate uniform global h-refinement and global hp-refinement). With this refinement scheme the local approximation spaces on coarser levels $V_{i,l}$ have a coarser polynomial resolution than the finer spaces $V_{i,l+1}$ such that the restriction and prolongation operators again should have an additional filtering property which leads to a smoother coarse grid approximation and correction. Thus, we expect that the convergence rate of our multilevel solvers are very much robust with respect to the polynomial degree due to this additional smoothing effect of the transfer operators. The results obtained for this experiment are summarized in Figures 7.44 and 7.45. Observe that the convergence behavior is again level-independent. Since we coupled the polynomial degree to the refinement level, the convergence behavior is also independent of the polynomial degree. From the line plots showing the convergence history for the iterative correction depicted in Figure 7.44 we can clearly see that a single iteration of our nested iteration solver is again sufficient to obtain an approximate solution within discretization accuracy. Note however that due to the global hp-refinement there are huge jumps in the improvement of the discretization errors from level to level (e.g. three orders of magnitude from level 7 to level 8). Hence, it is really surprising that a

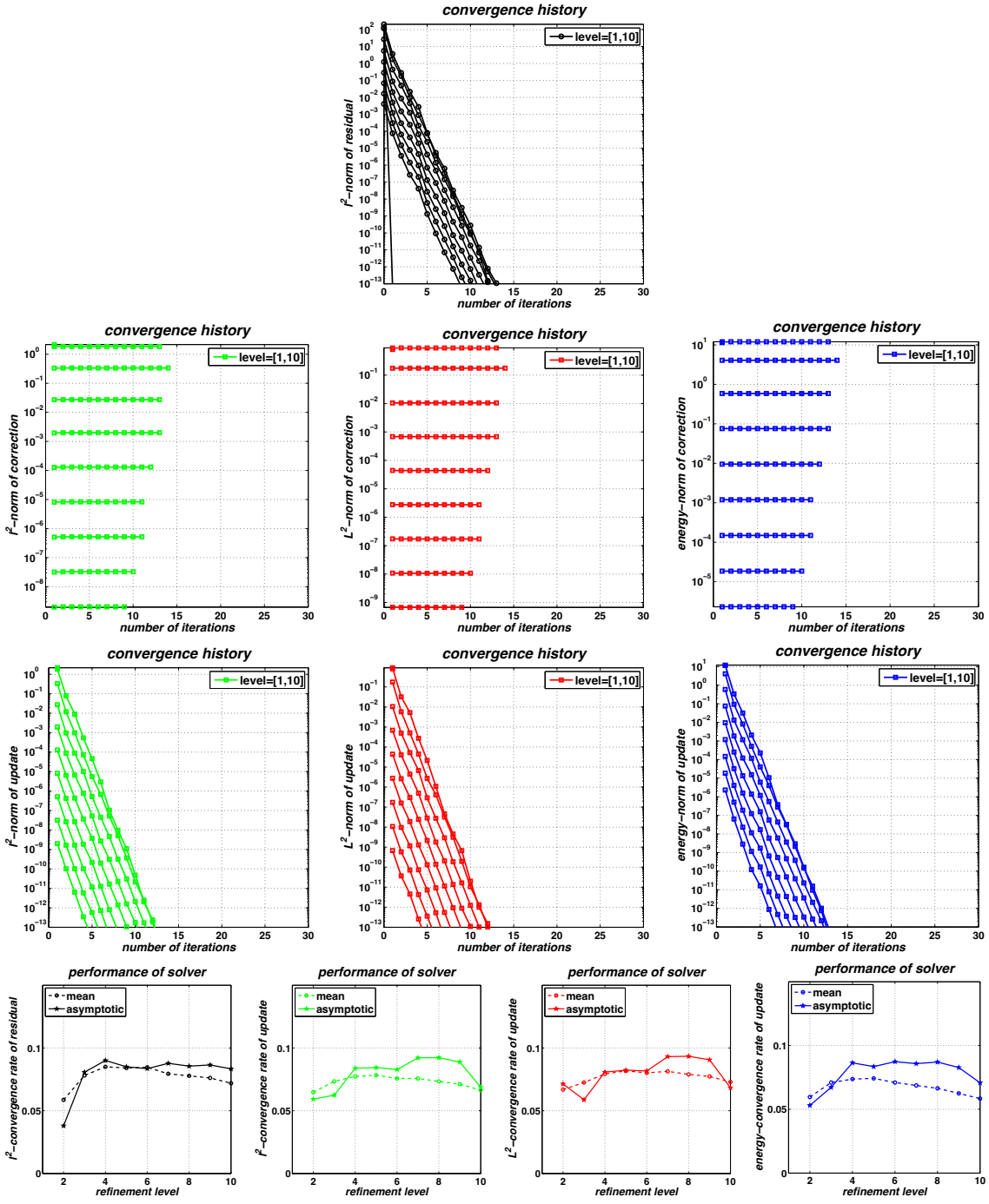


Figure 7.40. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 0$ and Dirichlet boundary conditions using Nitsche's method with the optimal regularization parameter and uniformly h-refined PPUM spaces V_l^{PU} with cubic local approximation spaces $\mathcal{P}^3$.*

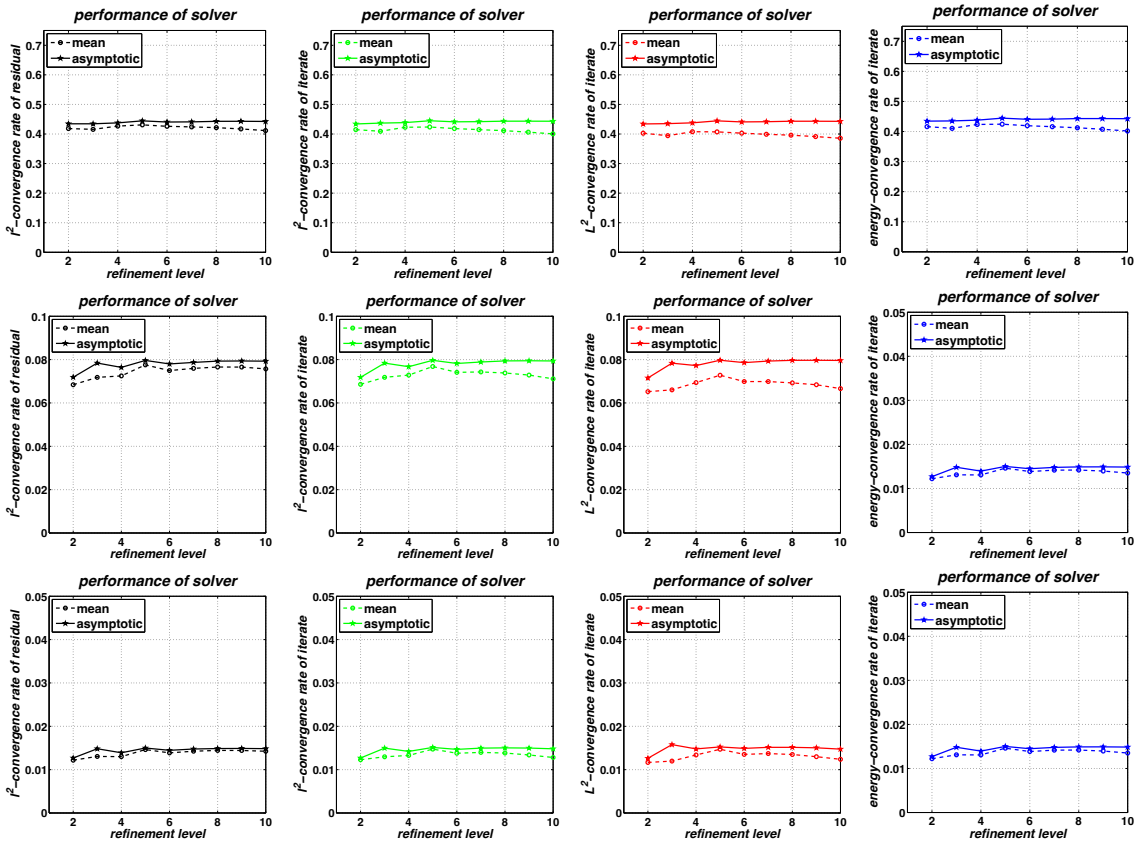


Figure 7.41. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 0$ and Dirichlet boundary conditions using Nitsche's method with the optimal regularization parameter using uniformly h-refined PPUM spaces V_l^{PU} with cubic local approximation spaces $\mathcal{P}^3$.*

single iteration of the solver is already sufficient. The convergence rates of our solvers given in Figures 7.44, and 7.45 are very stable up to level 8. Then, we find a larger deviation of the average and asymptotic convergence rates which is really just a rounding effect (another indication is of course that the asymptotic rates are better than the average rates which is usually not the case). To see this, we must consider the iterative update $\tilde{h}$ and its convergence behavior, see Figure 7.44. As we have already seen, the norms of the iterative correction $\tilde{c}$ decrease substantially from level to level. Thus, the norms of the iterative updates $\tilde{h}$ are very small absolute numbers. On level 9 the l^2-norm of the initial update and of the correction is of the order 10^{-10} and after 5 iterations the l^2-norm of the update is certainly below machine accuracy. Therefore, it is very likely that rounding effects can spoil the computation of norms at later iterations to some extent so that we cannot compute the convergence rate reliably. In essence, the iterations should have been stopped once the l^2-norm of the iterative update relative to the l^2-norm of the iterate is below machine accuracy.

We close this example with the convergence results of our multilevel solver for a three dimensional model problem (7.1) with $c = 0$ and Dirichlet boundary conditions implemented by the conforming local subspaces of $V_{i,l} = \mathcal{P}^1$. From the plots depicted in Figures 7.46 and 7.47 we essentially read of the same convergence behavior as in the two dimensional case discussed in detail above. The standard solver attains a rate around 0.15 and the stand-alone $V(s,s)$-cycles

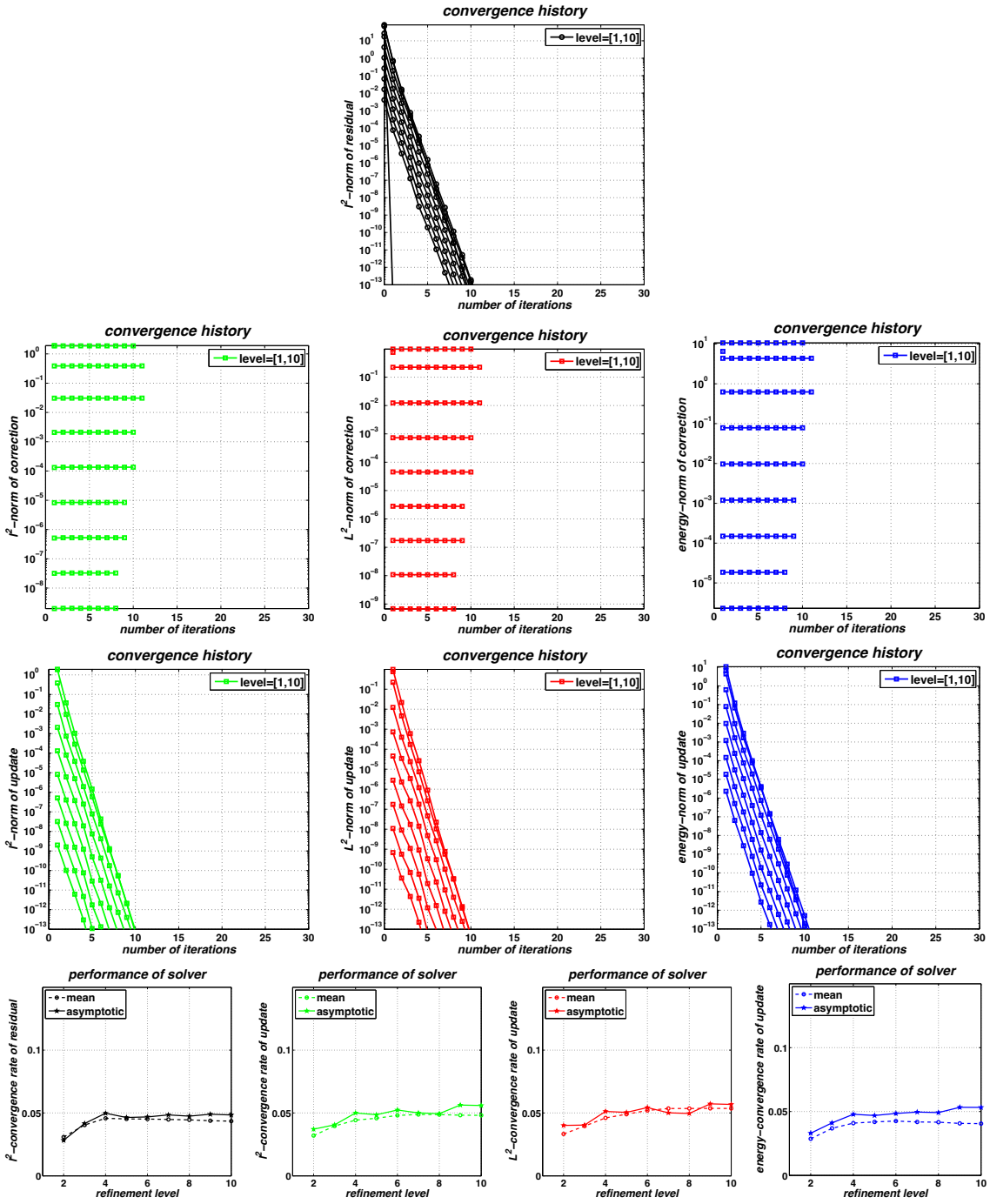


Figure 7.42. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 0$ and Dirichlet boundary conditions using uniformly h-refined PPUM spaces V_l^{PU} with the conforming subspaces of $\mathcal{P}^3$ as local approximation spaces.*

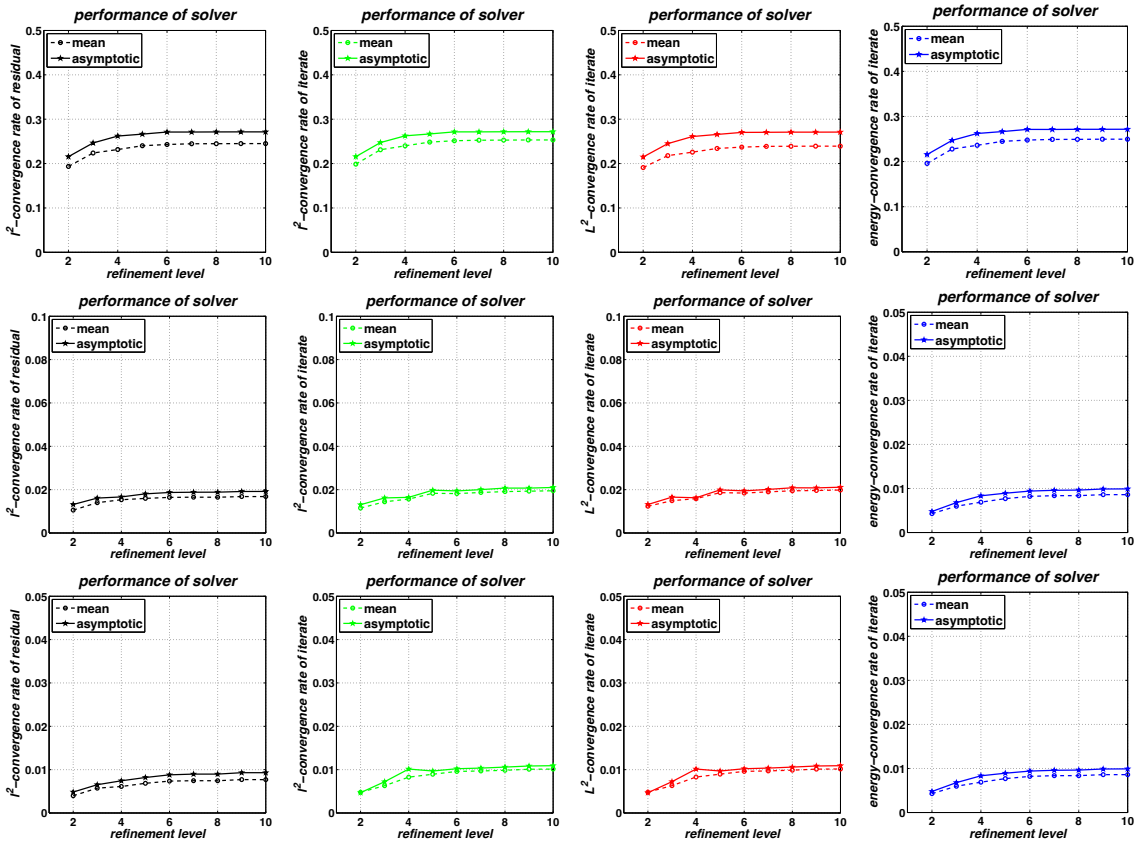


Figure 7.43. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 0$ and Dirichlet boundary conditions using uniformly h-refined PPUM spaces V_l^{PU} with the conforming subspaces of $\mathcal{P}^3$ as local approximation spaces.*

yield reduction factors of 0.6, 0.35, and 0.25 with $s = 1,3,5$ respectively.

Example 7.14. In the next example we consider the solution of linear systems arising from the PPUM discretization of systems of PDEs (7.2) on the domain $(-1,1)^D$ with $D = 2,3$. The Dirichlet boundary is given as

$$\Gamma_D := \{x \in [-1,1]^D \mid x_d = -1 \text{ for at least one } d = 1,\dots,D\}$$

and we employ linear polynomials as local approximation spaces. The essential boundary conditions are realized via the construction of conforming subspaces.

