On a multilevel parallel solver for higher order FEM on elliptic systems

Maharavo Randrianarivony

May 27, 2019

Abstract

We consider the efficient solving of the linear system originating from a higher order Finite Element Method. The proposed method is able to solve systems having a large number of unknowns with a few number of iterations and it requires only a tight memory capacity. For the stability, we use a hierarchical Scott-Zhang operator and the spider averaging operator. The method functions for elliptic systems in 2D and 3D and it is applied to the Poisson and the Navier-Lamé equations. We do not require that the domain be convex. The solution is not assumed to admit a $\mathbf{H}^2$-smoothness either. In addition, the mesh distribution is not necessarily concentrated at the boundary. Apart from theoretical consideration, we put a special emphasis on the numerical implementation: parallel execution, load balancing of the data among the processors, p-cascading from one coarse mesh onto a finer mesh, redistribution of the data after mesh refinements.

Contents

List of Tables

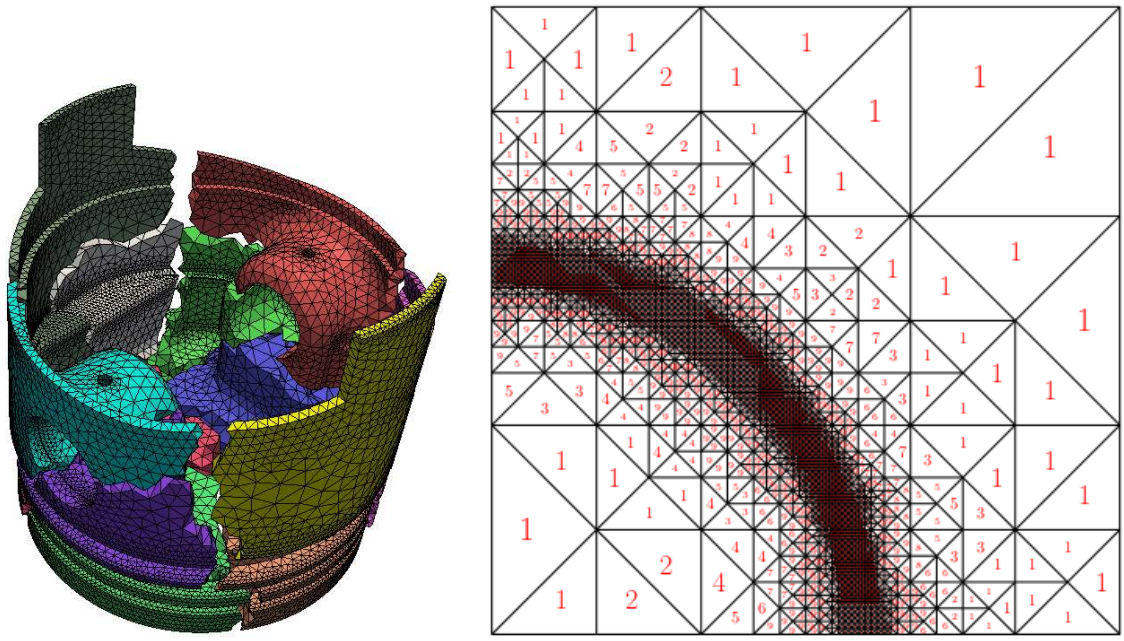
List of Figures

1 Introduction

The linear system solver is a very important component which cannot be neglected when considering a FEM (Finite Element Method) development. One valuable factor in deciding whether a particular FEM approach / implementation is efficient is based on the quality of its linear solver. The assembly of the FEM linear system is very fast because the matrix entries are usually computable exactly by using some transformation from the reference element. A longer time is therefore necessary in the linear solver than in the assembly of the system. We consider in this document the higher order setting of the FEM, i.e. the polynomial degree p in each element is allowed to be larger than unity. In term of convergence, such a setting is advantageous in the case of elasticity [45, 22] where the elastic properties approach the incompressible materials. Other simulations where the solution admits some local analiticity benefit also from higher order settings. The meshes $\mathcal{M}^{(\ell)}$ are supposed to be defined in a nested manner on a series of levels $\ell = 0, ..., L$. Such a case occurs usually in adaptive refinements but the method here can also be applied to globally uniform refinements. The polynomial degree, which is arbitrary but fixed, is uniformly constant in the entire FEM mesh. Apart from theoretical considerations, our main purpose here is also to report on the parallel implementation of the hierarchical p-solver. In addition, we describe some comparison between the presented approach and ParaSail implementation.

We survey now some pertaining works and we emphasize also our own contribution when relevant. For the construction of the p-basis functions inside each element, the basis functions are usually categorized into: vertex-modes, edge-modes, face-modes and interior-modes. To construct basis functions on the reference triangle (resp. tetrahedron), some transformations from the reference square (resp. hexahedron) are used in [41, 42] together with many combinations of 1D-Jacobi polynomials. Ainsworth *et al.* [2, 3, 4] and Kirby *et al.* [28, 29] use Bernstein polynomials which are easier to implement in the FEM context as the integrals of functions and derivatives on the reference simplex are expressed in a succinct formula. For the purely p-version, a construction [43] of a sequence u_p of degree p and of global smoothness $\mathcal{C}^{\ell-1}$ provides the estimates $\|u - u_p\|_{\mathbf{H}^\ell(\mathbf{\Omega})} \leq Cp^{\ell-k}\|u\|_{\mathbf{H}^k(\mathbf{\Omega})}$ (as $p \longrightarrow \infty$) which is a global approximation. In term of local approximation, a Clément type quasi-interpolant [31] provides in an element T the estimate $\|u - Iu\|_{\mathbf{L}_2(T)} + (h(T)/p(T))\|\nabla(u - Iu)\|_{\mathbf{L}_2(T)} \leq C(h(T)/p(T))\|\nabla(u)\|_{\mathbf{L}_2(\omega(T))}$ where $\omega(T)$ is some patch surrounding T. That is usually applied to convergence results and to a-posteriori analysis when combined with polynomial inverse estimates [34, 35]. For judiciously distributed mesh-size and polynomial degrees [10, 9], the *hp*-FEM achieves an exponential convergence of order $\exp(-\mu N^\gamma)$