In Figure 7.48 the convergence history for the preconditioned CG solver applied to our PPUM discretization of (7.2) in two dimensions is depicted. We can clearly observe a level-independent convergence behavior from these plots. The asymptotic error reduction per iteration is about 0.2 which corresponds very well to the rates obtain for a conforming discretization of the scalar model problem. However the measured convergence rates for the stand-alone multilevel iterations are slightly worse than in the scalar case, see Figure 7.49. Here, multiple smoothing steps do not yield the same improvement in the convergence rate. We find the rates to be no better than 0.4 for the $V(3,3)$- and the $V(5,5)$-cycles and about 0.6 for the $V(1,1)$-cycle. Since we obtain very

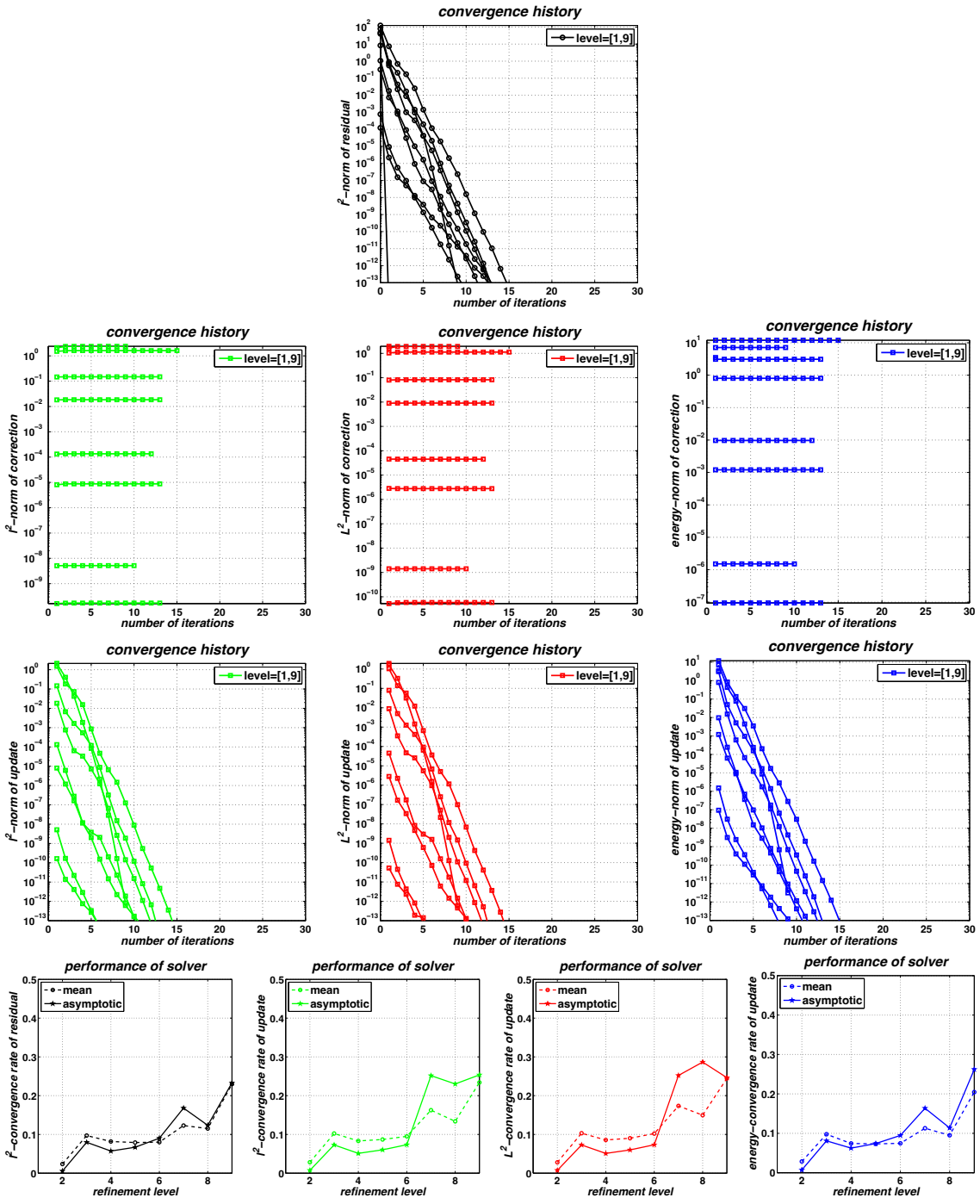


Figure 7.44. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 0$ and Dirichlet boundary conditions using uniformly h-refined PPUM spaces V_l^{PU} with the conforming subspaces of $\mathcal{P}^{p_l}$ with $p_l = 1,1,2,2,3,3,4,4,5$ for $l = 1,\ldots,9$ as local approximation spaces.*

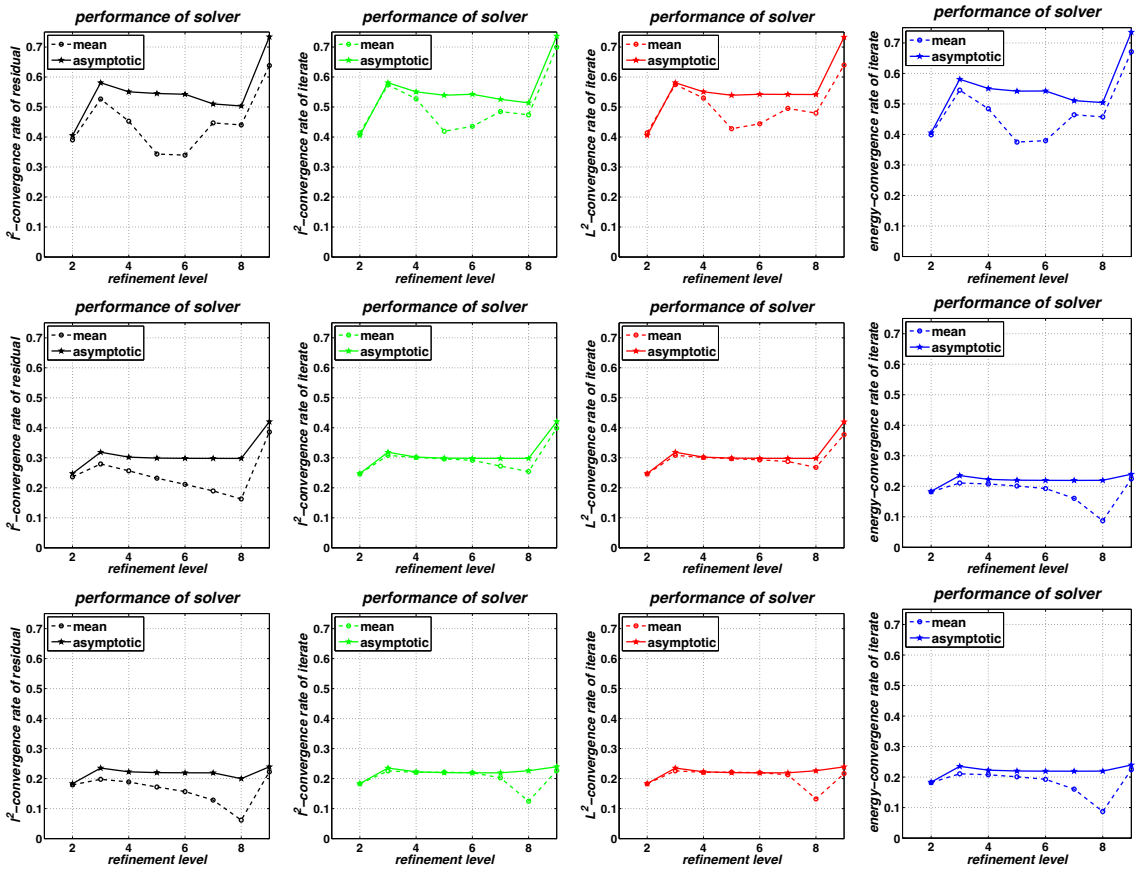


Figure 7.45. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to the discretization of (7.1) with $c = 0$ and Dirichlet boundary conditions using uniformly h-refined PPUM spaces V_l^{PU} with the conforming subspaces of $\mathcal{P}^{p_l}$ with $p_l = 1,1,2,2,3,3,4,4,5$ for $l = 1,\ldots,9$ as local approximation spaces.*

similar results also for a non-conforming discretization we assume that the smoothing property of our block-Gauss–Seidel iteration which operates on all physical variables simultaneously is somewhat reduced in the systems case.

The results obtained for the three-dimensional problem are collected in Figures 7.50 and 7.51. Here, we find essentially the same convergence behavior as in the two-dimensional case. Now the rates are about 0.25 for our default solver, and the $V(s,s)$-cycles yield the error reduction factors 0.65, 0.55, and 0.5 with $s = 1,3,5$ respectively.

Example 7.15. Let us now turn to problems with singular solutions. First we consider (7.2) on an L-shaped domain with $c = 0$ and homogeneous Dirichlet boundary conditions on Γ_D of (7.7) and Neumann boundary conditions on $\partial\Omega \setminus \Gamma_D$. Here, we employ a uniform h-refinement and use linear polynomials as local approximation spaces.

Note that the low-order regularity of the solution (7.8) in general causes (at least) a substantial increase in the convergence rates of classical multigrid methods [141]. Thus we should expect to find larger convergence rates in this example than for Example 7.12.

The results obtained for a non-conforming discretization with minimal regularization are given in Figures 7.52 and 7.53. It is somewhat surprising that the observed convergence rates are

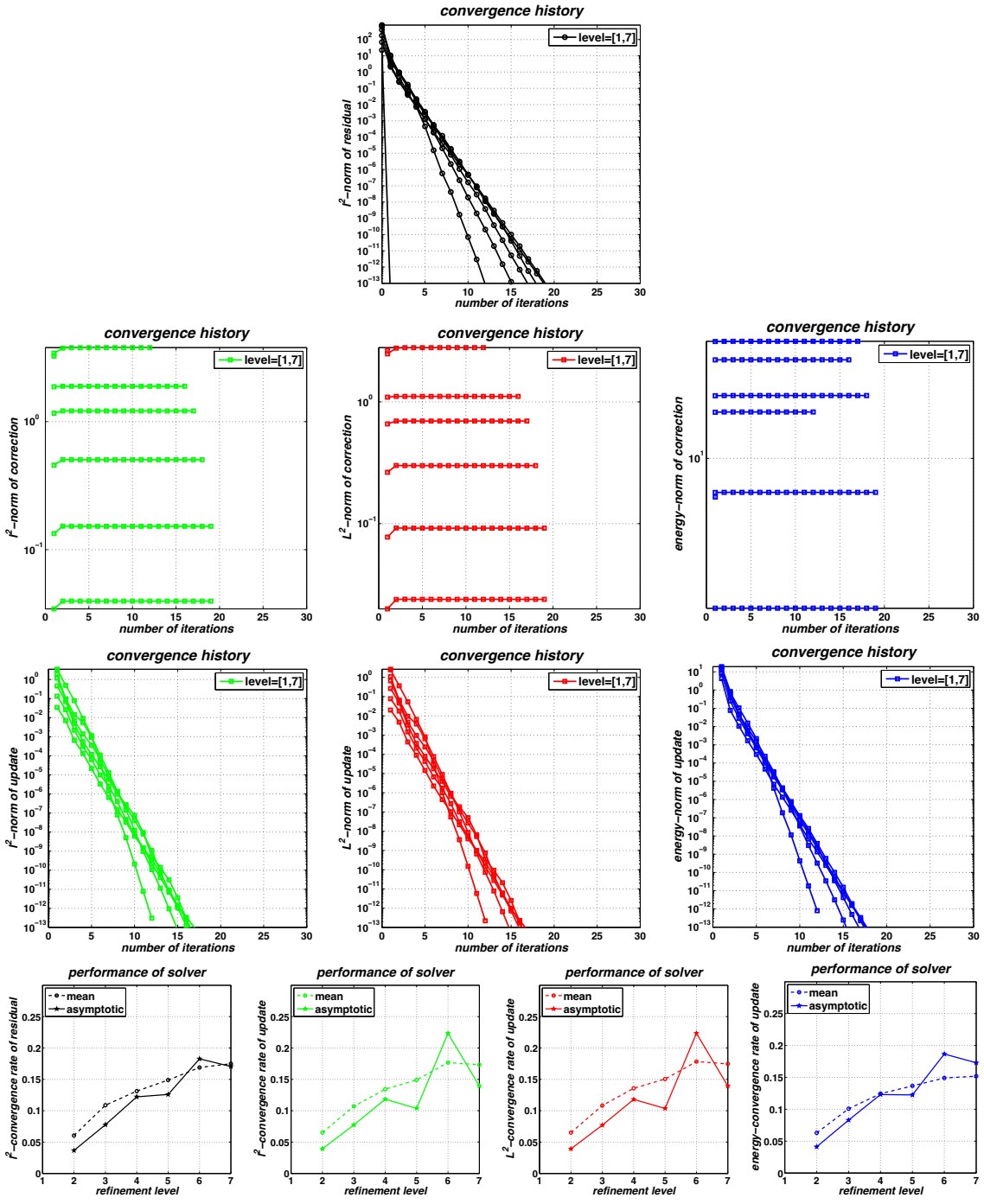


Figure 7.46. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to the discretization of (7.1) in three dimensions with $c = 0$ and Dirichlet boundary conditions using uniformly h-refined PPUM spaces V_l^{PU} with the conforming subspaces of $\mathcal{P}^1$ as local approximation spaces.*

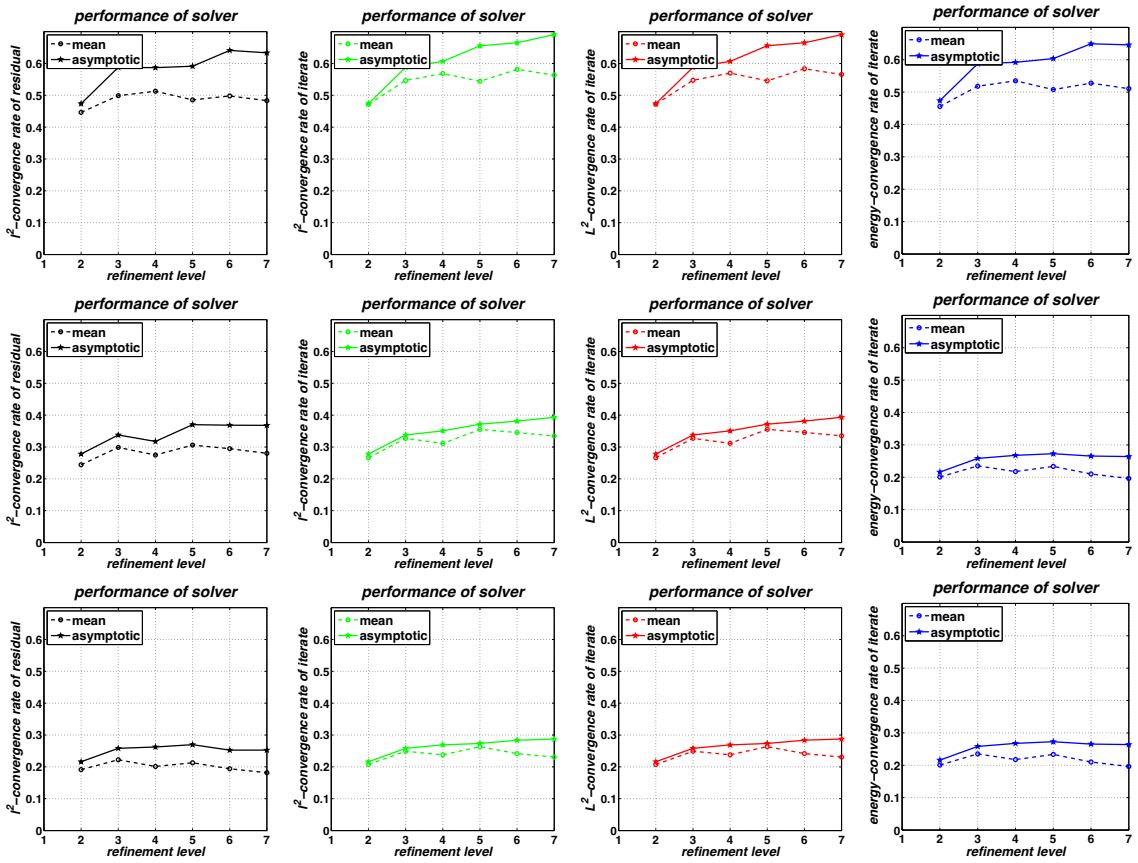


Figure 7.47. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to the discretization of (7.1) in three dimensions with $c = 0$ and Dirichlet boundary conditions using uniformly h-refined PPUM spaces V_l^{PU} with the conforming subspaces of $\mathcal{P}^1$ as local approximation spaces.*

almost identical to those of Example 7.12. Yet, it becomes obvious from the following argument. Recall that the error distribution obtained by Nitsche's method with minimal regularization is quite different from a conforming discretization, see Figure 7.5. The error is essentially shifted onto the boundary due to the fact that the boundary conditions are enforced as weakly as possible without breaking the definiteness of the stiffness matrix. Thus, the discrete problem is in some sense of higher regularity than the analytical problem. In essence the minimal regularization parameter in Nitsche's method (which controls the definiteness of the stiffness matrix) yields the maximal regularization of the approximate solution (with respect to its smoothness). Thus, our multilevel solver in fact operates on a sequence of problems that are more regular than the analytical problem. This allows the employed smoother to shift the error onto the Dirichlet boundary and on coarser levels these error components carry less energy due to the smaller regularization parameter. Furthermore, our transfer operators do not enforce boundary conditions explicitly. Hence, the use of a minimal regularization parameter in Nitsche's method preserves the quality of our smoother and coarse grid correction.