where N designates the degree of freedom (DOF) and μ, γ are parameters independent of the discretization and the polynomial degrees. Choosing between applying an h-refinement or a p-refinement (degree elevation) [20, 21] does affect the convergence. Some known strategies include the *Three Step Texas* [33], a method based on local analyticity [24], devising an error prediction [32] based on the APEE (a-posteriori error estimator) and choosing the type of refinement whichever gives the better error reduction. For the Spectral Element Method, we find in [12] an APEE for the hp-case which has been generalized in [32] to treat hp-FEM [31] for the Poisson problem where corner singularities are allowed [36]. In the theoretical viewpoint, the main features of the presented approach are as follows. First, we do not assume any convexity of the domain $\mathbf{\Omega}$. Such a restriction usually facilitates the description of some theory but it appears to be a serious limitation for practical purpose. Second, the exact solution is not required to have $\mathbf{H}^2$-smoothness. Further, we do not assume any special mesh size distribution of $\mathcal{M}^{(\ell)}$. That latter point is one substantial difference from the work in [23] where the mesh is somehow concentrated in the vicinity of the boundary $\partial\mathbf{\Omega}$. The mesh in [23] is very fine at the boundary and it becomes coarser and coarser toward the interior. That is, for an element $T \in \mathcal{M}^{(\ell)}$ touching the boundary $\overline{T} \cap (\partial\mathbf{\Omega}) \neq \emptyset$, the mesh size is of order $h_T \approx h_0(1/2)^\ell$ on level ℓ where h_0 is some initial mesh size. For remote elements from the boundary, the mesh size is of order of the distance from the boundary such as $h_T \approx \text{dist}(T, \partial\mathbf{\Omega})$. Apart from discarding that assumption, the detail of the theoretical approach here is also different from [23] where the above assumption is used very extensively. In the study of the preconditioner lower bound, our analysis of the stability uses a series of hierarchical Scott-Zhang operators which are defined on every level and which inherit some properties from the previous levels. In addition, the method in [23] was applied to scalar-valued equations whereas the method here is applied to Navier-Lamé equations implemented on parallel machines. For the case $p = 1$, there are a large number of analysis and implementation of preconditioners and we survey here only a few. The theoretical approach here has a similarity with the MDS (Multi-Diagonal Scaling) of [46] where the author requires $\mathbf{H}^2$-smoothness combined the Aubin-Nitsche trick to derive the stability property. In addition, it requires that the sizes of the elements on mesh $\mathcal{M}^{(\ell)}$ is of order $\mathcal{O}(h^\ell)$ which we do not assume in this document. In [46], the analysis about the relation between the MDS and the BPX (Bramble-Pasciak-Xu) [15] preconditioner is documented. We make use also of the spider-averaging operator based on level-sets [39] to deduce some important inter-level properties. For ease of presentation, the theoretical part is limited to fixed polynomial degree but they could be applied with minor modifications to hp-method where the polynomial degree is bounded by a certain prescribed maximal value $p_{\max}$



(a) (b)

Figure 1: (a)Mesh distribution among the processors, (b)Typical *hp*-polynomial distribution where $p = 1, ..., p_{\max} = 9$.

as illustrated in Fig. 1(b). As for the p-version, we mention the very mathematically sound 2-level preconditioner in [39] showing a very good condition number. In term of BEM (Boundary Element Method), the most recent approach presenting a parallel implementation is documented in [27]. By using a sophisticated hardware (27 peta-flops) it can handle about 1.5 million unknowns. The setup phase takes about $30mn$ followed by a BEM linear solver of less than $1h30mn$. The model BEM solver treats a Poisson equation using a piecewise constant finite dimensional trial space. An optimal domain decomposition is not used which results in a high memory consumption but some attempts are made to avoid multiple identical copies of the BEM data structure. By increasing the BEM-unknowns, the error appears to stagnate with small fluctuations after a starting good reduction.

The outline of this document is as follows. In the following section, we introduce the higher order multilevel discretization with view on solving the linear system on the finest level. That will be followed by section 4 containing the analysis of the preconditioner upper bound. Section 3 will be occupied by the stability analysis providing the preconditioner lower bound. After exposing in section 5 the generalization to elliptic

systems, we report on some computer implementation and some numerical results. The proposed theory estimates only the number of required iterations. We need numerical tests because the dependence of all various constants with respect to the problem parameters is not established theoretically. Although the method here is valid for any elliptic systems, we restrict to the Poisson and the Navier-Lamé equations in the numerical experiments.

2 Problem setting

2.1 Additive Schwarz Method

We briefly recall only the setting of ASM (Additive Schwarz Method) which is needed in this document. For more comprehensive results, consult [46, 39, 15, 23] and the reference therein. Consider a symmetric, continuous and coercive bilinear form $a(\bullet,\bullet)$ on a linear space V spanned by $\mathbf{\Phi} = [\phi_1, ..., \phi_N]^{\mathrm{T}}$. Let $A = \big[a(\phi_i, \phi_j)\big]_{i,j=1}^{N}$ be the corresponding SPD matrix (symmetric positive definite). Instead of solving $A\mathbf{x} = b$, one solves the preconditioned system $(C^{-1}A)\mathbf{x} = C^{-1}b$. The ASM is a means to construct a preconditioner C in which one considers some subspaces V_i of V such that $V = \dotplus_{i=1}^{m} V_i$. The ASM uses $P := \sum_{i=1}^{m} P_i$ where P_i is the orthogonal projection:

$$P_i u \in V_i : \qquad a(P_i u, v) = a(u, v) \quad \forall v \in V_i. \tag{2.1}$$

Any function $u \in V$ can be written in vectorial form as $u = \mathbf{\Phi}^{\mathrm{T}}\mathbf{u}$ where $\mathbf{u} \in \mathbb{R}^N$. Since V_i is a subspace of V, there is some matrix R_i of size $N_i \times N$ such that any function $u_i \in V_i$ is written in vector form as $u_i = \mathbf{\Phi}^{\mathrm{T}} R_i \mathbf{u}_i$ where $\mathbf{u}_i \in \mathbb{R}^{N_i}$. For any fixed $i = 1, ..., N$, the expression of $u_i := P_i u$ in (2.1) is as follows for any $v = \mathbf{\Phi}^{\mathrm{T}} R_i \mathbf{v} \in V_i$:

$$a(u_i, v) = a(\mathbf{\Phi}^{\mathrm{T}} R_i \mathbf{u}_i, \mathbf{\Phi}^{\mathrm{T}} R_i \mathbf{v}) = \mathbf{v}^{\mathrm{T}} R_i^{\mathrm{T}} A R_i \mathbf{u}_i \tag{2.2}$$

$$a(u, v) = a(\mathbf{\Phi}^{\mathrm{T}}\mathbf{u}, \mathbf{\Phi}^{\mathrm{T}} R_i \mathbf{v}) = \mathbf{v}^{\mathrm{T}} R_i^{\mathrm{T}} A \mathbf{u}. \tag{2.3}$$

Equalizing both equations for any $\mathbf{v} \in \mathbb{R}^{N_i}$ yields

$$R_i^{\mathrm{T}} A R_i \mathbf{u}_i = R_i^{\mathrm{T}} A \mathbf{u} \qquad \text{i.e.} \qquad \mathbf{u}_i = (R_i^{\mathrm{T}} A R_i)^{-1} R_i^{\mathrm{T}} A \mathbf{u}. \tag{2.4}$$