If we, however, use a large regularization (or even just a constant regularization on all levels, i.e. the regularization of the finest level) the convergence behavior of our solver is adversely

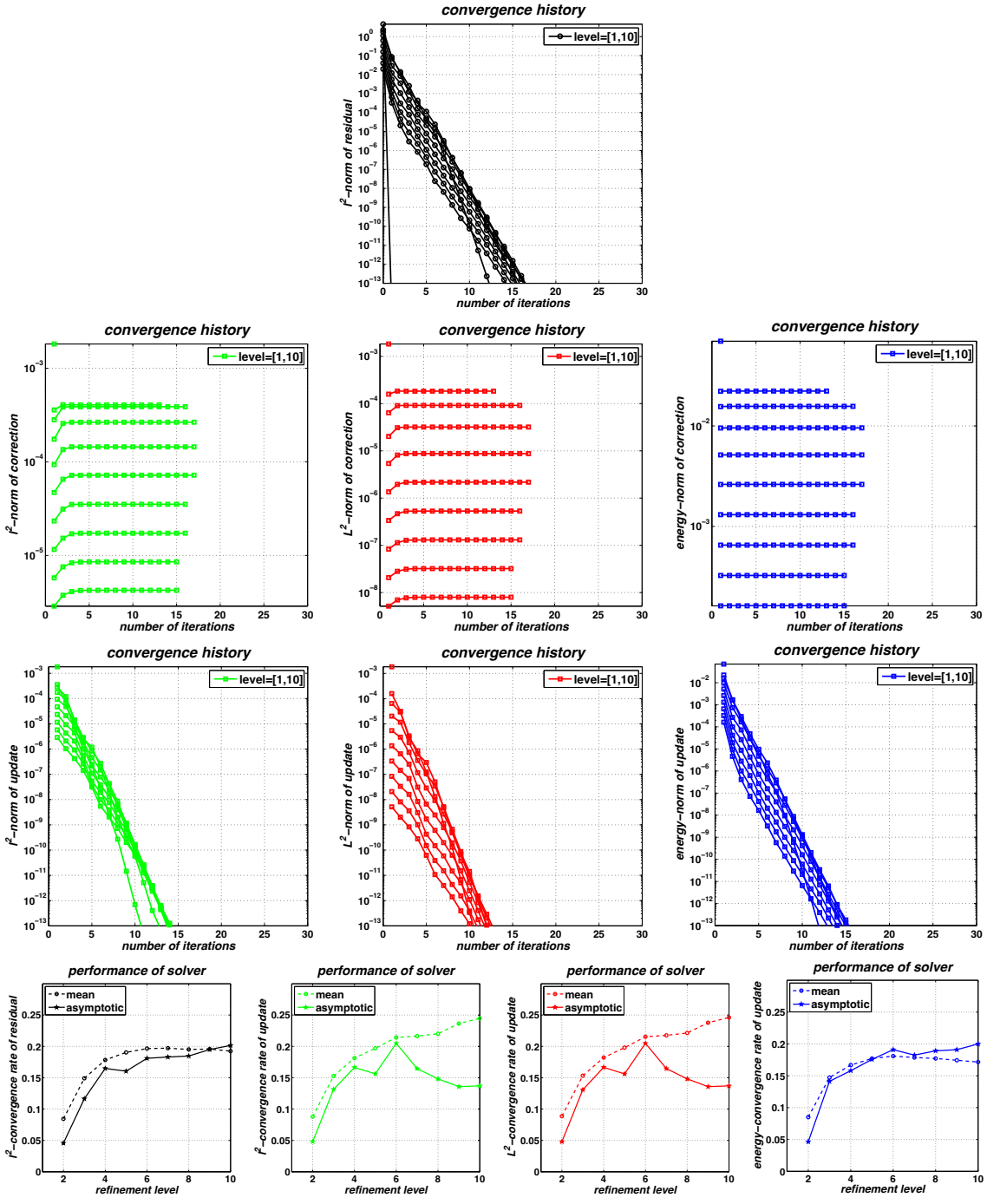


Figure 7.48. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to the discretization of (7.2) in two dimensions using $\mathcal{P}^1$ as local approximation spaces.*

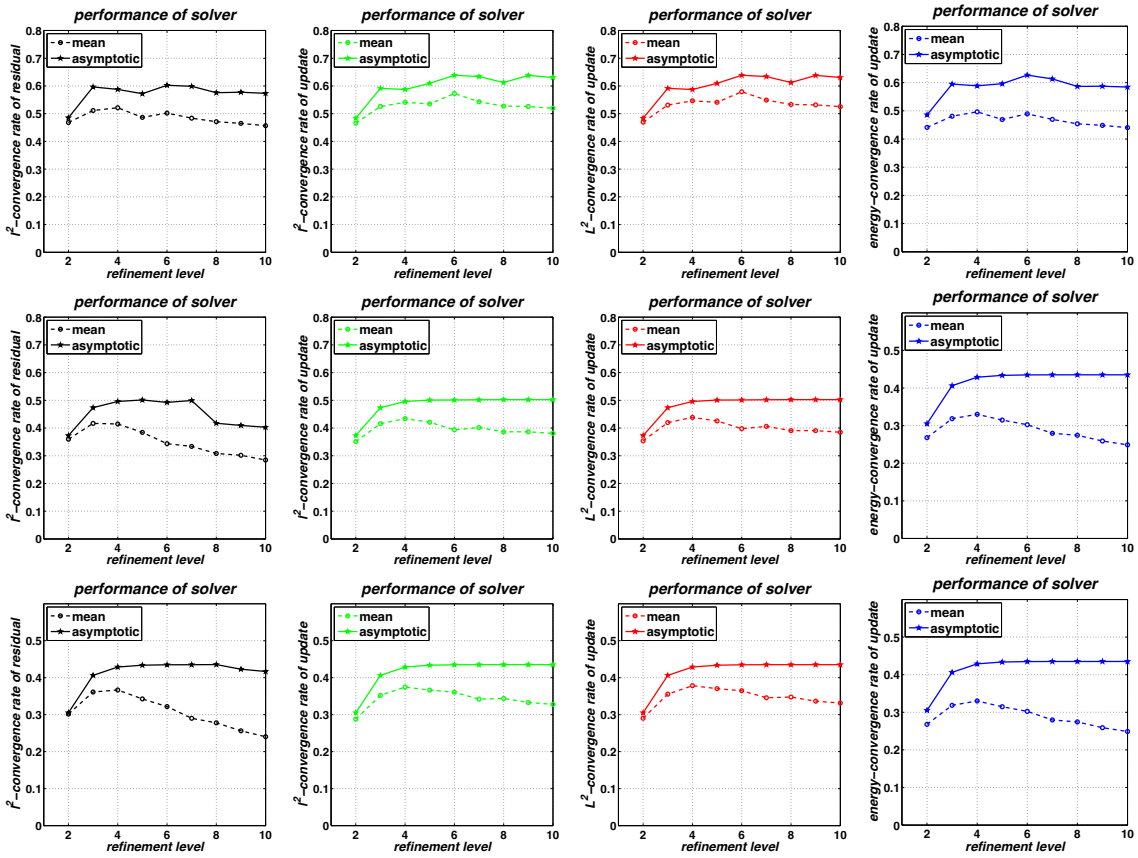


Figure 7.49. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to the conforming discretization of (7.2) in two dimensions using $\mathcal{P}^1$ as local approximation spaces.*

affected. Since the boundary value is (more) strictly enforced the error cannot be shifted to the boundary so that especially on coarser levels the error is distributed inside the domain and cannot be reduced efficiently by the smoother, compare Figure 7.5. Therefore, we find the anticipated larger convergence rates of our solvers for the conforming discretization, see Figures 7.54 and 7.55. Yet, the convergence rates are still level-independent (which is in general not the case for classical multigrid methods [141]). The $V(s,s)$-cycles yield rates not better than 0.75 with $s = 1,3,5$ which is larger than the improvement of the discretization error due to refinement so that also our default nested iteration solver requires more iterations per level, see Figure 7.54. From the convergence history of the iterative correction $\tilde{c}$ and the update $\tilde{h}$ we see that up to 7 iterations are needed to reach the asymptotic convergence regime (with respect to the L^2-norm). Thus the measured convergence rates grow slightly up to 0.25 for level $J = 11$ yet these rates will level off since they are certainly bounded by the rates of the $V(3,3)$-cycle which attains a rate of 0.8.

The convergence behavior of our multilevel solvers for a conforming discretization on the non-convex domain with re-entrant corner and an elliptic hole of Example 7.5 is almost identical, see Figures 7.56 and 7.57.

Example 7.16. If the singularities of the solution are however resolved by adaptive refinement, we expect to find similar convergence rates of our multilevel solvers as for problems with regular

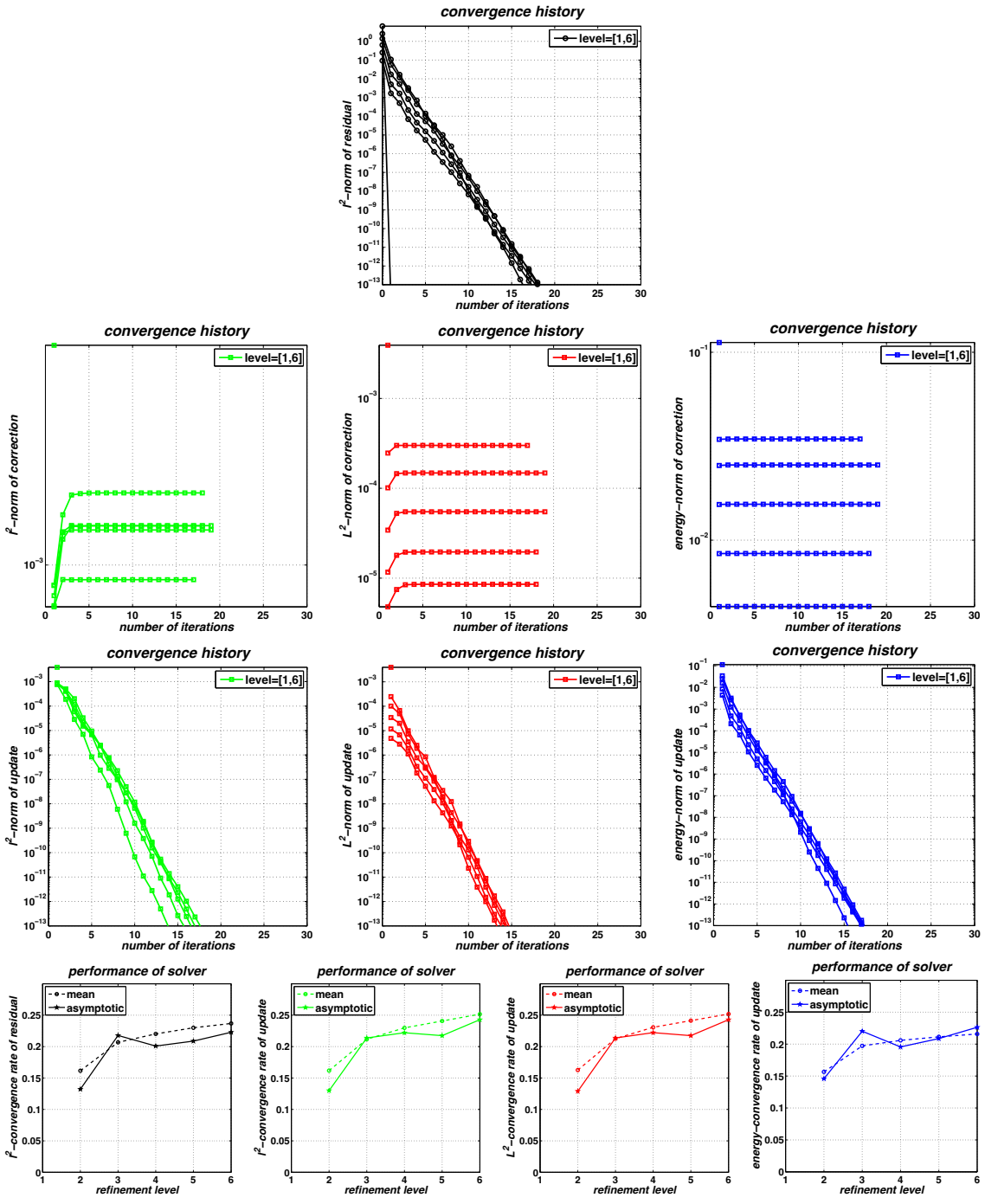


Figure 7.50. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to the discretization of (7.2) in three dimensions using $\mathcal{P}^1$ as local approximation spaces.*

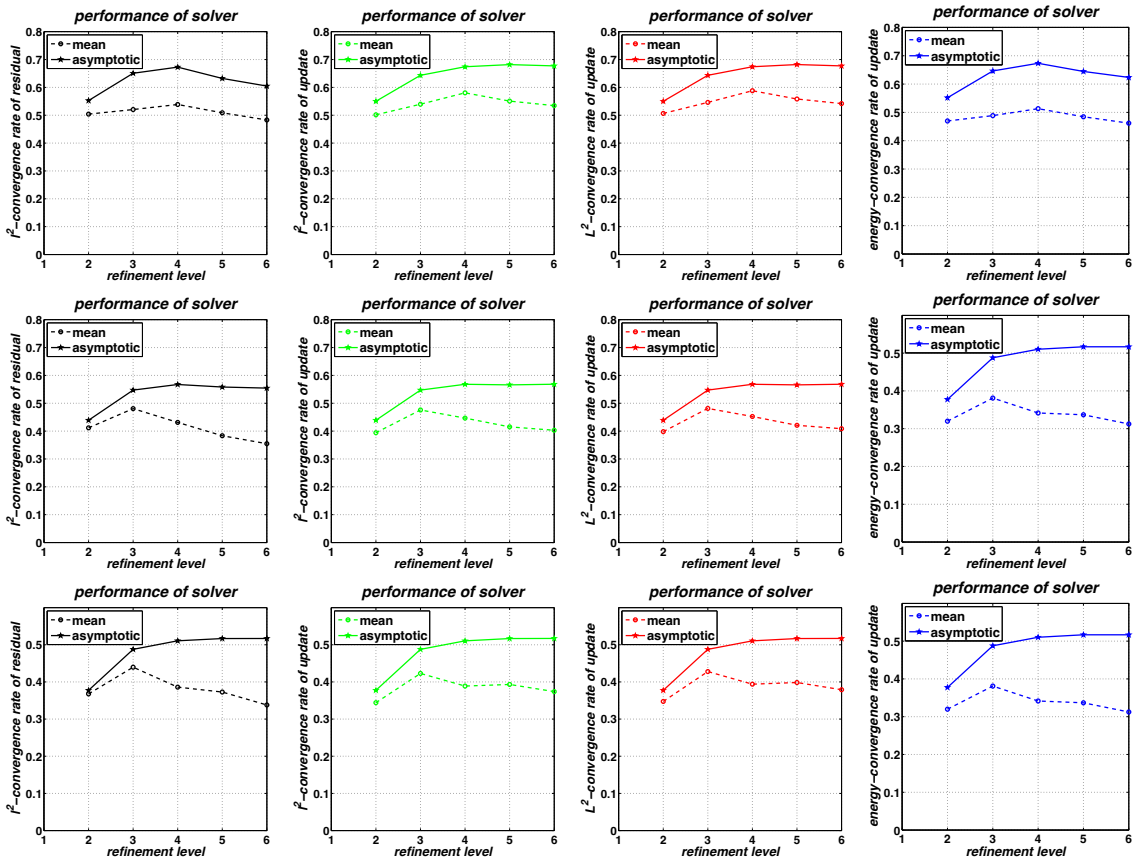


Figure 7.51. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to the conforming discretization of (7.2) in three dimensions using $\mathcal{P}^1$ as local approximation spaces.*

solutions. The results obtained for an hp-adaptive conforming discretization on the L-shaped domain are displayed in Figure 7.58. Here, we find a convergence rate around 0.15 for our nested iteration approach which agree well with those obtained for a uniform hp-refinement, compare Figures 7.44. These plots clearly indicate that a single iteration on each level is sufficient to attain an approximate solution with discretization accuracy.

The convergence behavior of our multilevel solver for an hp-adaptive conforming discretization on the complicated domain of Example 7.6, see Figure 7.15, depicted in Figure 7.59 is also very similar with a rate of about 0.1.

Example 7.17. Our hierarchical enrichment scheme also resolves singular behavior of the solution. Thus a uniform h-refinement with hierarchical enrichment should yield a similar convergence behavior of our multilevel solver as for the corresponding regular problem.