Computing $w = Pu$ for any given $u = \mathbf{\Phi}^{\mathrm{T}}\mathbf{u} \in V$ in vector form where $w = \mathbf{\Phi}^{\mathrm{T}}\mathbf{w} \in V$ proceeds as follows. One has $w = \sum_{i=1}^{m} w_i$ where $w_i = P_i u = \mathbf{\Phi}^{\mathrm{T}} R_i \mathbf{w}_i \in V_i$. From (2.4), one has $\mathbf{w}_i = (R_i^{\mathrm{T}} A R_i)^{-1} R_i^{\mathrm{T}} A \mathbf{u}$. Therefore,

$$\mathbf{\Phi}^{\mathrm{T}}\mathbf{w} = \sum_{i=1}^{m} \mathbf{\Phi}^{\mathrm{T}}[R_i \mathbf{w}_i] = \mathbf{\Phi}^{\mathrm{T}} \Big[\sum_{i=1}^{m} R_i (R_i^{\mathrm{T}} A R_i)^{-1} R_i^{\mathrm{T}} \Big] A \mathbf{u}, \tag{2.5}$$

or equivalently,

$$\mathbf{w} = \Big[\sum_{i=1}^{m} R_i (R_i^{\mathrm{T}} A R_i)^{-1} R_i^{\mathrm{T}} \Big] A\mathbf{u}. \tag{2.6}$$

The ASM preconditioner is given by

$$C^{-1} := \sum_{i=1}^{m} R_i (R_i^{\mathrm{T}} A R_i)^{-1} R_i^{\mathrm{T}} \tag{2.7}$$

which amounts to solving m smaller problems $(R_i^{\mathrm{T}} A R_i)^{-1}$ of size $(N_i \times N_i)$ and applying the restriction and prolongation operators R_i^{T} and R_i. Suppose one has

$$\mu_0 \, a(u,u) \le a(Pu,u) \le \mu_1 \, a(u,u) \qquad \forall u \in V. \tag{2.8}$$

According to (2.6) and (2.7), one deduces $a(Pu,u) = \mathbf{u}^{\mathrm{T}} A C^{-1} A \mathbf{u}$. Hence,

$$\mu_0 \, \mathbf{u}^{\mathrm{T}} A \mathbf{u} \le \mathbf{u}^{\mathrm{T}} (A C^{-1} A) \mathbf{u} \le \mu_1 \, \mathbf{u}^{\mathrm{T}} A \mathbf{u} \qquad \forall \mathbf{u} \in \mathbb{R}^N. \tag{2.9}$$

Any eigenvalue $(C^{-1}A)\mathbf{u} = \lambda \mathbf{u}$ is an eigenvalue of $(AC^{-1}A)\mathbf{u} = \lambda A \mathbf{u}$. From (2.9) the Rayleigh quotient

$$\frac{\mathbf{u}^{\mathrm{T}} A C^{-1} A \mathbf{u}}{\mathbf{u}^{\mathrm{T}} A \mathbf{u}} \tag{2.10}$$

has value in the range $[\mu_0, \mu_1]$. Hence one has the condition number of the preconditioned system

$$\kappa(C^{-1}A) = \frac{\lambda_{\max}(C^{-1}A)}{\lambda_{\min}(C^{-1}A)} \le \frac{\mu_1}{\mu_0}. \tag{2.11}$$

By using the conjugate gradient iterative solver for $(C^{-1}A)\mathbf{x} = C^{-1}b$, the expected convergence speed for the k-th iterate is

$$\mathcal{O}\left[\frac{\sqrt{\kappa(C^{-1}A)} - 1}{\sqrt{\kappa(C^{-1}A)} + 1} \right]^k \tag{2.12}$$

which means that a large condition number results in a slow convergence. Our purpose in this document is to determine a space decomposition $V = \dotplus_{i=1}^{m} V_i$ and to prove for $P = \sum_i P_i$ the upper/lower bounds in (2.8) by determining the constants μ_0 and μ_1. Convergence is not the only important property when choosing a preconditioner. Other valuable factors include: (1)efficiency of applying the preconditioner C^{-1} to a given vector, (2)memory capacity required to store the preconditioner, (3)facility of updating a current preconditioner if the FEM setting changes.

2.2 Higher order multilevel discretization

Let $\mathbf{\Omega} \subset \mathbb{R}^d$ where $d = 2, 3$ be some polyhedral domain which is not necessarily convex. The space of square integrable scalar valued functions is

$$\mathbf{L}^2(\mathbf{\Omega}) := \Big\{ f : \mathbf{\Omega} \to \mathbb{R}, \qquad \|f\|^2_{\mathbf{L}^2(\mathbf{\Omega})} := \int_{\mathbf{\Omega}} \big|f(\mathbf{x})\big|^2 d\mathbf{x} < \infty \Big\}. \tag{2.13}$$

The Sobolev space on $\mathbf{\Omega}$ for a non-negative integer k is

$$\mathbf{H}^k(\mathbf{\Omega}) := \Big\{ f \in \mathbf{L}^2(\mathbf{\Omega}) : \ \big\|\partial^{\boldsymbol{\alpha}} f\big\|_{\mathbf{L}^2(\mathbf{\Omega})} < \infty \quad \text{for all} \quad |\boldsymbol{\alpha}| \le k \Big\} \tag{2.14}$$

where the differentiation $\partial^{\boldsymbol{\alpha}} f$ is interpreted in the sense of distribution. The Sobolev space $\mathbf{H}^k(\mathbf{\Omega})$ is endowed with the norm

$$\big\|f\big\|^2_{\mathbf{H}^k(\mathbf{\Omega})} := \sum_{|\boldsymbol{\alpha}| \le k} \big\|\partial^{\boldsymbol{\alpha}} f\big\|^2_{\mathbf{L}^2(\mathbf{\Omega})}. \tag{2.15}$$

Consider a nested sequence of meshes of the same domain $\mathbf{\Omega}$:

$$\mathcal{M}^{(0)} \subset \mathcal{M}^{(1)} \subset \cdots \subset \mathcal{M}^{(\ell)} \subset \cdots \subset \mathcal{M}^{(L)}. \tag{2.16}$$

Each mesh $\mathcal{M}^{(\ell)}$ is composed of triangular or tetrahedral elements admitting the next properties: (1)a nonempty intersection of two different elements $T_i, T_j \in \mathcal{M}^{(\ell)}$ is either a common node, a complete edge or a complete triangular face (for $d = 3$), (2)we have the covering $\mathbf{\Omega} = \bigcup_{T \in \mathcal{M}^{(\ell)}} T$. For an element $T \in \mathcal{M}^{(\ell)}$, we denote

$$\begin{aligned} h(T) &:= \operatorname{diam}(T) = \sup\Big\{ |\mathbf{x} - \mathbf{y}|,\, \mathbf{x}, \mathbf{y} \in T \Big\}, \\ \rho(T) &:= \text{supremum of the diameters of all balls contained in } T, \\ \sigma(T) &:= h(T)/\rho(T) = \text{aspect ratio of } T. \end{aligned}$$

We assume quasi-uniformity in the sense that there exists a constant $\sigma_0 > 0$ which is independent of ℓ such that

$$\sigma(T) \le \sigma_0 < \infty \qquad \text{for all} \qquad T \in \mathcal{M}^{(\ell)}, \quad \ell = 0, ..., L. \tag{2.17}$$

By denoting

$$\mathcal{M}^{(\ell)} = \{ T_i^{(\ell)} : \ i = 1, ..., N_\ell \} \tag{2.18}$$

where $T_i^{(\ell)}$ are triangular or tetrahedral elements for the 2D and 3D cases respectively, we have

$$h^{(\ell)} := \max_{i=1,...,N_\ell} h(T_i^{(\ell)}) \tag{2.19}$$

$$h^{(0)} \ge h^{(1)} \ge \cdots \ge h^{(\ell)} \ge \cdots \ge h^{(L)} \equiv h. \tag{2.20}$$

Denote by $\mathcal{N}^{(\ell)}$ and $\mathcal{J}^{(\ell)}$ the set of all nodes and the set of internal nodes of $\mathcal{M}^{(\ell)}$ respectively. The patch centered at a node $n \in \mathcal{N}^{(\ell)}$ with respect to the mesh $\mathcal{M}^{(\ell)}$ is defined as

$$\omega_n^{(\ell)} := \cup\{T \in \mathcal{M}^{(\ell)} : \, n \in T\} \tag{2.21}$$

and let $\mathcal{N}_n^{(\ell)}$ denote the set of nodes of $\omega_n^{(\ell)}$. We define the patch size and the patch of second layer as

$$\begin{aligned} h_n^{(\ell)} &:= \operatorname{diam}(\omega_n^{(\ell)}) && (2.22)\\ \omega_{n,2}^{(\ell)} &:= \cup\big\{T \in \mathcal{M}^{(\ell)} : \, \overline{T} \cap \overline{\omega}_n^{(\ell)} \neq \emptyset\big\}. && (2.23) \end{aligned}$$

The node sets $\Lambda^{(\ell)}$ are defined as follows:

$$\begin{aligned} \Lambda^{(0)} &:= \mathcal{J}^{(0)} && (2.24)\\ \Lambda^{(\ell)} &:= \Big[\cup\{\mathcal{N}_n^{(\ell)} : \quad n \in (\mathcal{J}^{(\ell)} \setminus \mathcal{J}^{(\ell-1)})\}\Big] \cap \mathcal{J}^{(\ell)} \qquad \forall \ell \geq 1. && (2.25) \end{aligned}$$

The members of $\Lambda^{(\ell)}$ for $\ell \geq 1$ consist of two categories of nodes: (1)the newly created internal nodes of $\mathcal{M}^{(\ell)}$ which are not in $\mathcal{M}^{(\ell-1)}$ and (2)the internal nodes of $\mathcal{M}^{(\ell)}$ which are connected by some edges with nodes from the first category (see Fig. 2).

For ease of presentation, we first concentrate on the Poisson problem with homogeneous Dirichlet boundary condition in this section. Thus, we consider the bilinear form $a(u,v) := \langle \nabla u, \nabla v \rangle_{\mathbf{\Omega}}$. For a given degree $p \geq 1$, the higher order FEM setting uses the next finite dimensional subspace of $\mathbf{H}_0^1(\mathbf{\Omega})$ on the highest level

$$\mathcal{S}_h^p(\mathbf{\Omega}) := \big\{u \in \mathcal{C}^0(\mathbf{\Omega}) \cap \mathbf{H}_0^1(\mathbf{\Omega}) : \quad u \in \mathcal{P}_p(T) \quad \forall\, T \in \mathcal{M}^{(L)}\big\} \tag{2.26}$$

where $\mathcal{P}_p(T)$ denotes the space of polynomials in T spanned by the monomials $x_1^{i_1} \cdots x_d^{i_d}$ where $0 \leq i_1 + \cdots + i_d \leq p$. Constructions of flexible basis functions for $\mathcal{S}_h^p(\mathbf{\Omega})$ are found in [41, 42, 5, 4, 23, 28, 29] while efficient FEM-implementations using Bernstein-Bézier are found in [3, 2]. The basis functions are usually categorized as vertex modes, edge modes, face modes (in 3D only) and internal bubble modes.

Corresponding to the nested meshes (2.16), let $\mathcal{V}^{(\ell)}$ denote the piecewise linear FE-subspaces satisfying

$$\mathcal{V}^{(0)} \subset \mathcal{V}^{(1)} \subset \cdots \subset \mathcal{V}^{(\ell)} \subset \cdots \subset \mathcal{V}^{(L)}. \tag{2.27}$$

On level $\ell = 0, ..., L$, the linear shape function centered at the node $n \in \mathcal{J}^{(\ell)}$ with respect to the mesh $\mathcal{M}^{(\ell)}$ is denoted by $\phi_n^{(\ell)}$. Further, we introduce the next linear spaces

$$\begin{aligned} \mathcal{V}_n^{(\ell)} &:= \operatorname{span}\big\{\phi_n^{(\ell)}\big\} \quad \text{for} \quad \ell = 0, ..., L \quad \text{and} \quad n \in \Lambda^{(\ell)} && (2.28)\\ \mathcal{V}(\Lambda^{(\ell)}) &:= \operatorname{span}\big\{\phi_n^{(\ell)} : \, n \in \Lambda^{(\ell)}\big\} \subset \mathcal{V}^{(\ell)}. && (2.29) \end{aligned}$$

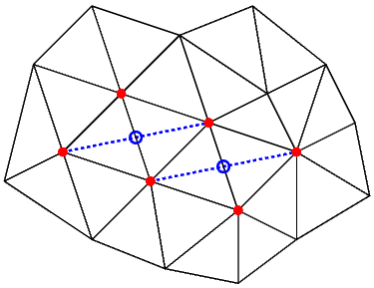


Figure 2: Newly created nodes in a mesh.

For the polynomial degree p and a node $n \in \mathcal{J}^{(L)}$ on the finest level, introduce

$$\mathcal{V}_{(p,n)} := \{u \in S_h^p(\mathbf{\Omega}) : \text{supp}(u) \subset \omega_n^{(L)}\} \tag{2.30}$$

which is defined on the finest mesh $\mathcal{M}^{(L)}$. In addition, we will need the following Ritz projections $P^{(\ell)}u \in \mathcal{V}^{(\ell)}$, $P_n^{(\ell)}u \in \mathcal{V}_n^{(\ell)}$ and $P_{(p,n)}u \in \mathcal{V}_{(p,n)}$:

$$a(P^{(\ell)}u, v) = a(u, v) \quad \forall v \in \mathcal{V}^{(\ell)} \quad \text{for} \quad \ell = 0, ..., L, \tag{2.31}$$

$$a(P_n^{(\ell)}u, v) = a(u, v) \quad \forall v \in \mathcal{V}_n^{(\ell)} \quad \text{for} \quad \ell = 0, ..., L, \quad n \in \Lambda^{(\ell)}, \tag{2.32}$$

$$a(P_{(p,n)}u, v) = a(u, v) \quad \forall v \in \mathcal{V}_{(p,n)}. \tag{2.33}$$

Our purpose is to investigate the following ASM operator

$$P := \sum_{\ell=0}^{L} \sum_{n \in \Lambda^{(\ell)}} P_n^{(\ell)} + \sum_{n \in \mathcal{J}^{(L)}} P_{(p,n)}. \tag{2.34}$$