In Figures 7.60 and 7.61 we give the results obtained for the linear elastic fracture mechanics problem 7.16. Observe that the rates agree very well with those of Example 7.3 where we considered a smooth elasticity problem. The preconditioned CG solver attains a rate of 0.2 and the $V(s,s)$-cycles converge with a rate around 0.6. Thus, the hierarchical enrichment with singular functions has no adverse effect on the convergence behavior of our multilevel solvers. The error reduction is independent of the number of levels, the enrichment functions and the regularity of

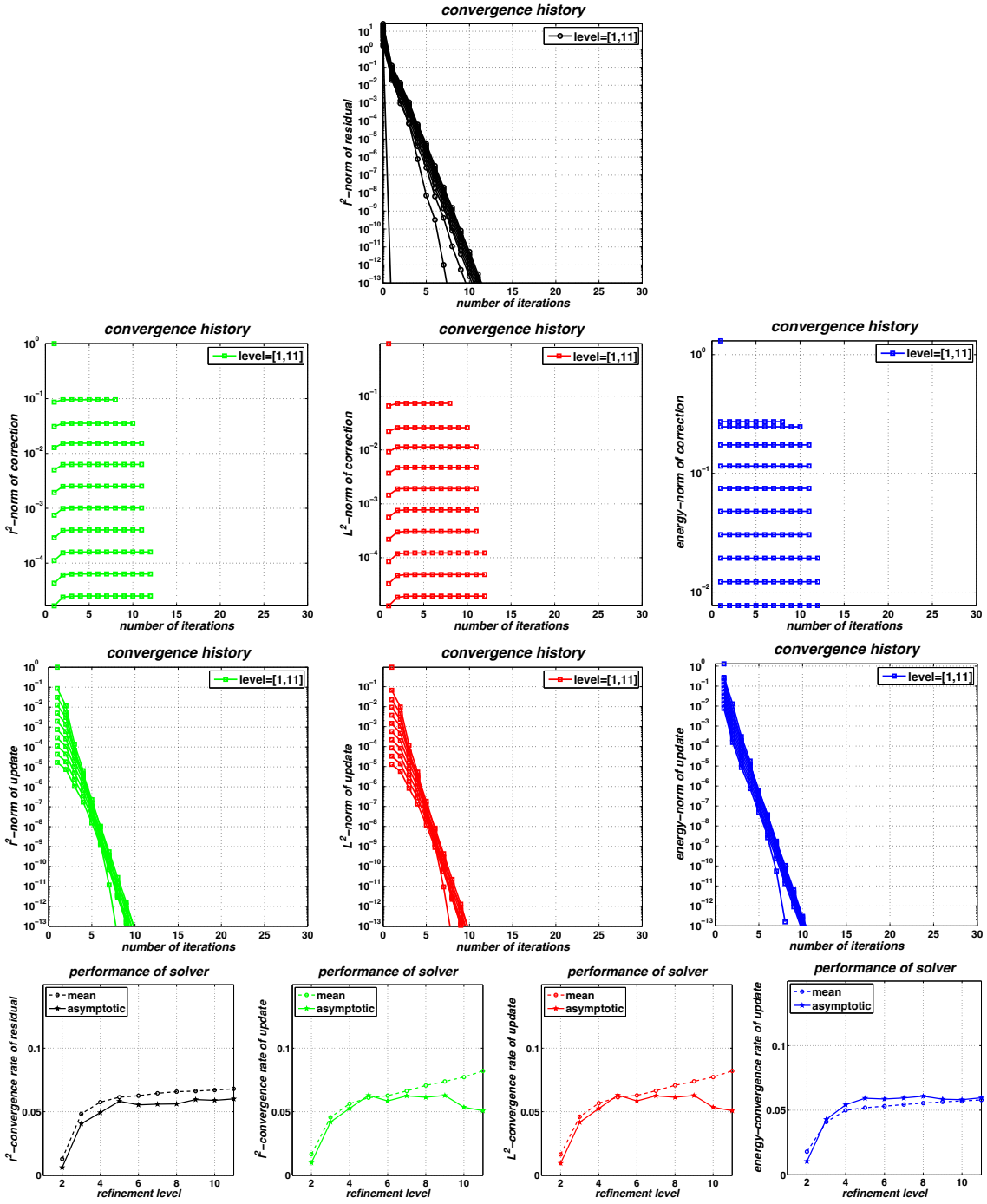


Figure 7.52. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to a non-conforming discretization of (7.2) on an L-shaped domain using $\mathcal{P}^1$ as local approximation spaces.*

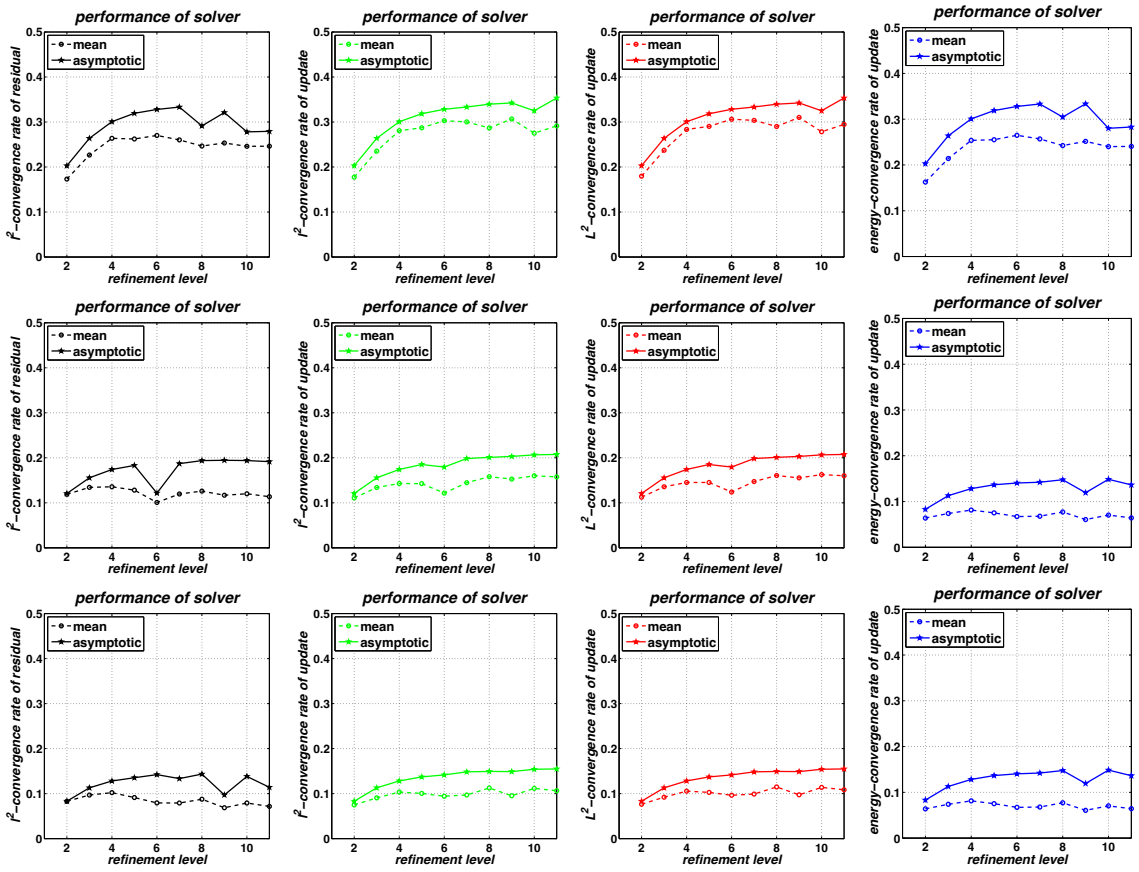


Figure 7.53. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to a non-conforming discretization of (7.2) on an L-shaped domain using $\mathcal{P}^1$ as local approximation spaces.*

the solution.

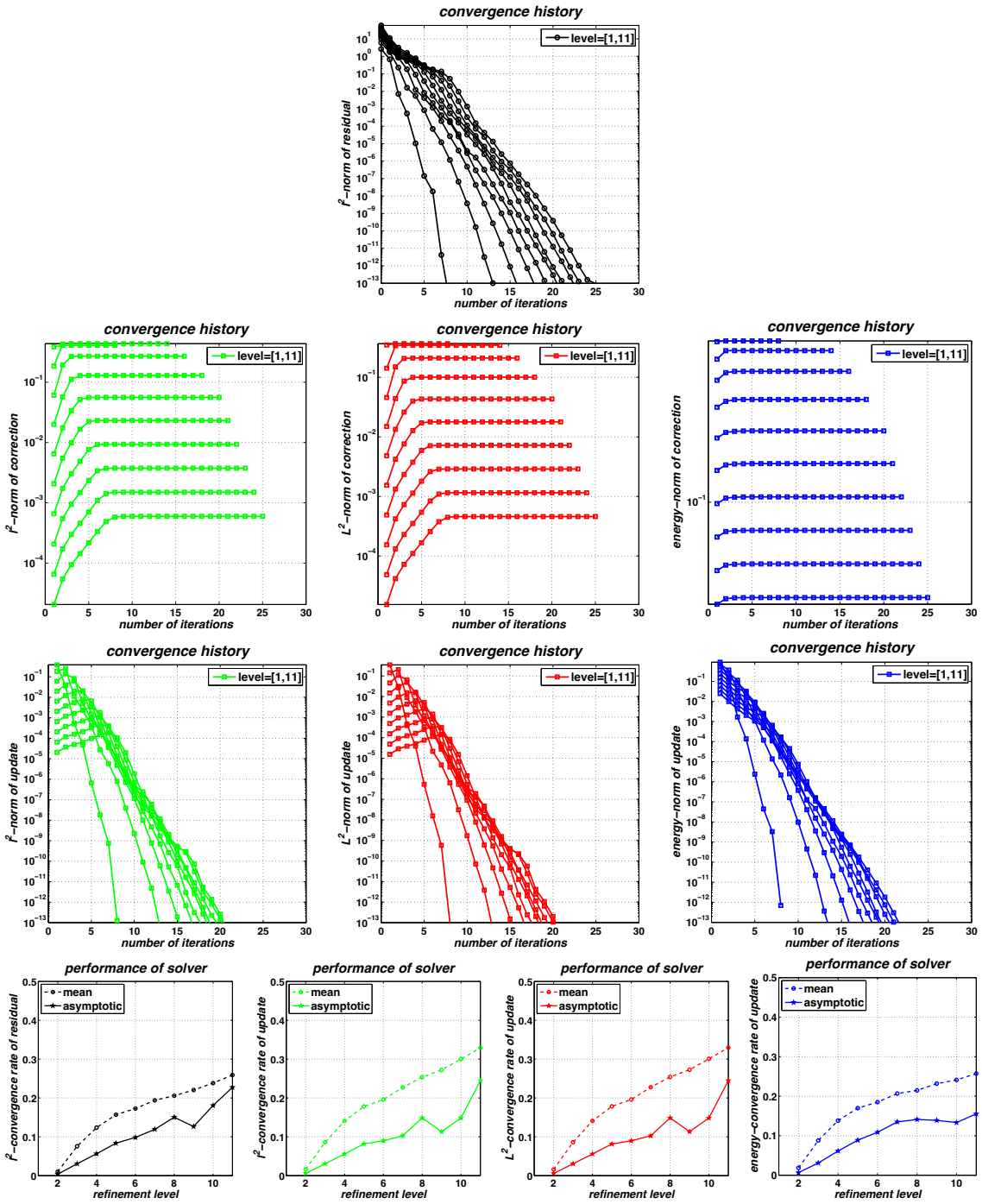


Figure 7.54. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to a conforming discretization of (7.2) on an L-shaped domain using $\mathcal{P}^1$ as local approximation spaces.*

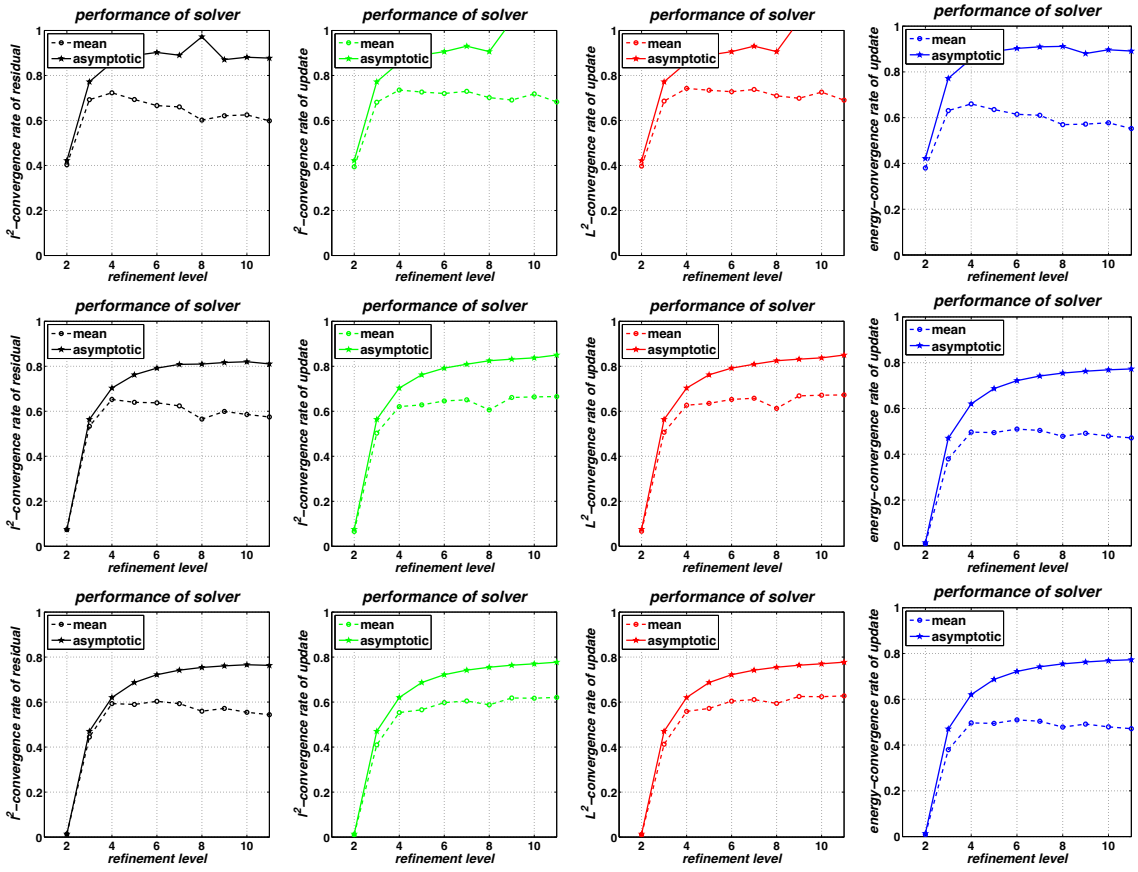


Figure 7.55. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to a conforming discretization of (7.2) on an L-shaped domain using $\mathcal{P}^1$ as local approximation spaces.*

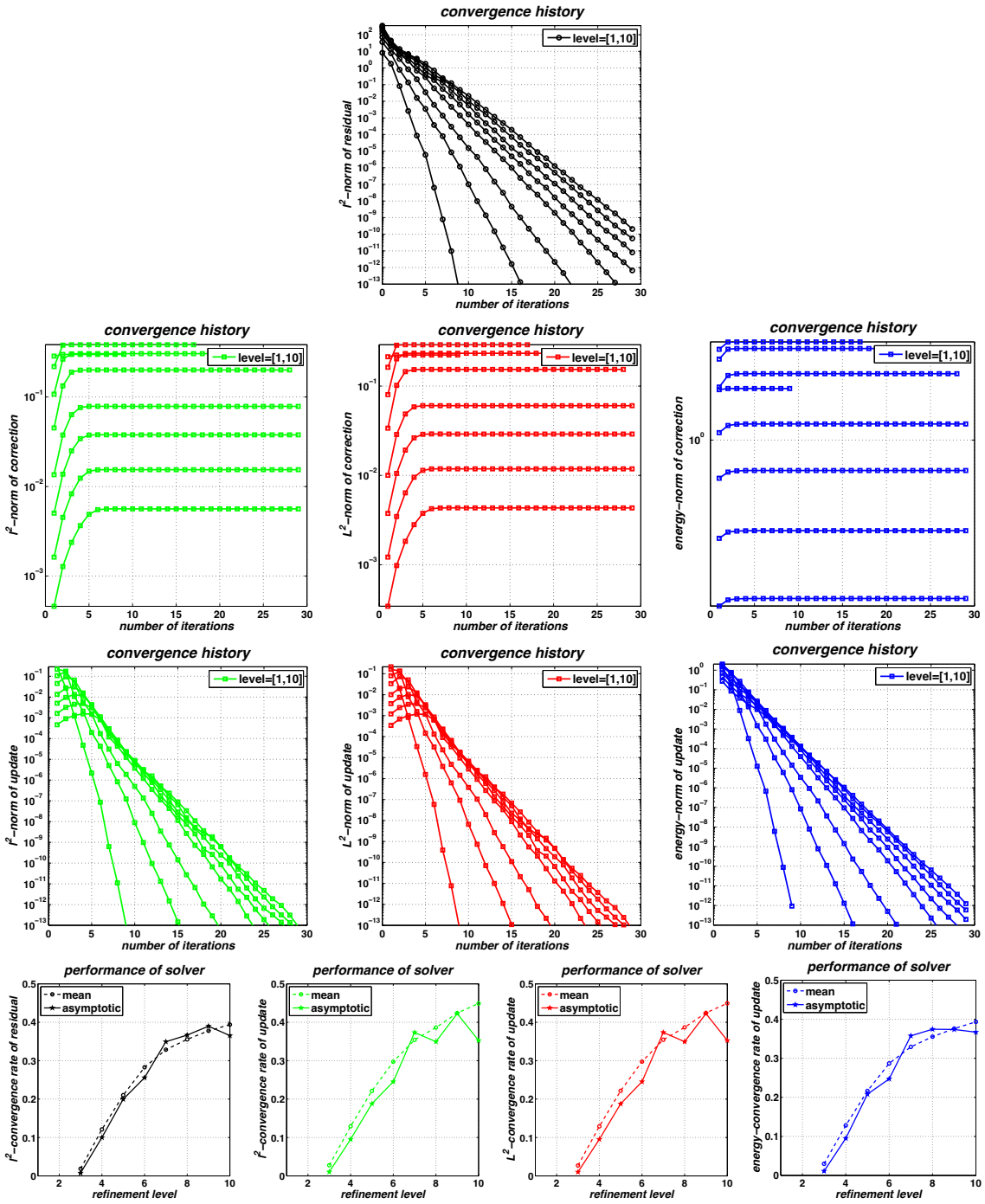


Figure 7.56. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to a conforming discretization of (7.2) on a non-convex domain with a re-entrant corner and an elliptic hole using $\mathcal{P}^1$ as local approximation spaces.*