3 Preconditioner lower bound

This section describes the lower bound inequality from (2.8). From here onward, we use the usual shorthand $X \precsim Y$ if there is a constant c such that $X \leq cY$ in which c is independent on h, p and L. In addition, $X \simeq Y$ amounts to $X \precsim Y \precsim X$. With regard to the ASM operator (2.34), we define the hierarchical piecewise linear (HPL) operator

$$P^{\mathbf{HPL}} := \sum_{\ell=0}^{L} \sum_{n \in \Lambda^{(\ell)}} P_n^{(\ell)} \tag{3.35}$$

which corresponds to the linear part of (2.34). Introduce the next local higher order (LHO) operator of degree $p \geq 1$ which corresponds to the final term of (2.34)

$$P_p^{\mathbf{LHO}} := \sum_{n \in \mathcal{J}^{(L)}} P_{(p,n)}. \tag{3.36}$$

For a node n on a mesh $\mathcal{M}$, recall that ϕ_n denote the piecewise linear hat function centered at n. Introduce the level sets [39]:

$$\gamma_n(s) := \{\mathbf{x} \in \omega_n : \phi_n(\mathbf{x}) = s\} \quad \text{for} \quad s \in [0,1], \tag{3.37}$$

$$\gamma_n(\mathbf{x}) := \gamma_n(\phi_n(\mathbf{x})) \quad \text{for} \quad \mathbf{x} \in \omega_n \tag{3.38}$$

such that $\gamma_n(0)$ coincides with the patch boundary $\partial\omega_n$. The spider averaging operator [39] for an integrable function u is defined as

$$(\Pi_n u)(\mathbf{x}) := \frac{1}{|\gamma_n(\mathbf{x})|} \int_{\gamma_n(\mathbf{x})} u(\mathbf{y}) d\mathbf{y} \qquad \text{for} \quad \mathbf{x} \in \omega_n \tag{3.39}$$

where $|\gamma_n(\mathbf{x})|$ denotes the measure of the set $\gamma_n(\mathbf{x})$. Since Π_n is not necessarily zero on $\partial\omega_n$, a correction term is added:

$$\widetilde{\Pi}_n u := \Pi_n u - (\Pi_n u)|_{\partial\omega_n}(1 - \phi_n) \tag{3.40}$$

in order to have homogeneous boundary values. The next properties are found in [39].

PROPERTIES. *One has the boundedness*

$$\|\Pi_n u\|_{\mathbf{L}_2(\omega_n)} \precsim \|u\|_{\mathbf{L}_2(\omega_n)}, \tag{3.41}$$

$$\|\nabla(\Pi_n u)\|_{\mathbf{L}_2(\omega_n)} \precsim \|\nabla u\|_{\mathbf{L}_2(\omega_n)}. \tag{3.42}$$

If the function u is continuous, then

$$(\Pi_n u)(n) = (\widetilde{\Pi}_n u)(n) = n. \tag{3.43}$$

For $r_n(\mathbf{x}) := |\mathbf{x} - n|$, one has

$$\left\| r_n^{-1}(\phi_n u - \widetilde{\Pi}_n u) \right\|_{\mathbf{L}_2(\omega_n)} \precsim \|\nabla u\|_{\mathbf{L}_2(\omega_n)}. \tag{3.44}$$

Since we will construct multilevel Scott-Zhang operators, let us briefly recall some definitions [40] on a mesh $\mathcal{M}$. The original Scott-Zhang operator is defined for higher polynomial degrees but we need it only for the piecewise linear setting in dimension d. For each node $n_i \in \mathcal{N}$, choose a $(d-1)$-simplex σ_i according to the next criteria. (1)If $n_i \notin \partial\mathbf{\Omega}$, pick any $(d-1)$-simplex σ_i such that

$$n_i \in \overline{\sigma}_i. \tag{3.45}$$

(2)If $n_i \in \partial\mathbf{\Omega}$, pick a $(d-1)$-simplex σ_i such that

$$n_i \in \overline{\sigma}_i \quad \text{and} \quad \sigma_i \subset \partial\mathbf{\Omega}. \tag{3.46}$$

The choice of the simplex σ_i is therefore not unique. Let $\{n_{i,j}\}_{j=1}^d$ be the nodal points of σ_i and $\{\phi_{i,j}\}_{j=1}^d$ their corresponding basis functions. Their dual basis $\{\psi_{i,j}\}_{j=1}^d$ in σ_i satisfies

$$\int_{\sigma_i} \phi_{i,j}(\zeta)\,\psi_{i,k}(\zeta)d\zeta = \delta_{j,k} \quad \text{for} \quad j,k = 1,...,d. \tag{3.47}$$

Denote $n_i \equiv n_{i,1}$ and $\psi_i \equiv \psi_{i,1}$. The Scott-Zhang operator on an arbitrary mesh $\mathcal{M}$ is given by

$$(\mathcal{SZ})v := \sum_{n_i \in \mathcal{N}} \alpha_i \phi_{n_i} \quad \text{where} \quad \alpha_i := \int_{\sigma_i} \psi_i(\zeta)v(\zeta)d\zeta. \tag{3.48}$$

3.1 Construction

This section is occupied by the construction of the decomposition of a function with respect to the spaces $\mathcal{V}(\Lambda^{(\ell)})$. Denote by $\widetilde{\Pi}_n^{(\ell)}$ the spider averaging operator (3.40) with respect to the node n in the mesh $\mathcal{M}^{(\ell)}$ from (2.16).

LEMMA. *Consider a polygonal domain $\mathbf{\Omega}$ which is not necessarily convex. For any $u_h \in \mathcal{V}^{(L)}$, there exist*

$$u^{(\ell)} \in \mathcal{V}^{(\ell)}, \qquad v^{(\ell)} \in \mathcal{V}(\Lambda^{(\ell+1)}) \tag{3.49}$$

such that

$$u_h = u^{(0)} + \sum_{\ell=0}^{L-1} v^{(\ell)} \tag{3.50}$$

$$\sum_{n \in \Lambda^{(\ell+1)}} \left|\widetilde{\Pi}_n^{(\ell+1)}[v^{(\ell)}]\right|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} \precsim |u^{(\ell+1)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} + |v^{(\ell)}|^2_{\mathbf{H}^1(\mathbf{\Omega})}. \tag{3.51}$$

In particular,

$$\mathcal{V} = \mathcal{V}^{(0)} + \mathcal{V}(\Lambda^{(1)}) + \cdots + \mathcal{V}(\Lambda^{(L)}) \tag{3.52}$$

holds.

PROOF. Let $(\mathcal{SZ})^{(\ell)} : \mathbf{L}_2(\mathbf{\Omega}) \longrightarrow \mathcal{V}^{(\ell)}$ denote the Scott-Zhang operator w.r.t. the mesh $\mathcal{M}^{(\ell)}$. Set $u^{(L)} := u_h \in \mathcal{V}^{(L)}$ and define recursively for $\ell = L-1, L-2, ..., 0$

$$u^{(\ell)} := (\mathcal{SZ})^{(\ell)} u^{(\ell+1)} \in \mathcal{V}^{(\ell)} \tag{3.53}$$

$$v^{(\ell)} := u^{(\ell+1)} - (\mathcal{SZ})^{(\ell)} u^{(\ell+1)} \in \mathcal{V}^{(\ell+1)}. \tag{3.54}$$

For the nested sequence of meshes $\mathcal{M}^{(\ell)}$, the following construction of $(\mathcal{SZ})^{(\ell)}$ aims at ensuring that $v^{(\ell)} \in \mathcal{V}(\Lambda^{(\ell+1)})$. Since the construction of the Scott-Zhang operator is not unique, the process relies on the choice of the $(d-1)$-simplices σ_i. Concerning the construction of $(\mathcal{SZ})^{(0)}$ on the coarsest mesh $\mathcal{M}^{(0)}$, for a node n_i in $\mathcal{J}^{(0)}$, choose σ_i according to (3.45)–(3.48). Suppose $(\mathcal{SZ})^{(\ell-1)}$ has been constructed on the mesh

$\mathcal{M}^{(\ell-1)}$ and let us construct $(\mathcal{SZ})^{(\ell)}$ on the mesh $\mathcal{M}^{(\ell)}$. Consider a node $n_i \in \mathcal{J}^{(\ell)}$ and distinguish two cases. (1)If $n_i \notin \Lambda^{(\ell)}$, thus necessarily $n_i \in \mathcal{M}^{(\ell-1)}$, keep the choice of σ_i as in the construction of $(\mathcal{SZ})^{(\ell-1)}$. (2)If $n_i \in \Lambda^{(\ell)}$, i.e. n_i is either a newly created node or a node adjacent to a newly created one, choose σ_i according to (3.45)–(3.48) with respect to the mesh $\mathcal{M}^{(\ell)}$. Repeat that procedure recursively until $(\mathcal{SZ})^{(L)}$ is constructed. The functions $u^{(\ell+1)}$ and $(\mathcal{SZ})^{(\ell)}u^{(\ell+1)}$ coincide at all nodes of $\mathcal{N}^{(\ell+1)} \setminus \Lambda^{(\ell+1)}$ according to (3.45)–(3.48). Hence, $v^{(\ell)} = u^{(\ell+1)} - (\mathcal{SZ})^{(\ell)}u^{(\ell+1)}$ belongs to $\mathcal{V}(\Lambda^{(\ell+1)}) \subset \mathcal{V}^{(\ell+1)}$. At the top level L, one has $(\mathcal{SZ})^{(L)}u_h \equiv u_h$ because the Scott-Zhang operator preserves functions in $\mathcal{V}^{(L)} \equiv \mathcal{V}$. We have

$$u_h = u^{(L-1)} + v^{(L-1)} \tag{3.55}$$

$$= u^{(L-2)} + [v^{(L-2)} + v^{(L-1)}] = \cdots = u^{(0)} + \sum_{\ell=0}^{L-1} v^{(\ell)}. \tag{3.56}$$

As a consequence, one has the space decomposition

$$\mathcal{V} = \dot{+}_{\ell=0}^{L} \mathcal{V}(\Lambda^{(\ell)}). \tag{3.57}$$

Since the local spider averaging operator $\widetilde{\Pi}_n^{(\ell)}$ preserves the nodal values according to (3.43), we have

$$v^{(\ell)} = \sum_{n \in \Lambda^{(\ell+1)}} \widetilde{\Pi}_n^{(\ell+1)}[v^{(\ell)}] \in \mathcal{V}(\Lambda^{(\ell+1)}). \tag{3.58}$$

From the definition of the correction term (3.40) for each node $n \in \Lambda^{(\ell+1)}$

$$\big|\widetilde{\Pi}_n^{(\ell+1)}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} = \big|\Pi_n^{(\ell+1)}[v^{(\ell)}] - (1-\phi_n^{(\ell+1)})\Pi_n^{(\ell+1)}\big|_{\partial\omega_n^{(\ell+1)}}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})}. \tag{3.59}$$

Since $\Pi_n^{(\ell+1)}\big|_{\partial\omega_n^{(\ell+1)}}[v^{(\ell)}]$ is constant on the whole $\partial\omega_n^{(\ell+1)} = \gamma_n(0)$, one has

$$\big|\widetilde{\Pi}_n^{(\ell+1)}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} \le \big|\Pi_n^{(\ell+1)}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} + \tag{3.60}$$

$$\big|\Pi_n^{(\ell+1)}\big|_{\partial\omega_n^{(\ell+1)}}[v^{(\ell)}]\big|^2 \big|1-\phi_n^{(\ell+1)}\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})}. \tag{3.61}$$

From $|(1-\phi_n^{(\ell+1)})|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} \precsim 1$, obtain

$$\begin{aligned}\big|\widetilde{\Pi}_n^{(\ell+1)}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} &\precsim \big|\Pi_n^{(\ell+1)}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} + \big\|\Pi_n^{(\ell+1)}[v^{(\ell)}]\big\|^2_{\mathbf{L}^\infty(\omega_n^{(\ell+1)})} \\ &\precsim \big|\Pi_n^{(\ell+1)}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} + \frac{1}{\operatorname{diam}(\omega_n^{(\ell+1)})^2}\big\|\Pi_n^{(\ell+1)}[v^{(\ell)}]\big\|^2_{\mathbf{L}^2(\omega_n^{(\ell+1)})}.\end{aligned}$$

By using (3.41) and (3.42) obtain

$$\begin{aligned}\big|\widetilde{\Pi}_n^{(\ell+1)}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} &\precsim \big\|\nabla[v^{(\ell)}]\big\|^2_{\mathbf{L}^2(\omega_n^{(\ell+1)})} + \frac{1}{[h_n^{(\ell+1)}]^2}\|v^{(\ell)}\|^2_{\mathbf{L}^2(\omega_n^{(\ell+1)})} \\ &= \big\|\nabla[v^{(\ell)}]\big\|^2_{\mathbf{L}^2(\omega_n^{(\ell+1)})} + \frac{1}{[h_n^{(\ell+1)}]^2}\big\|u^{(\ell+1)} - (\mathcal{SZ})^{(\ell)}u^{(\ell+1)}\big\|^2_{\mathbf{L}^2(\omega_n^{(\ell+1)})}.\end{aligned}$$

Let $\eta = \eta(n) \in \mathcal{J}^{(\ell)}$ be such that $\omega_n^{(\ell+1)} \subset \omega_\eta^{(\ell)}$ and $h_n^{(\ell+1)} \simeq h_\eta^{(\ell)}$. One has

$$\big|\widetilde{\Pi}_n^{(\ell+1)}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} \precsim \big\|\nabla[v^{(\ell)}]\big\|^2_{\mathbf{L}^2(\omega_n^{(\ell+1)})} + \frac{1}{[h_\eta^{(\ell)}]^2}\big\|u^{(\ell+1)} - (\mathcal{SZ})^{(\ell)}u^{(\ell+1)}\big\|^2_{\mathbf{L}^2(\omega_\eta^{(\ell)})}. \tag{3.62}$$