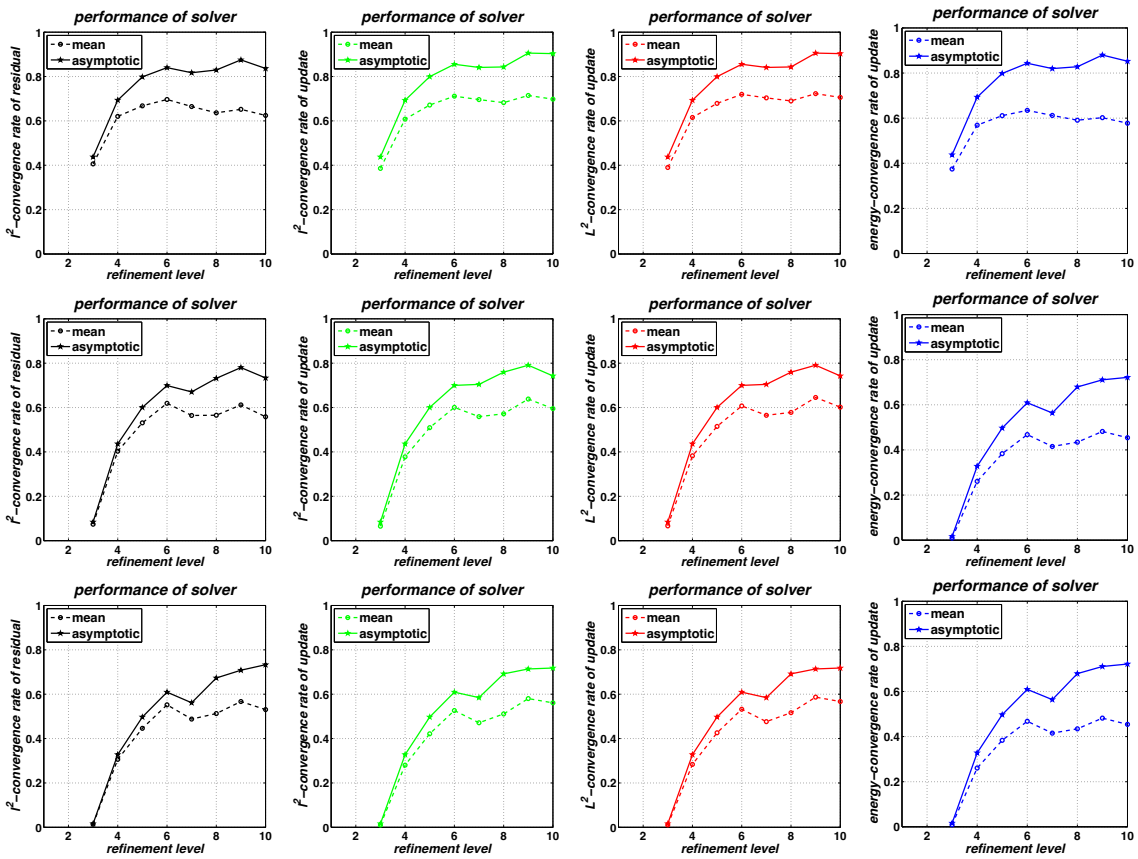


Figure 7.57. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to a conforming discretization of (7.2) on on a non-convex domain with a re-entrant corner and an elliptic hole using $\mathcal{P}^1$ as local approximation spaces.*

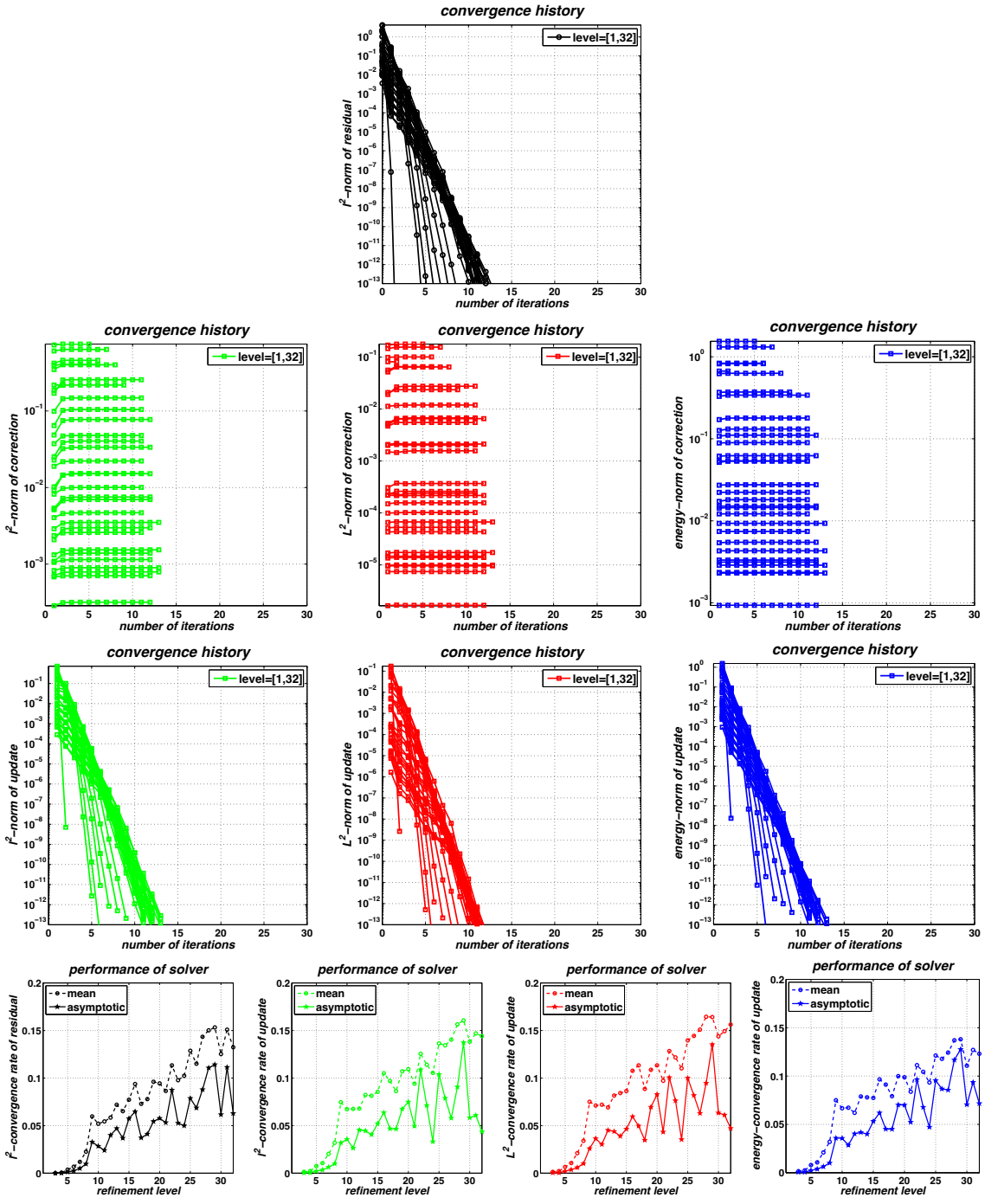


Figure 7.58. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to a conforming discretization of (7.2) on an L-shaped domain using adaptive hp-refinement.*

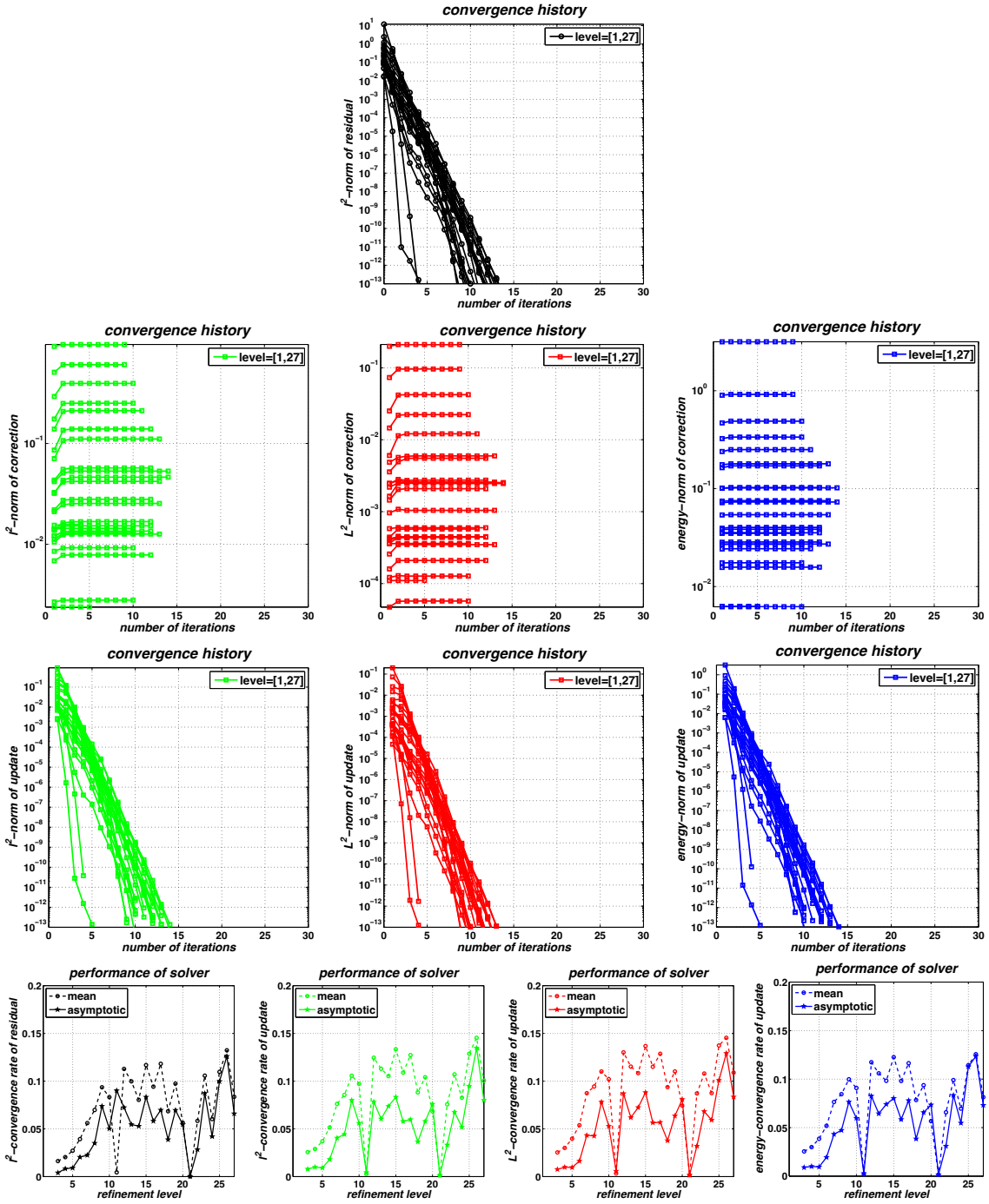


Figure 7.59. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to a conforming discretization of (7.2) on a non-convex domain with a re-entrant corner, a non-convex hole and oscillatory outer boundary using adaptive hp-refinement.*

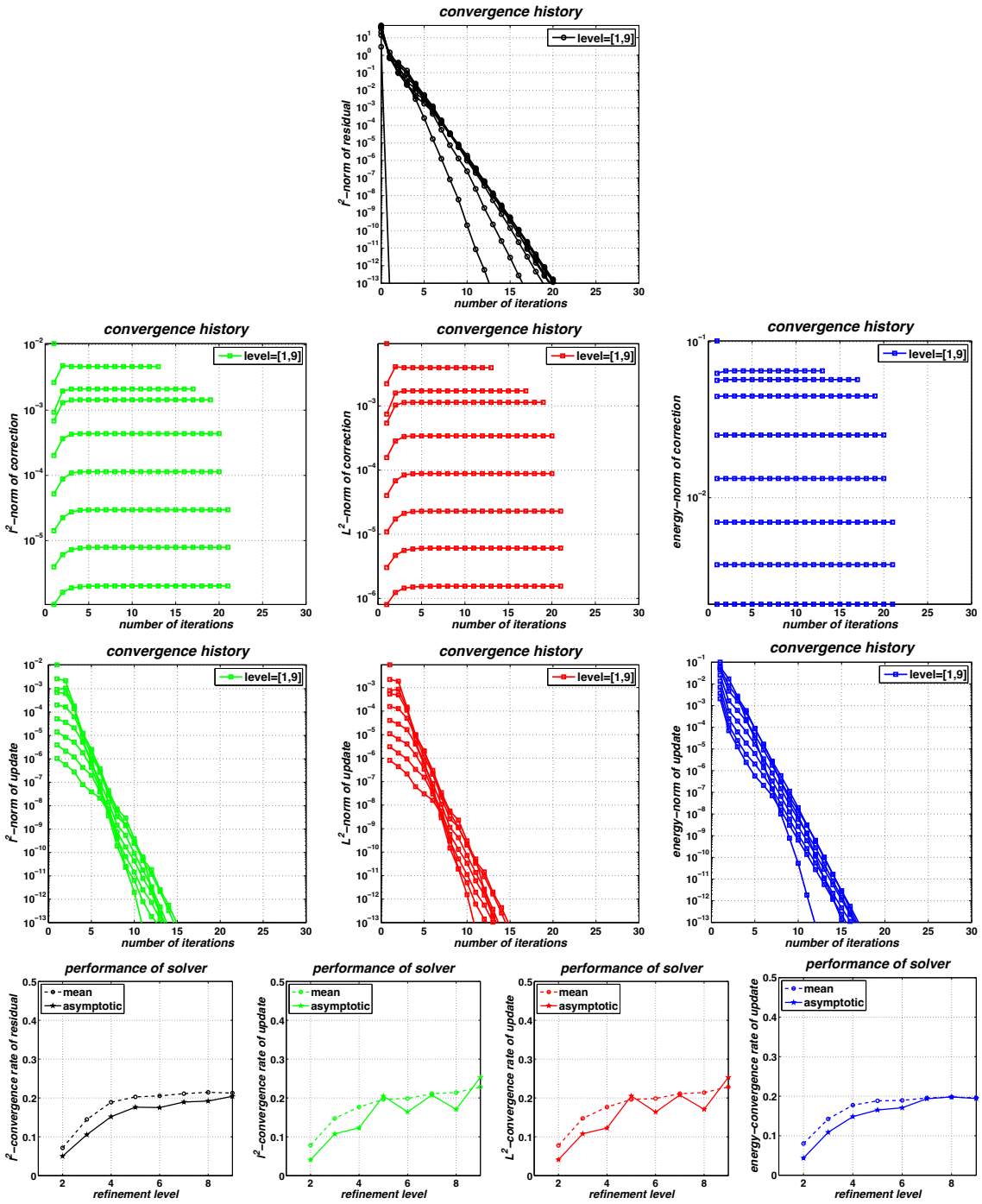


Figure 7.60. *Convergence history and rates for a nested iteration solver with a conjugate gradient method as inner solver preconditioned by a multilevel $V(2,2)$-cycle with block-Gauss–Seidel smoothing applied to a conforming discretization of (7.16) using uniform h-refinement and hierarchically enriched linear local approximation spaces.*

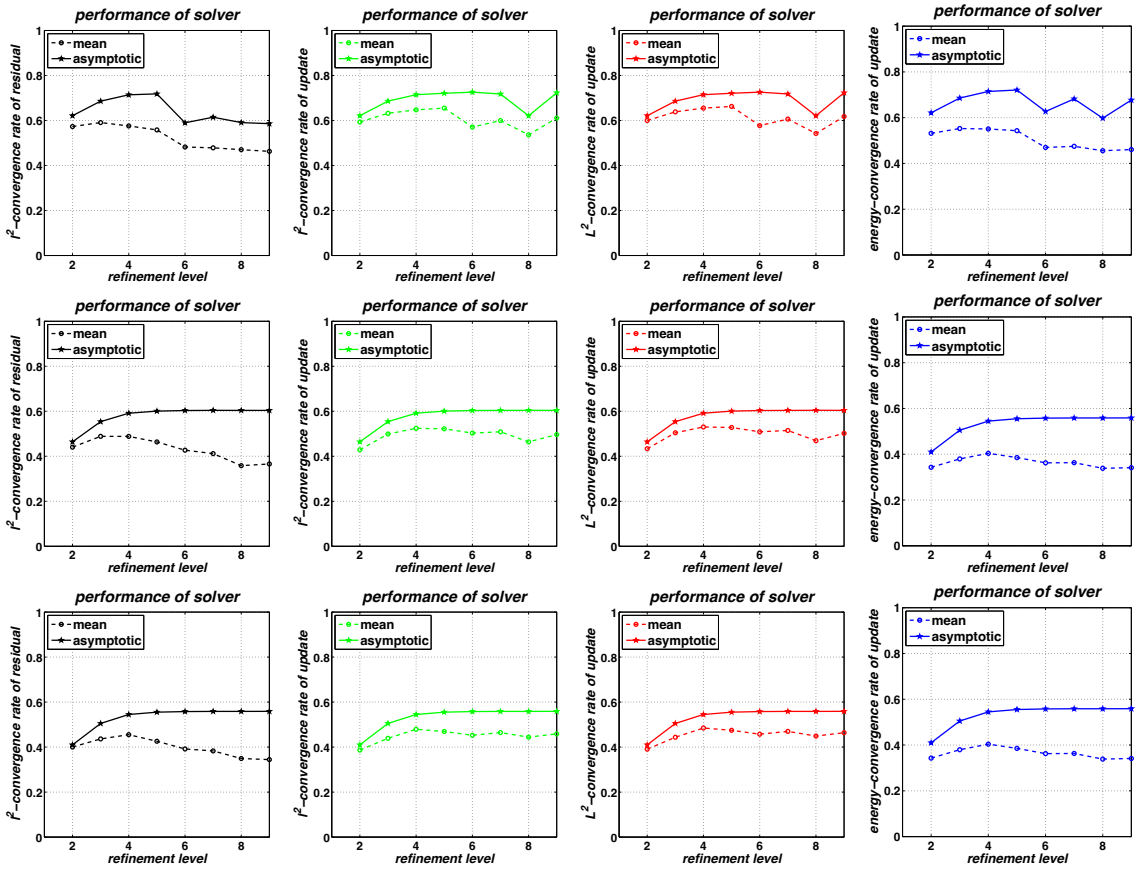


Figure 7.61. *Convergence rates for the multilevel $V(s,s)$-cycles with $s = 1,3,5$ (top to bottom row) and block-Gauss–Seidel smoothing applied to a conforming discretization of (7.16) using uniform h-refinement and hierarchically enriched linear local approximation spaces.*

Chapter 8
Concluding Remarks

In this manuscript we presented the meshfree generalization of the classical finite element method. To this end, we introduced two abstract methodical ingredients: The moving least squares method for the meshfree construction of a partition of unity and the partition of unity method which provides the framework to smoothly splice arbitrary local approximation spaces together.

Even though these two components yield optimal reproduction and approximation properties they must be complemented with the assumption of the flat top property of the partition of unity to attain a stable basis and thereby a computationally efficient method. The multilevel particle-partition of unity method (PPUM) developed in this manuscript employs all these ingredients. Hence, the method is applicable to arbitrary clouds of discretization points, supports the use of problem-dependent local approximation spaces, implements an automatic construction of a stable basis system, and comes equipped with a highly efficient multilevel solver.

Furthermore, we presented two fully automatic approaches to the notoriously cumbersome realization of essential boundary conditions in meshfree methods. The localized overlapping Nitsche formulation presented here is much more suited for adaptive discretizations than the global formulation employed previously [64, 125] where a slight deterioration of the convergence behavior of our multilevel solver was observed for adaptive discretizations [125]. With the overlapping approach we find no dependence of the convergence rate of our multilevel solver on the distribution of discretization points near the boundary. Yet, this non-conforming technique requires the analytical derivation of an appropriate weak form which somewhat hinders its speedy application to new problem classes. Furthermore, it is in some sense limited to stationary problems or implicit time-discretization schemes. Thus, there is a growing demand for a conforming boundary treatment. The availability of a conforming boundary treatment broadens the scope of our PPUM and simplifies the realization of new applications substantially.

In principle the construction of conforming approximation spaces is straightforward — just request a collection of conforming local approximation spaces as input. This approach however is practically irrelevant since it implies that the geometry of the domain and the approximation to the boundary is fixed on all discretization levels. Furthermore, this assumption denies us the automatic refinement and coarsening of the resulting PPUM space. Thus, the automatic construction of an efficient multilevel solver is not feasible. Here, we need to assume that on all levels of the discretization we are given appropriate conforming local approximation spaces. Clearly, this approach is asking for too much detail from the user. Especially for general domains and with enriched local approximation spaces. This is in essence the situation with all approaches to a conforming boundary treatment in meshfree methods — it can in principle be done but only under very restrictive assumptions on the input data. The presented algebraic approach now overcomes this issue for all partition of unity methods if the flat top property is satisfied. To our knowledge

our approach is the only technique that is fully automatic and generally applicable to arbitrary input and thereby very easy on the user.

We impose no additional restrictions on the local approximation spaces nor do we assume a specific distribution of discretization nodes near or even on the boundary. We explicitly construct the conforming subspaces of the employed local approximation space after we have defined the current approximation to the boundary of the domain. Therefore, we can refine and adapt the local approximation spaces automatically on each level without any user interaction. This ease on the user however has a prize — we may obtain slightly larger errors than with the non-conforming approach. The quality of the approximation is asymptotically optimal but near the boundary we obtain a quite different approximation than with Nitsche's method. Moreover, the efficiency of our multilevel solver may be somewhat reduced (but still optimal) with the conforming boundary treatment due to the explicit enforcement of the boundary conditions on all levels.

Recall that the presented construction of our multilevel solver is independent of the employed local approximation spaces. We make no assumption on the structure of these spaces or their basis functions. We only require a specific local geometric hierarchy of the employed cover patches which however is automatically ensured by our tree-based cover construction scheme and our local refinement procedure. The presented numerical results show that our multilevel solver converges with a rate that is independent of the number of discretization points (i.e. cover patches or partition of unity functions). Moreover, the convergence behavior is independent of the employed enrichment functions and only slightly dependent on the polynomial degree. In fact, if we enforce that the polynomial degrees employed on coarser levels are truly smaller than on finer levels we find no dependence of the multilevel convergence rate on the polynomial degree at all (at least for the practical range $p = 1, \dots, 5$).

Besides the classical hp-refinement of our approximation space we furthermore presented a hierarchical enrichment scheme that can be viewed as an algebraic refinement scheme. With the help of this technique we recover the optimal convergence rate of the uniform h-version independently of the regularity of the solution. Moreover, we attain a kind of super-convergence *near* its singularities, theoretically and numerically.

In summary, the methodology presented in this manuscript is applicable to arbitrary clouds of discretization points and arbitrary local approximation spaces and provides approximation spaces of arbitrary regularity. Thus, it is not limited to a specific application but rather a general purpose approach with much room for the utilization of application-adapted (a priori) information. Here, the user can focus completely on the approximation properties and must not be concerned with the stability of the method since the presented local preconditioning scheme automatically ensures the stability of the constructed basis while maintaining the required approximation properties.

Extensions

Thus we anticipate that the incorporation of numerical enrichment functions e.g. by a global-local approach [40] in our meshfree generalized finite element method is straightforward and will not impact the convergence behavior of our multilevel solver.

Also the automatic construction of quasi-divergence free approximation spaces using the local splitting techniques developed for the conforming boundary treatment is a promising approach for fluid flow applications. In principle the proposed construction is applicable to all equality constraints imposed on the approximation space. If the constraints involve function values only the constraints can be eliminated completely, for gradient-based constraints we attain quasi-optimal subspaces only.

Another important issue is the interplay of our conforming boundary treatment and the geometry approximation. Recall that the presented numerical integration scheme is targeted to

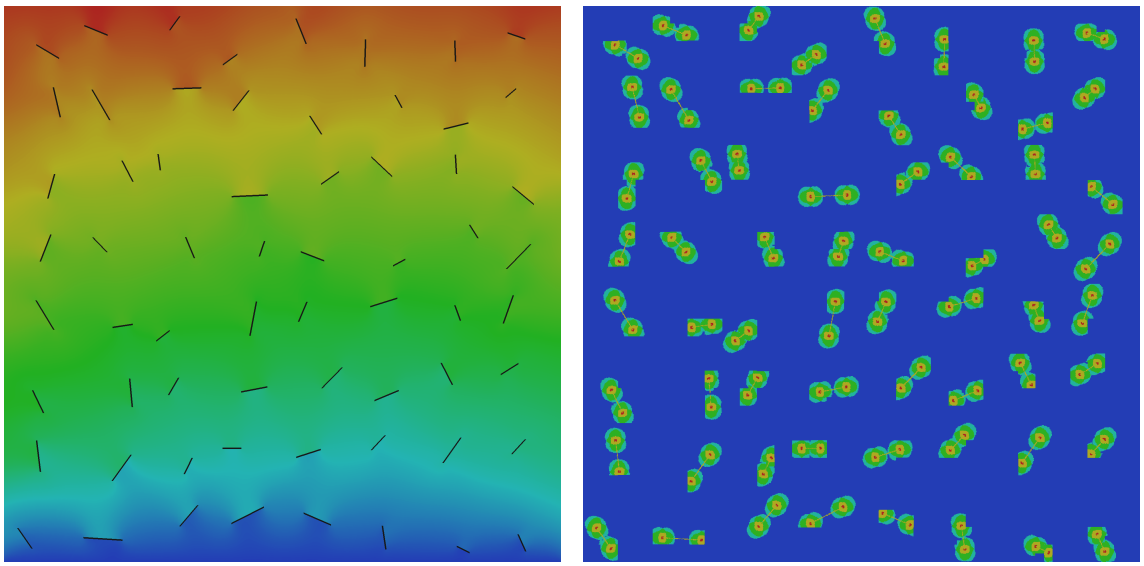


Figure 8.1. *The magnitude of the deformation field (left) and the distribution of the enrichment degrees of freedom (right) on level* $k = 10$ *of a hierarchically enriched (starting on level* 5*) fracture mechanics problem with* 64 *randomly distributed cracks.*

provide a high quality approximation to the domain on all levels and is in some sense independent of the local approximation spaces. However that means that the geometric complexity of the approximated boundary is in general too high for the employed local approximation spaces especially on coarse levels. Thus, the quality of the coarse grid correction of our multilevel solver for a conforming discretization is somewhat compromised. Here, we propose two strategies which might overcome this issue: The local approximation spaces at the boundary can be enriched with a (scaled) distance function to the (approximated) boundary or we can even try to construct a local coordinate system that is aligned with the local boundary. Obviously, the latter approach is substantially simplified if the local boundary approximation within a patch is linear (in general a curve that can be represented exactly by the employed local approximation spaces). Hence, if we re-order the steps of the construction of the numerical integration cells and approximate the boundary directly on the tree-cells we obtain a less accurate approximation to the boundary but one that is better suited for its representation with the employed local approximation spaces. Moreover, the identification of geometric singularities is simplified with this coarser boundary approximation so that the automatic enrichment of the respective local approximation spaces with the respective corner singularity can be done in optimal complexity.

Further, we can substantially reduce the total operation counter of the proposed method with the help of the conforming splitting of the approximation space. Recall that we currently discretize the right-hand side explicitly, i.e. $\hat{f} := (\hat{f}_i^m)$ where

$$\hat{f}_i^m := \int_\Omega f \varphi_i \vartheta_i^m \, dx + \int_{\Gamma_N} g_N \varphi_i \vartheta_i^m \, ds + \beta \int_{\Gamma_D} g_D (\varphi_i \vartheta_i^m - (\nabla \varphi_i \vartheta_i^m \cdot n)) \, ds$$

for a Nitsche discretization of a Poisson problem with Dirichlet boundary $\Gamma_D \subset \partial\Omega$ and Neumann boundary conditions on $\Gamma_N := \partial\Omega \setminus \Gamma_D$. Thus, the caching strategy mentioned in Remark 6.4 for differential operators with constant coefficients is not applicable to the assembly of $\hat{f}$. Observe that we cannot directly interpolate the function f to obtain its coefficient vector $\tilde{f}$ so that we may not use the approximation $\hat{f} = M\tilde{f}$ commonly employed in the classical FEM. Yet, with the help

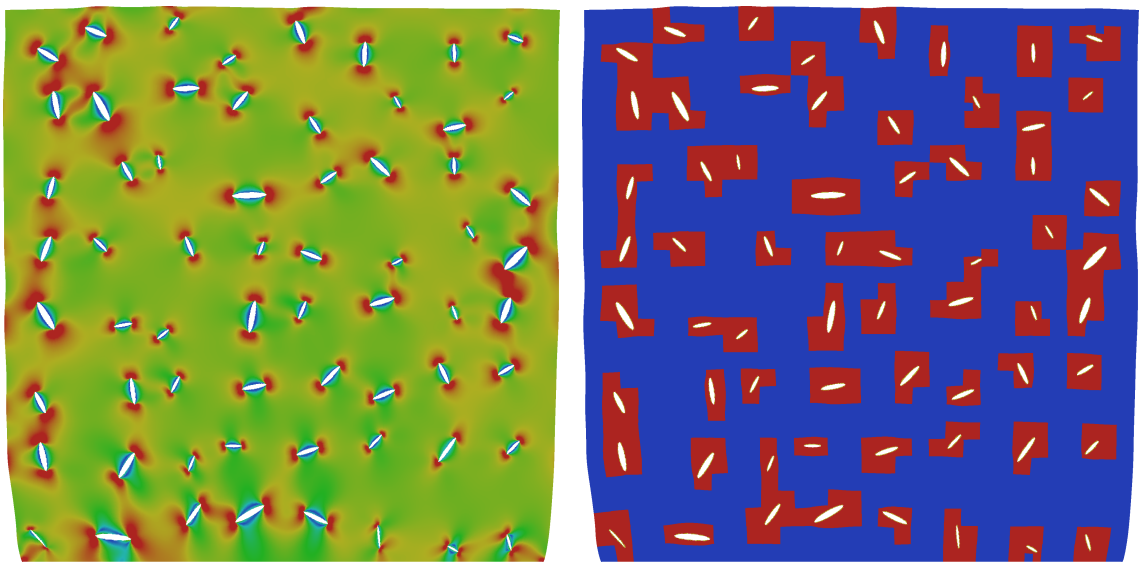


Figure 8.2. *The von Mises stress distribution (left) and the enrichment zones (right) on the (scaled) deformed configuration on level* $k = 10$ *of a hierarchically enriched (starting on level* 5*) fracture mechanics problem with* 64 *randomly distributed cracks.*

of the local-to-local mass matrix $\widetilde{M}$ and the conforming splitting of the approximation space we can obtain discrete representations of the data f, g_D and g_N, i.e. the respective coefficient vectors $\tilde{f}$, $\tilde{g}_D$ and $\tilde{g}_N$, very efficiently. Then, we can define an approximation to the right-hand side with the help of the global mass matrix M which satisfies the assumptions for our caching technique. Observe though that this procedure requires the design of an appropriate load estimate for our parallel load balancing scheme which is rather involved since the effects on memory usage and operation counts are not uniform. Prior to this development though a detailed study of the overall approximation error attainable with this approach must be carried out.

We close our treatise on the meshfree generalization of the classical finite element method with some snapshots of further numerical results attained with the developed particle-partition of unity method. In Figure 8.1 we give the contour plots of the magnitude of the deformation field obtained for a hierarchically enriched discretization of a fracture mechanics problem with 64 randomly distributed cracks and the distribution of the pure enrichment degrees of freedom. The respective von Mises stress is depicted in Figure 8.2 on the (scaled) deformed configuration. We can clearly observe the singularities at the crack tips. Note that we initialized the hierarchical enrichment scheme on level $k = 5$ only. Therefore, we see some geometric artifacts (in fact the tree-cell structure on level $k = 5$) in the respective enrichment zones and the distribution of the pure enrichment degrees of freedom. But still around all tips do we find multiple layers of enriched patches which resolve the singular behavior of the solution in the resulting enrichment zone. The visualization of such high resolution results however remains a very challenging problem. For the plots given in the Figures 8.1 and 8.2 we used the cell centers, i.e. 1048576 points, only whereas the approximation employed 4296883 (scalar) degrees of freedom for each physical variable with 1151155 discontinuous (directly at the crack line) or singular enrichment functions. In Figure 8.3 we show the von Mises stress distribution for an h-adaptive discretization of a crack problem with minimal enrichment. Here, we can clearly observe that the refinement jumps align with the contour lines of the singularities of the solution. Note that here we only employ around 64 singular enrichment functions so that adaptive refinement of the singularities is very much

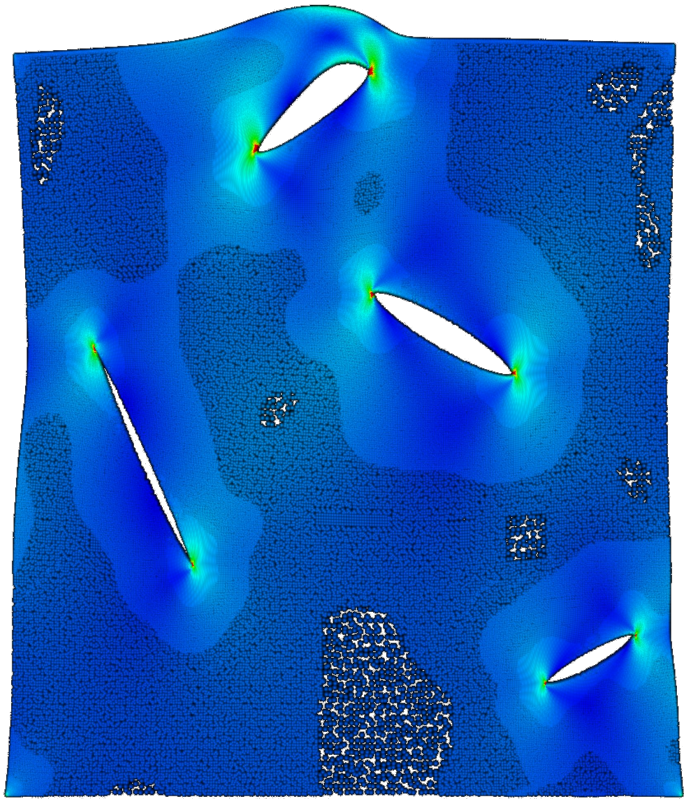


Figure 8.3. *The von Mises stress distribution on the (scaled) deformed configuration of an h-adaptive discretization with minimal enrichment.*

necessary.

The results depicted in Figures 8.4 and 8.5 are given to confirm that we are not limited to linear cracks or disjoint cracks. The intersection of multiple cracks for instance is rather straight forward due to our local preconditioning approach. Those patches that cover the point of intersection not only employ the enrichment functions for each crack $H_{\pm}^{C_1}\mathcal{P}^{p_i}$ and $H_{\pm}^{C_2}\mathcal{P}^{p_i}$ but also the additional multiplicative enrichment $H_{\pm}^{C_1}H_{\pm}^{C_2}\mathcal{P}^{p_i}$, i.e.

$$\mathcal{E}_i = H_{\pm}^{C_1}\mathcal{P}^{p_i} + H_{\pm}^{C_2}\mathcal{P}^{p_i} + H_{\pm}^{C_1}H_{\pm}^{C_2}\mathcal{P}^{p_i}.$$

The extension to the intersection of more than two intersecting cracks is straightforward. Recall that in the choice of the enrichment space we must only be concerned with the approximation properties *not* the stability. Thus, we must only include all necessary discontinuities which are certainly generated by our approach. If more than two cracks intersect at a given point the above approach yields the correct discontinuities but generates linearly dependent functions. This however is no cause of concern in our method since our local preconditioning technique will reliably determine these dependencies and will provide a stable basis which encodes all necessary discontinuities automatically.