Use the local approximation property [40] of the Scott-Zhang operator $(\mathcal{SZ})^{(\ell)}$ in the patch $\omega_\eta^{(\ell)}$

$$\big|\widetilde{\Pi}_n^{(\ell+1)}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} \precsim \big|v^{(\ell)}\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} + \big|u^{(\ell+1)}\big|^2_{\mathbf{H}^1(\omega_{\eta,2}^{(\ell)})}. \tag{3.63}$$

Taking the sum over the nodes of $\Lambda^{(\ell+1)}$,

$$\sum_{n\in\Lambda^{(\ell+1)}} \big|\widetilde{\Pi}_n^{(\ell+1)}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} \precsim \sum_{n\in\Lambda^{(\ell+1)}} |v^{(\ell)}|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} + \sum_{n\in\Lambda^{(\ell+1)}} |u^{(\ell+1)}|^2_{\mathbf{H}^1(\omega_{\eta(n),2}^{(\ell)})}. \tag{3.64}$$

Due to the quasi-uniformity of the meshes, the patches mutually intersect only in a finite number of times. Deduce

$$\sum_{n\in\Lambda^{(\ell+1)}} \big|\widetilde{\Pi}_n^{(\ell+1)}[v^{(\ell)}]\big|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} \precsim |v^{(\ell)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} + |u^{(\ell+1)}|^2_{\mathbf{H}^1(\mathbf{\Omega})}. \tag{3.65}$$

■

3.2 Stability analysis

We will study now the stability of the function decomposition which was constructed in the previous section. In particular, that provides the ASM lower bound in (2.8).

THEOREM. *Consider a polygonal domain $\mathbf{\Omega}$ which is not necessarily convex. Suppose* $\mathrm{Card}(\mathcal{N}^{(0)}) = \mathcal{O}(1)$. *For any $u_{hp} \in \mathcal{V}_h^p(\mathbf{\Omega})$, one has*

$$a(u_{hp}, u_{hp}) \leq Ca((P^{\mathbf{HPL}} + P^{\mathbf{LHO}})u_{hp}, u_{hp}). \tag{3.66}$$

where C is a constant independent of the mesh sizes $h^{(\ell)}$, the polynomial degree $p \geq 1$ and the maximal level L.

PROOF. Consider any $u^{(L)} \equiv u_h \in \mathcal{V}^{(L)}$. A hierarchical tree is immediately defined by the decomposition (3.53) and (3.54). Process the hierarchical tree (see Fig. 3) bottom-up

$$I^{(0)} := |u^{(0)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} + \sum_{n\in\Lambda^{(1)}} \big|\widetilde{\Pi}_n^{(1)}[v^{(0)}]\big|^2_{\mathbf{H}^1(\omega_n^{(1)})}. \tag{3.67}$$

By using (3.51) for $\ell = 0$, obtain

$$\begin{aligned} I^{(0)} &= |u^{(0)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} + |v^{(0)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} + |u^{(1)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} && (3.68)\\ &= \big|(\mathcal{SZ})^{(0)}u^{(1)}\big|^2_{\mathbf{H}^1(\mathbf{\Omega})} + \big|u^{(1)} - (\mathcal{SZ})^{(0)}u^{(1)}\big|^2_{\mathbf{H}^1(\mathbf{\Omega})} + |u^{(1)}|^2_{\mathbf{H}^1(\mathbf{\Omega})}. && (3.69)\end{aligned}$$

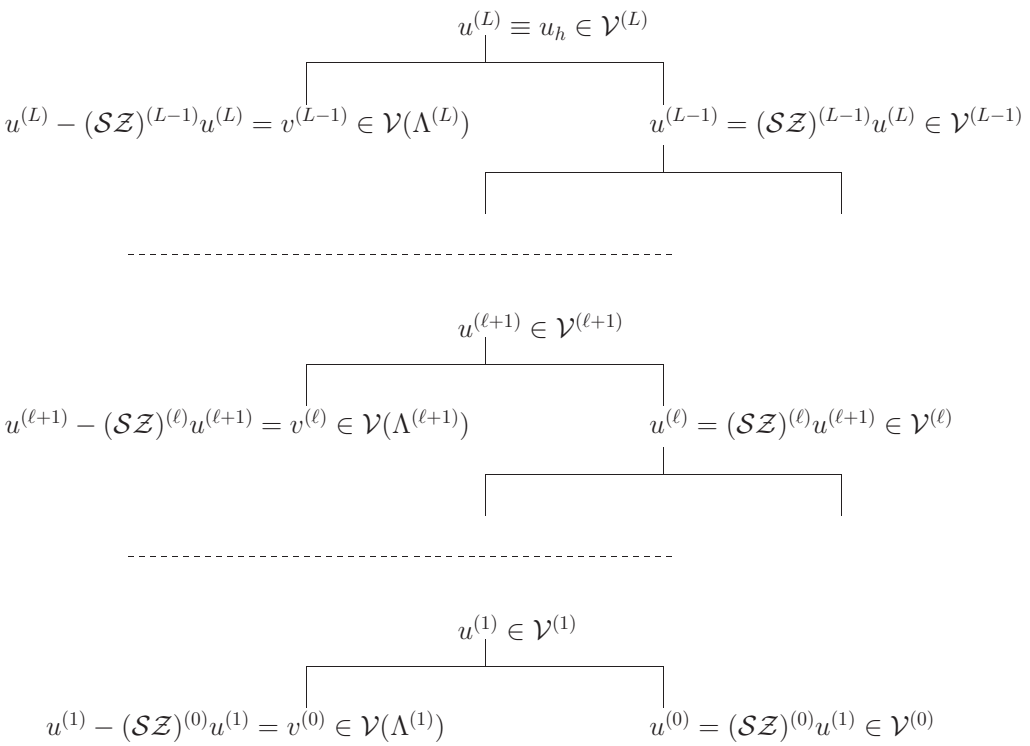


Figure 3: Binary tree of hierarchical function decomposition.