If the crack line is curved the tip enrichment functions are in some sense valid in front of the tip only. Recall that the tip functions are discontinuous across the negative x-axis of the local tip coordinate system, compare Figure 5.7. Thus, we can employ the singular enrichment for patches (away from the tip) which do not intersect this (artificial) line of discontinuity. Even though the

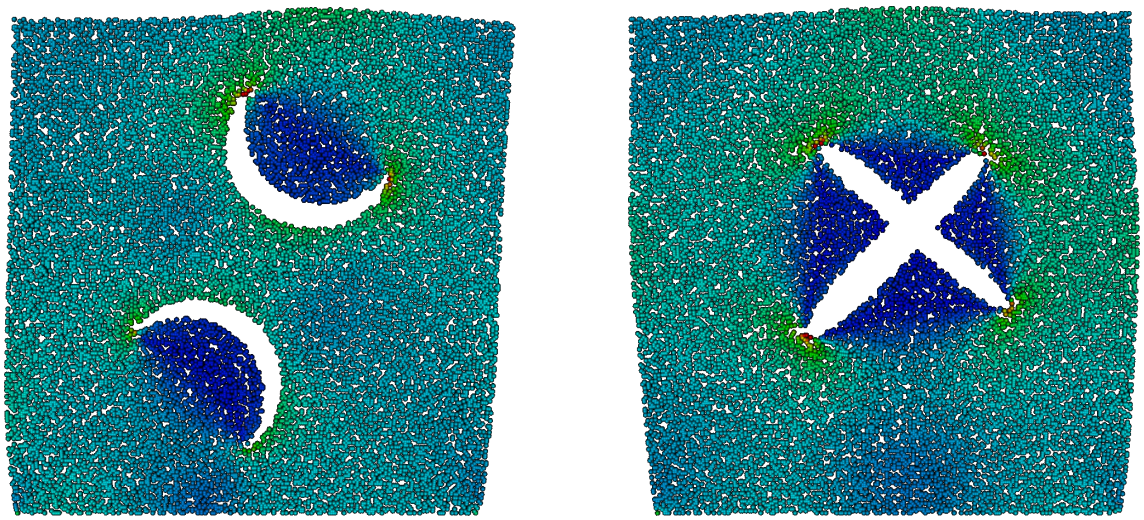


Figure 8.4. *The von Mises stress distribution for a fracture mechanics problem with curved cracks (left) and intersecting cracks (right).*

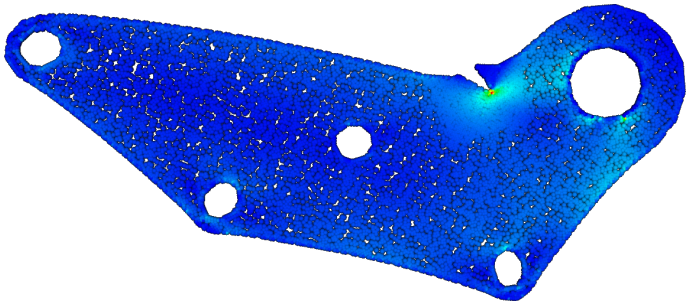


Figure 8.5. *The von Mises stress distribution on the (scaled) deformed configuration for an h-adaptive discretization with minimal enrichment of a cracked lever.*

enrichment functions may not perfectly match the singular behavior of the solution away from the tip they still provide additional approximation power. However it is essential that we do not introduce artificial discontinuities into the approximation space. This is especially important for the convergence of our multilevel solver. All discontinuities in our multilevel hierarchy must be aligned otherwise the quality of our transfer operators is spoiled. This issue however may be overcome by a slight generalization of the construction of our transfer operators. Here, we restrict the L^2-projection approach to the smooth components of the local approximation spaces only and assuming that the enrichment functions are designed to capture the correct physical behavior on the coarse and the fine level respectively we can simply use the identity matrix to transfer information. This approach would also allow us to employ a coarse representation of the crack line on coarser levels without introducing artificial discontinuities. However the details of this approach and its assessment must be left to future research.

Bibliography

[1] D. N. ARNOLD, F. BREZZI, B. COCKBURN, AND D. MARINI, *Discontinuous Galerkin Methods for Elliptic Problems*, in Discontinuous Galerkin Methods: Theory, Computation and Applications, B. Cockburn, G. Karniadakis, and C.-W. Shu, eds., Springer, 1999, pp. 89–101.

[2] H. BABOVSKY, *Die Boltzmann–Gleichung*, B. G. Teubner, 1998.

[3] I. BABUŠKA, *Numerical Solution of Boundary Value Problems by the Perturbated Variational Principle*, Tech. Note BN-624, University of Maryland, 1969.

[4] ——, *The Finite Element Method with Lagrangian Multipliers*, Numer. Math., 20 (1973), pp. 179–192.

[5] I. BABUŠKA, U. BANERJEE, AND J. E. OSBORN, *Meshless and Generalized Finite Element Methods: A Survey of Some Major Results*, in Meshfree Methods for Partial Differential Equations, M. Griebel and M. A. Schweitzer, eds., vol. 26 of Lecture Notes in Computational Science and Engineering, Springer, 2002, pp. 1–20.

[6] ——, *Quadrature Problem in the Meshless Method*. Presentation at Third International Workshop on Meshfree Methods for Partial Differential Equations, 2005.

[7] I. BABUŠKA, G. CALOZ, AND J. E. OSBORN, *Special Finite Element Methods for a Class of Second Order Elliptic Problems with Rough Coefficients*, SIAM J. Numer. Anal., 31 (1994), pp. 945–981.

[8] I. BABUŠKA AND J. M. MELENK, *The Partition of Unity Finite Element Method: Basic Theory and Applications*, Comput. Meth. Appl. Mech. Engrg., 139 (1996), pp. 289–314. Special Issue on Meshless Methods.

[9] ——, *The Partition of Unity Method*, Int. J. Numer. Meth. Engrg., 40 (1997), pp. 727–758.

[10] I. BABUŠKA AND W. C. RHEINBOLDT, *Error Estimates for Adaptive Finite Element Computations*, SIAM J. Numer. Anal., 15 (1978), pp. 736–754.

[11] I. BABUŠKA AND T. STROUBOULIS, *The Finite Element Method and Its Reliability*, Oxford University Press, 2001.

[12] U. BANERJEE, I. BABUŠKA, AND J. E. OSBORN, *Effect of Quadrature on Generalized Finite Element Method*. Presentation at Fourth International Workshop on Meshfree Methods for Partial Differential Equations, 2007.

[13] R. E. BANK AND A. WEISER, *Some A Posteriori Error Estimators for Elliptic Partial Differential Equations*, Math. Comp., 44 (1985).

[14] R. BECKER, P. HANSBO, AND R. STENBERG, *A Finite Element Method for Domain Decomposition with Non-Matching Grids*, Math. Modell. Numer. Anal., 37 (2003), pp. 209–225.

[15] S. BEISSEL AND T. BELYTSCHKO, *Nodal Integration of the Element-Free Galerkin Method*, Comput. Meth. Appl. Mech. Engrg., 139 (1996), pp. 49–74.

[16] T. BELYTSCHKO AND T. BLACK, *Elastic Crack Growth in Finite Elements with Minimal Remeshing*, Int. J. Numer. Meth. Engrg., 45 (1999), pp. 601–620.

[17] T. BELYTSCHKO, Y. KRONGAUZ, D. ORGAN, M. FLEMING, AND P. KRYSL, *Meshless Methods: An Overview and Recent Developments*, Comput. Meth. Appl. Mech. Engrg., 139 (1996), pp. 3–47. Special Issue on Meshless Methods.

[18] T. BELYTSCHKO, Y. Y. LU, AND L. GU, *Element-free Galerkin methods*, Int. J. Numer. Meth. Engrg., 37 (1994), pp. 229–256.

[19] ——, *Crack Propagation by Element-free Galerkin methods*, Engrg. Frac. Mech., 51 (1995), pp. 295–315.

[20] T. BELYTSCHKO, N. MOËS, S. USUI, AND C. PARIMI, *Arbitrary Discontinuities in Finite Elements*, Int. J. Numer. Meth. Engrg., 50 (2001), pp. 993–1013.

[21] M. BERN, D. EPPSTEIN, AND J. GILBERT, *Provably Good Mesh Generation*, J. Comput. Sys. Sci., 48 (1994), pp. 384–409.

[22] D. BRAESS, *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*, Cambridge University Press, 2001.

[23] D. BRAESS AND R. VERFÜRTH, *Multi-Grid Methods for Nonconforming Finite Element Methods*, SIAM J. Numer. Anal., 27 (1990), pp. 979–986.

[24] J. H. BRAMBLE, *The Lagrange Multiplier Method for Dirichlet's Problem*, Math. Comp., 37 (1981), pp. 1–11.

[25] J. H. BRAMBLE AND X. ZHANG, *Handbook of Numerical Analysis*, in The Analysis of Multigrid Methods, P. G. Ciarlet and J. L. Lions, eds., vol. VII, Elsevier, 2000, pp. 173–416.

[26] A. BRANDT, *Multi-level Adaptive Technique (MLAT) for Fast Numerical Solution to Boundary Value Problems*, in Proc. Third Int. Conf. Numerical Methods in Fluid Mechanics 1972, H. Cabannes and R. Temam, eds., vol. 18 of Lecture Notes in Physics, Springer, 1973, pp. 82–89.

[27] ——, *Multi-level Adaptive Solutions to Boundary-Value Problems*, Math. Comp., 31 (1977), pp. 333–390.

[28] ——, *Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics*, tech. rep., GMD, 1984.

[29] W. L. BRIGGS, V. E. HENSON, AND S. F. MCCORMICK, *A Multigrid Tutorial*, SIAM, 2nd ed., 2000.

[30] J. S. CHEN, C. T. WU, AND S. YOON, *Non-linear Version of Stabilized Conforming Nodal Integration for Galerkin Mesh-free Methods*, Int. J. Numer. Meth. Engrg., 53 (2002), pp. 2587–2615.

[31] J. S. CHEN, C. T. WU, S. YOON, AND Y. YOU, *A Stabilized Conforming Nodal Integration for Galerkin Mesh-free Methods*, Int. J. Numer. Meth. Engrg., 50 (2001), pp. 435–466.

[32] D. CHO AND L. ZIKATANOV, *A Multilevel Preconditioning for Generalized Finite Element Method Problems on Unstructured Simplicial Meshes*, J. Numer. Math, 15 (2007), pp. 163–180.

[33] W. DAHMEN, *Multiscale Analysis, Approximation, and Interpolation Spaces*, in Approximation Theory VIII, C. K. Chui and L. L. Schumaker, eds., vol. 2, World Scientific, 1995, pp. 47–88.

[34] W. DAHMEN, S. DEKEL, AND P. PETRUSHEV, *Multilevel Preconditioning for Partition of Unity Methods: Some Analytic Concepts*, Numer. Math., 107 (2007), pp. 503–532.

[35] R. DAVE, J. DUBINSKI, AND L. HERNQUIST, *Parallel TreeSPH*, New Astronomy, 2 (1997), pp. 277–297.

[36] S. DE AND K. J. BATHE, *The Method of Finite Spheres*, Comput. Mech., 25 (2000), pp. 329–345.

[37] G. A. DILTS, *Moving-Least-Square-Particle Hydrodynamics I: Consistency and Stability*, Int. J. Numer. Meth. Engrg., 44 (1999), pp. 1115–1155.

[38] ——, *Moving-Least-Square-Particle Hydrodynamics II: Conservation and Boundaries*, Int. J. Numer. Meth. Engrg., 48 (2000), pp. 1503–1524.

[39] J. DOLBOW AND T. BELYTSCHKO, *Numerical Integration of the Galerkin Weak Form in Meshfree Methods*, Comput. Mech., 23 (1999), pp. 219–230.

[40] C. A. DUARTE AND D.-J. KIM, *Analysis and Applications of a Generalized Finite Element Method with Global-Local Enrichment Functions*, Comput. Meth. Appl. Mech. Engrg., (2007). to appear.

[41] C. A. DUARTE AND J. T. ODEN, *An h-p Adaptive Method using Clouds*, Comput. Meth. Appl. Mech. Engrg., 139 (1996), pp. 237–262.

[42] C. A. DUARTE, L. G. RENO, AND A. SIMONE, *A Higher Order Generalized FEM for through-the-thickness Branched Cracks*, Int. J. Numer. Meth. Engrg., 72 (2007), pp. 325–351.

[43] C. A. M. DUARTE, *A Review of Some Meshless Methods to Solve Partial Differtial Equations*, Tech. Rep. 95-06, TICAM, University of Texas, 1995.

[44] C. A. M. DUARTE, I. BABUŠKA, AND J. T. ODEN, *Generalized Finite Element Methods for Three Dimensional Structural Mechanics Problems*, Comput. Struc., 77 (2000), pp. 215–232.

[45] C. A. M. DUARTE, O. N. H. T. J. LISZKA, AND W. W. TWORZYDLO, *A Generalized Finite Element Method for the Simulation of Three-dimensional Dynamic Crack Propagation*, Int. J. Numer. Meth. Engrg., 190 (2001), pp. 2227–2262.

[46] C. A. M. DUARTE AND J. T. ODEN, *hp Clouds – A Meshless Method to Solve Boundary Value Problems*, Numer. Meth. for PDE, 12 (1996), pp. 673–705.

[47] M. EIGEL, E. GEORGE, AND M. KIRKILIONIS, *The Partition of Unity Meshfree Method for Solving Transport-Reaction Equations on Complex Domains: Implementation and Applications in the Life Sciences*, in Meshfree Methods IV, M. Griebel and M. A. Schweitzer, eds., vol. 65, Springer, 2008. to appear.

[48] G. FASSHAUER, *Meshfree Methods*, in Handbook of Theoretical and Computational Nanotechnology, M. Rieth and W. Schommers, eds., American Scientific Publishers, 2006, pp. 33–97.

[49] ——, *Meshfree Approximation Methods with MATLAB*, World Scientific, 2007.

[50] R. P. FEDORENKO, *A Relaxation Method for Solving Elliptic Difference Equations*, U.S.S.R. Comput. Math. Math. Phys., 1 (1962), pp. 1092–1096.

[51] C. FRANKE AND R. SCHABACK, *Convergence Orders of Meshless Collocation Methods using Radial Basis Functions*, Adv. in Comput. Math., 8 (1998), pp. 381–399.

[52] ——, *Solving Partial Differential Equations by Collocation using Radial Basis Functions*, Appl. Math. and Comput., 93 (1998), pp. 73–82.

[53] T.-P. FRIES AND H.-G. MATTHIES, *Classification and Overview of Meshfree Methods*, Informationbericht 2003-3, Technische Universität Braunschweig, 2004.

[54] T. GERSTNER AND M. GRIEBEL, *Numerical Integration using Sparse Grids*, Numer. Alg., 18 (1998), pp. 209–232.

[55] R. A. GINGOLD AND J. J. MONAGHAN, *Smoothed Particle Hydrodynamics: Theory and Application to non-spherical Stars*, Mon. Not. R. Astr. Soc., 181 (1977), pp. 375–389.

[56] ——, *Kernel Estimates as a Basis for General Particle Methods in Hydrodynamics*, J. Comput. Phys., 46 (1982), pp. 429–453.

[57] R. T. GLASSEY, *The Cauchy Problem in Kinetic Theory*, SIAM, 1996.

[58] M. GRIEBEL, S. KNAPEK, AND G. ZUMBUSCH, *Numerical Simulation in Molecular Dynamics*, Springer, 2007.

[59] M. GRIEBEL, P. OSWALD, AND M. A. SCHWEITZER, *A Particle-Partition of Unity Method—Part VI: A p-robust Multilevel Solver*, in Meshfree Methods for Partial Differential Equations II, M. Griebel and M. A. Schweitzer, eds., vol. 43 of Lecture Notes in Computational Science and Engineering, Springer, 2005, pp. 71–92.

[60] M. GRIEBEL AND M. A. SCHWEITZER, *A Particle-Partition of Unity Method for the Solution of Elliptic, Parabolic and Hyperbolic PDE*, SIAM J. Sci. Comput., 22 (2000), pp. 853–890.

[61] ——, *A Particle-Partition of Unity Method—Part II: Efficient Cover Construction and Reliable Integration*, SIAM J. Sci. Comput., 23 (2002), pp. 1655–1682.

[62] ——, *A Particle-Partition of Unity Method—Part III: A Multilevel Solver*, SIAM J. Sci. Comput., 24 (2002), pp. 377–409.

[63] ——, *A Particle-Partition of Unity Method—Part IV: Parallelization*, in Meshfree Methods for Partial Differential Equations, M. Griebel and M. A. Schweitzer, eds., vol. 26 of Lecture Notes in Computational Science and Engineering, Springer, 2002, pp. 161–192.

[64] ——, *A Particle-Partition of Unity Method—Part V: Boundary Conditions*, in Geometric Analysis and Nonlinear Partial Differential Equations, S. Hildebrandt and H. Karcher, eds., Springer, 2002, pp. 517–540.

[65] ——, *A Particle-Partition of Unity Method—Part VII: Adaptivity*, in Meshfree Methods for Partial Differential Equations III, M. Griebel and M. A. Schweitzer, eds., vol. 57 of Lecture Notes in Computational Science and Engineering, Springer, 2006, pp. 121–148.

[66] ——, eds., *Meshfree Methods for Partial Differential Equations IV*, vol. 65 of Lecture Notes in Computational Science and Engineering, Springer, 2008.

[67] M. GRIEBEL AND G. W. ZUMBUSCH, *Parallel Adaptive Subspace Correction Schemes with Applications to Elasticity*, Comput. Meth. Appl. Mech. Engrg., 184 (2000), pp. 303–332.

[68] F. C. GÜNTHER AND W. K. LIU, *Implementation of Boundary Conditions for Meshless Methods*, Comput. Meth. Appl. Mech. Engrg., 163 (1998), pp. 205–230.

[69] W. HACKBUSCH, *A Fast Numerical Method for Elliptic Boundary Value Problems with Variable Coefficients*, in Second GAMM-Conf. Numerical Methods in Fluid Mechanics, E. H. Hirschel and W. Geller, eds., DFVLR, 1977, pp. 50–57.

[70] ——, *Multi-Grid Methods and Applications*, vol. 4 of Springer Series in Computational Mathematics, Springer, 1985.

[71] ——, *Iterative Solution of Large Sparse Linear Systems of Equations*, Springer, 1994.

[72] ——, *A Sparse Matrix Arithmetic based on H-Matrices. Part I: Introduction to H-Matrices*, Computing, 62 (1999), pp. 89–108.

[73] W. HACKBUSCH, B. .KHOROMSKIJ, AND S. SAUTER, *On H^2-Matrices*, in Lectures on Applied Mathematics, H.-J. Bungartz, R. H. W. Hoppe, and C. Zenger, eds., Springer, 2000, pp. 9–29.

[74] W. HAN AND X. MENG, *Some Studies of the Reproducing Kernel Particle Method*, in Meshfree Methods for Partial Differential Equations, M. Griebel and M. A. Schweitzer, eds., vol. 26 of Lecture Notes in Computational Science and Engineering, Springer, 2002, pp. 193–210.

[75] X. HAN, S. OLIVEIRA, AND D. STEWART, *Finding Sets Covering a Point with Application to Mesh-Free Galerkin Methods*, SIAM J. Comput., 30 (2000), pp. 1368–1383.

[76] A. HUERTA, T. BELYTSCHKO, T. FERNÁNDEZ-MÉNDEZ, AND T. RABCZUK, *Meshfree Methods*, vol. 1 of Encyclopedia of Computational Mechanics, Wiley, 2004, ch. 10, pp. 279–309.

[77] A. ISKE, *Multiresolution Modelling in Scattered Data Modelling*, vol. 37 of Lecture Notes in Computational Science and Engineering, Springer, 2004.

[78] E. J. KANSA, *Multiquadratics—A Scattered Data Approximation Scheme with Applications to Computational Fluid-Dynamics—I Surface Approximations and Partial Derivative Estimates*, Comput. Math. Appl., 19 (1990), pp. 127–145.

[79] ——, *Multiquadratics—A Scattered Data Approximation Scheme with Applications to Computational Fluid-Dynamics—II Solutions to Parabolic, Hyperbolic and Elliptic Partial Differential Equations*, Comput. Math. Appl., 19 (1990), pp. 147–161.

[80] O. KLAAS AND M. S. SHEPARD, *Automatic Generation of Octree-based Three-Dimensional Discretizations for Partition of Unity Methods*, Comput. Mech., 25 (2000), pp. 296–304.

[81] M. KLINGLER, P. LEINEN, AND H. YSERENTANT, *The Finite Mass Method on Domains with Boundary*, SIAM J. Sci. Comp., 26 (2005), pp. 1744–1759.

[82] ——, *A Restart Procedure for the Finite Mass Method*, SIAM J. Sci. Comp., 30 (2007), pp. 117–133.

[83] D. E. KNUTH, *The Art of Computer Programming*, vol. 3 Searching and Sorting, Addison Wesley, Second ed., 1998.

[84] Y. KRONGAUZ AND T. BELYTSCHKO, *Enforcement of Essential Boundary Conditions in Meshless Approximations using Finite Elements*, Comput. Meth. Appl. Mech. Engrg., 131 (1996), pp. 133–145.

[85] L. KRONSJØ AND G. DAHLQUIST, *On the Design of Nested Iterations for Elliptic Difference Equations*, BIT, 12 (1972), pp. 63–71.

[86] A. KUNOTH, *Multilevel Preconditioning – Appending Boundary Conditions by Lagrange Multipliers*, Adv. Comput. Math., 4 (1995), pp. 145–170. Special Issue on Multiscale Methods.

[87] P. LANCASTER, *Moving Weighted Least–Squares Methods*, in Polynomial and Spline Approximation, B. N. Sahney, ed., vol. 49 of NATO Adv. Stud. Series C, D. Reidel, 1979, pp. 103–120.

[88] P. LANCASTER AND K. SALKAUSKAS, *Surfaces Generated by Moving Least Squares Methods*, Math. Comp., 37 (1981), pp. 141–158.

[89] S. LI AND W. K. LIU, *Reproducing Kernel Hierarchical Partition of Unity, Part I—Formulation and Theory*, Int. J. Numer. Meth. Engrg., 45 (1999), pp. 251–288.

[90] ——, *Meshfree Particle Methods*, Springer, 2004.

[91] C. LIA AND G. CARRARO, *A Parallel Tree SPH Code for Galaxy Formation*, Month. Not. Roy. Astro. Soc., 314 (2000), pp. 145–161.

[92] T. LISKA AND J. ORKISZ, *The Finite Difference Method at Arbitrary Irregular Grids and its Application in Applied Mechanics*, Comput. & Struct., 11 (1980), pp. 83–95.

[93] G. R. LIU AND Y. T. GU, *An Introduction to Meshfree Methods and Their Programming*, Springer, 2005.

[94] W. K. LIU, S. JUN, AND Y. F. ZHANG, *Reproducing Kernel Particle Methods*, Int. J. Numer. Meth. Fl., 20 (1995), pp. 1081–1106.

[95] W. K. LIU, S. LI, AND T. BELYTSCHKO, *Moving Least Square Reproducing Kernel Method Part I: Methodology and Convergence*, Comput. Meth. Appl. Mech. Engrg., 143 (1997), pp. 113–154.

[96] Y. Y. LU, T. BELYTSCHKO, AND L. GU, *A New Implementation of the Element Free Galerkin Method*, Comput. Math. Appl. Mech. Engrg., 113 (1994), pp. 397–414.

[97] L. B. LUCY, *A Numerical Approach to the Testing of the Fission Hypothesis*, Astro. J., 82 (1977), pp. 1013–1024.

[98] J. M. MELENK, *On approximation in meshless methods*, in Frontiers in Numerical Analysis (Durham 2004), J. Blowey and A. Craig, eds., Universitext, Springer, 2005, pp. 65–136.

[99] J. M. MELENK AND B. WOHLMUTH, *On Residual-based a posteriori Error Estimation in hp-FEM*, Adv. Comput. Math., 15 (2001), pp. 311–331.

[100] N. MOËS, J. DOLBOW, AND T. BELYTSCHKO, *A Finite Element Method for Crack Growth without Remeshing*, Int. J. Numer. Meth. Engrg., 46 (1999), pp. 131–150.

[101] S. MOHAMMADI, *Extended Finite Element Method*, Blackwell Publishing, 2008.

[102] J. J. MONAGHAN, *Why Particle Methods Work*, SIAM J. Sci. Stat. Comput., 3 (1982), pp. 422–433.

[103] ——, *An Introduction to SPH*, Comput. Phys. Comm., 48 (1988), pp. 89–96.

[104] K. NANBU, *Direct Simulation Scheme derived from the Boltzmann Equation*, J. Phys. Soc. Japan, 49 (1980), pp. 20–49.

[105] ——, *Theoretical Basis on the Direct Simulation Monte Carlo Method*, in Rarefied Gas Dynamics, V. Boffi and C. Cercignani, eds., vol. 1, Teubner, 1986.

[106] H. NEUNZERT, A. KLAR, AND J. STRUCKMEIER, *Particle Methods: Theory and Applications*, Tech. Rep. 95-153, Arbeitsgruppe Technomathematik, Universität Kaiserslautern, 1995.

[107] H. NEUNZERT AND J. STRUCKMEIER, *Particle Methods for the Boltzmann Equation*, Acta Numerica, (1995), pp. 417–457.

[108] J. NITSCHE, *Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind*, Abh. Math. Sem. Univ. Hamburg, 36 (1970–1971), pp. 9–15.

[109] J. T. ODEN AND C. A. DUARTE, *Clouds, Cracks and FEM's*, Recent Developments in Computational and Applied Mechanics, International Center for Numerical Methods in Engineering, CIMNE, Barcelona, Spain, 1997, pp. 302–321.

[110] P. OSWALD, *Multilevel Finite Element Approximation*, Teubner Skripten zur Numerik, Teubner, 1994.

[111] N. PARÉS, P. DÍEZ, AND A. HUERTA, *Subdomain-based Flux-free a posteriori Error Estimators*, Comput. Meth. Appl. Mech. Engrg., 195 (2006), pp. 297–323.

[112] L. F. PAVARINO, *Additive Schwarz Methods for the p-Version Finite Element Method*, Numer. Math., 66 (1994), pp. 493–515.

[113] J. PITKÄRANTA, *Boundary Subspaces for the Finite Element Method with Lagrange Multipliers*, Numer. Math., 33 (1979), pp. 273–289.

[114] ——, *Local Stability Conditions for the Babuška Method of Lagrange Multipliers*, Math. Comp., 35 (1980), pp. 1113–1129.

[115] ——, *The Finite Element Method with Lagrange Multipliers for Domains with Corners*, Math. Comp., 37 (1981), pp. 13–30.

[116] A. POTHEN, *Graph Partitioning Algorithms with Applications to Scientific Computing*, in Parallel Numerical Algorithms, D. E. Keyes, A. Sameh, and V. Venkatakrishnan, eds., Kluwer Academic Publishers, 1997, pp. 323–368.

[117] C. RIKER AND S. M. HOLZER, *The Mixed-Cell-Complex Partition-of-Unity Method*, Comput. Meth. Appl. Mech. Engrg., (2008). accepted.

[118] H. SAGAN, *Space-Filling Curves*, Springer, 1994.

[119] H. SAMET, *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*, Addison–Wesley, 1990.

[120] ——, *The Design and Analysis of Spatial Data Structures*, Addison–Wesley, 1990.

[121] R. SCHABACK AND H. WENDLAND, *Kernel Techniques: From Machine Learning to Meshless Methods*, Acta Numerica, 15 (2006), pp. 543–639.

[122] C. SCHWAB, *p- and hp-Finite Element Methods*, Clarendon Press, 1998.

[123] H. A. SCHWARZ, *Über einige Abbildungsaufgaben*, Vierteljahresschrift Naturforsch. Ges. Zürich, 15 (1870), pp. 272–286.

[124] M. A. SCHWEITZER, *Ein Partikel–Galerkin–Verfahren mit Ansatzfunktionen der Partition of Unity Method*, Diplomarbeit, Institut für Angewandte Mathematik, Universität Bonn, 1997.

[125] ——, *A Parallel Multilevel Partition of Unity Method for Elliptic Partial Differential Equations*, vol. 29 of Lecture Notes in Computational Science and Engineering, Springer, 2003.

[126] ——, *Implementation and Parallelization of Meshfree Methods*, in Frontiers in Numerical Analysis (Durham 2004), J. Blowey and A. Craig, eds., Universitext, Springer, 2005, pp. 195–257.

[127] ——, *A Particle-Partition of Unity Method Part VIII: Hierarchical Enrichment*, tech. rep., Sonderforschungsbereich 611, Rheinische Friedrich-Wilhelms-Univeristät Bonn, 2008. submitted.

[128] ——, *An Adaptive hp-Version of the Multilevel Particle–Partition of Unity Method*, Comput. Meth. Appl. Mech. Engrg., (2008). accepted.

[129] ——, *An Algebraic Treatment of Essential Boundary Conditions in the Particle–Partition of Unity Method*, tech. rep., Sonderforschungsbereich 611, Rheinische Friedrich-Wilhelms-Univeristät Bonn, 2008. submitted.

[130] ——, *Stable Enrichment and Local Preconditioning in the Particle–Partition of Unity Method*, in Meshfree Methods IV, M. Griebel and M. A. Schweitzer, eds., vol. 65, Springer, 2008. to appear.

[131] D. SHEPARD, *A Two-Dimensional Interpolation Function for Irregularly Spaced Data*, in Proceedings 1968 ACM Nat. Conf., ACM, 1968, pp. 517–524.

[132] B. F. SMITH, P. E. BJØRSTAD, AND W. D. GROPP, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, 1996.

[133] R. STENBERG, *Mortaring by a Method of J. A. Nitsche*, in Computational Mechanics, New Trends and Applications, S. Idelsohn, E. Onate, and E. Dvorkin, eds., 1998, pp. 1–6.

[134] T. STROUBOULIS, I. BABUŠKA, AND K. COPPS, *The Design and Analysis of the Generalized Finite Element Method*, Comput. Meth. Appl. Mech. Engrg., 181 (2000), pp. 43–69.

[135] T. STROUBOULIS, K. COPPS, AND I. BABUŠKA, *The Generalized Finite Element Method*, Comput. Meth. Appl. Mech. Engrg., 190 (2001), pp. 4081–4193.

[136] T. STROUBOULIS, L. ZHANG, AND I. BABUŠKA, *Generalized Finite Element Method using mesh-based Handbooks: Application to Problems in Domains with many Voids*, Comput. Meth. Appl. Mech. Engrg., 192 (2003), pp. 3109–3161.

[137] ——, *p-Version of the Generalized FEM using mesh-based Handbooks with Applications to Multiscale Problems*, Int. J. Numer. Meth. Engrg., 60 (2004), pp. 1639–1672.

[138] T. STROUBOULIS, L. ZHANG, D. WANG, AND I. BABUŠKA, *A posteriori Error Estimation for Generalized Finite Element Methods*, Comput. Meth. Appl. Mech. Engrg., 195 (2006), pp. 852–879.

[139] J. W. SWEGLE, S. W. ATTAWAY, F. J. MELLO, AND D. L. HICKS, *An Analysis of the Smoothed Particle Hydrodynamics*, Tech. Rep. SAND93-2513 UC-705, Sandia National Laboratories, 1994.

[140] B. SZABÓ AND I. BABUŠKA, *Finite Element Analysis*, John Wiley & Sons, 1991.

[141] U. TROTTENBERG, C. W. OSTERLEE, AND A. SCHÜLLE, *Multigrid*, Academic Press, 2001.

[142] U. TROTTENBERG, C. W. OSTERLEE, AND A. SCHÜLLER, *Multigrid*, Academic Press, 2001, Appendix A: An Introduction to Algebraic Multigrid by K. STÜBEN, pp. 413–532.

[143] I. TSUKERMAN, *A Class of Difference Schemes with Flexible Local Approximation*, The Journal of Computational Physics, 211 (2006), pp. 659–699.

[144] ——, *Computational Methods for Nanoscale Applications: Particles, Plasmons and Waves*, Springer, 2007.

[145] R. VERFÜRTH, *A Review of A Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*, Wiley Teubner, 1996.

[146] M. S. WARREN AND J. K. SALMON, *A Parallel Hashed Oct-Tree N-Body Algorithm*, in Supercomputing '93, IEEE Comput. Soc., 1993, pp. 12–21.

[147] ——, *A Portable Parallel Particle Program*, Comput. Phys. Comm., 87 (1995).

[148] H. WENDLAND, *Numerical Solution of Variational Problems by Radial Basis Functions*, in Approximation Theory IX, C. K. Chui and L. L. Schumaker, eds., vol. 2: Computational Aspects, Vanderbilt University Press, 1998, pp. 361–368.

[149] ——, *Meshless Galerkin Methods using Radial Basis Functions*, Math. of Comput., 68 (1999), pp. 1521–1531.

[150] ——, *Local Polynomial Reproduction and Moving Least Squares Approximation*, IMA Journal of Numerical Analysis, 21 (2001), pp. 285–300.

[151] ——, *Scattered Data Approximation*, Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, 2005.

[152] J. XU, *Iterative Methods by Space Decomposition and Subspace Correction*, SIAM Review, 34 (1992), pp. 581–613.

[153] J. XU AND L. T. ZIKATANOV, *On Multigrid Methods for Generalized Finite Element Methods*, in Meshfree Methods for Partial Differential Equations, M. Griebel and M. A. Schweitzer, eds., vol. 26 of Lecture Notes in Computational Science and Engineering, Springer, 2002, pp. 401–418.

[154] H. YSERENTANT, *Old and New Convergence Proofs for Multigrid Methods*, Acta Numerica 93, (1993), pp. 285–326.

[155] ——, *A New Class of Particle Methods*, Numer. Math., 76 (1997), pp. 87–109.

[156] ——, *A Particle Model of Compressible Fluids*, Numer. Math., 76 (1997), pp. 111–142.

[157] O. C. ZIENKIEWICZ AND J. Z. ZHU, *A Simple Error Estimator and Adapative Procedure for Practical Engineering Analysis*, Int. J. Numer. Meth. Engrg., (1987).

[158] G. W. ZUMBUSCH, *On the Quality of Space-Filling Curve Induced Partitions*, Z. Angew. Math. Mech., 81 Suppl. 1 (2001), pp. 25–28.

[159] ——, *Parallel Multilevel Methods. Adaptive Mesh Refinement and Loadbalancing*, Advances in Numerical Mathematics, Teubner, 2003.

[160] *Catalogue of Finite Element Books*. http://www.solid.ikp.liu.se/fe/index.html.

[161] *National Agency for Finite Element Methods and Standards*. http://www.nafems.org/.