From the global stability [40] of the Scott-Zhang operator $(\mathcal{SZ})^{(0)}$, obtain therefore

$$I^{(0)} \precsim |u^{(1)}|^2_{\mathbf{H}^1(\mathbf{\Omega})}. \tag{3.70}$$

Define for $\ell \leq L-1$

$$I^{(\ell)} := |u^{(0)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} + \sum_{\lambda=1}^{\ell+1} \sum_{n\in\Lambda^{(\lambda)}} \left|\widetilde{\Pi}_n^{(\lambda)}[v^{(\lambda-1)}]\right|^2_{\mathbf{H}^1(\omega_n^{(\lambda)})}. \tag{3.71}$$

Suppose by induction that $I^{(\ell-1)} \precsim |u^{(\ell)}|^2_{\mathbf{H}^1(\mathbf{\Omega})}$. We have by using (3.51) of the former lemma and the stability of $(\mathcal{SZ})^{(\ell)}$

$$\begin{aligned} I^{(\ell)} &= I^{(\ell-1)} + \sum_{n\in\Lambda^{(\ell+1)}} \left|\widetilde{\Pi}_n^{(\ell+1)}[v^{(\ell)}]\right|^2_{\mathbf{H}^1(\omega_n^{(\ell+1)})} && (3.72)\\ &\precsim I^{(\ell-1)} + |v^{(\ell)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} + |u^{(\ell+1)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} && (3.73)\\ &\precsim |u^{(\ell)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} + |v^{(\ell)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} + |u^{(\ell+1)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} \precsim |u^{(\ell+1)}|^2_{\mathbf{H}^1(\mathbf{\Omega})}. && (3.74) \end{aligned}$$

Processing in the same manner till $I^{(L-1)}$, obtain

$$|u^{(0)}|^2_{\mathbf{H}^1(\mathbf{\Omega})} + \sum_{\ell=1}^{L} \sum_{n\in\Lambda^{(\ell)}} \left|\widetilde{\Pi}_n^{(\ell)}[v^{(\ell-1)}]\right|^2_{\mathbf{H}^1(\omega_n^{(\ell)})} \precsim |u^{(L)}|^2_{\mathbf{H}^1(\mathbf{\Omega})}. \tag{3.75}$$

Define $u_n^{(\ell)} := \widetilde{\Pi}_n^{(\ell)}[v^{(\ell-1)}] \in \mathcal{V}_n^{(\ell)}$ and obtain from (3.75)

$$a(u^{(0)}, u^{(0)}) + \sum_{\ell=1}^{L} \sum_{n \in \Lambda^{(\ell)}} a(u_n^{(\ell)}, u_n^{(\ell)}) \precsim a(u^{(L)}, u^{(L)}). \tag{3.76}$$

In particular, since $\mathrm{Card}(\mathcal{N}^{(0)}) = \mathcal{O}(1)$, one has

$$\sum_{\ell=0}^{L} \sum_{n \in \Lambda^{(\ell)}} a(u_n^{(\ell)}, u_n^{(\ell)}) \precsim a(u^{(L)}, u^{(L)}). \tag{3.77}$$

The next procedure is treated as in [23] where the function $u_{hp} \in S_h^p(\mathbf{\Omega})$ is decomposed in the highest level $\mathcal{M}^{(L)}$ using [39] such that $u_{hp} = u^{(L)} + u^{\mathbf{LHO}}$ in which

$$u^{(L)} \in \mathcal{V}^{(L)}, \qquad u^{\mathbf{LHO}} \in \mathop{+}_{n \in \mathcal{J}^{(L)}} \mathcal{V}_{(p,n)}\,, \tag{3.78}$$

$$a(u^{(L)}, u^{(L)}) + a(u^{\mathbf{LHO}}, u^{\mathbf{LHO}}) \precsim a(u_{hp}, u_{hp}). \tag{3.79}$$

By combining them with (3.77), a function decomposition is found such that

$$\sum_{\ell=0}^{L} \sum_{n \in \Lambda^{(\ell)}} a(u_n^{(\ell)}, u_n^{(\ell)}) + a(u^{\mathbf{LHO}}, u^{\mathbf{LHO}}) \precsim a(u_{hp}, u_{hp}). \tag{3.80}$$

From the standard LIONS' lemma, deduce

$$a(u_{hp}, u_{hp}) \precsim a\big((P^{\mathbf{HPL}} + P^{\mathbf{LHO}})u_{hp}, u_{hp}\big). \tag{3.81}$$

■

REMARK. *The above result is still valid if* $\mathrm{Card}(\mathcal{N}^{(0)})$ *is not necessarily* $\mathcal{O}(1)$ *but the coarsest mesh* $\mathcal{M}^{(0)}$ *has to be treated as a particular case and (2.34) becomes*

$$P = P^{(0)} + \sum_{\ell=1}^{L} \sum_{n \in \Lambda^{(\ell)}} P_n^{(\ell)} + \sum_{n \in \mathcal{J}^{(L)}} P_{(p,n)}. \tag{3.82}$$

■

REMARK. *We briefly want to recall some typical method (see e.g. [46]) for decomposing a function* $u_h \in \mathcal{V}^{(L)}$ *which is the FE-approximation of*

$$a(u, v) = \langle f, v \rangle_{\mathbf{H}^{-1}(\mathbf{\Omega}) \times \mathbf{H}_0^1(\mathbf{\Omega})}. \tag{3.83}$$

The function is represented as

$$u_h = \sum_{\ell=0}^{L} \sum_{n \in D^{(\ell)}} u_n^{(\ell)} \qquad s.t. \qquad a(u_h, u_h) \precsim \sum_{\ell=0}^{L} \sum_{n \in D^{(\ell)}} a(u_n^{(\ell)}, u_n^{(\ell)}) \tag{3.84}$$

in which $D^{(\ell)}$ are related to some domains on level ℓ. The main difference between the method in this section and the usual methods requiring $\mathbf{H}^2$-smoothness of u is that they first decompose u_h on the levels:

$$u_h = u^{(0)} + \cdots + u^{(L)} \quad \textit{such that} \quad \sum_{\ell=0}^{L} |u^{(\ell)}|^2_{\mathbf{H^1(\Omega)}} \precsim |u_h|^2_{\mathbf{H^1(\Omega)}}. \tag{3.85}$$

On each level ℓ, the function $u^{(\ell)}$ is further decomposed into $v_n^{(\ell)}$ (by using for e.g. a partition of unity) such that

$$\sum_{n} |v_n^{(\ell)}|^2_{\mathbf{H^1(\Omega)}} \precsim |u^{(\ell)}|^2_{\mathbf{H^1(\Omega)}}. \tag{3.86}$$

That is obtained for example from Aubin-Nitsche trick to derive an $\mathbf{H}^1$-bound from the $\mathbf{H}^2$-smoothness which is naturally guaranteed if the domain is convex. The idea proposed in this section applies if the solution u is non-smooth and the domain is not necessarily convex. As opposed to (3.86), the presented estimate uses

$$\sum_{n} |v_n^{(\ell)}|^2_{\mathbf{H^1(\Omega)}} \precsim |v^{(\ell)}|^2_{\mathbf{H^1(\Omega)}} + |u^{(\ell+1)}|^2_{\mathbf{H^1(\Omega)}} \tag{3.87}$$

where the second term on the right hand side involves an expression on the upper level $(\ell+1)$. In addition, the regions $\Lambda^{(\ell)}$ are not constrained to any size restriction and no special type of mesh distribution is required either. The approach does not even necessitate that u_h be a FE-approximation of any boundary value problem.

■

4 Preconditioner upper bound

4.1 Reordering according to the patch diameters

At first sight, the presented approach has similarities with [46]. The main difference from [46] is that (1)the elements of $\mathcal{M}^{(\ell)}$ are not necessarily of the same size $\mathcal{O}(h^\ell)$, (2)for each level ℓ, the union of $\omega_n^{(\ell)}$ for $n \in \Lambda^{(\ell)}$ does not cover the whole domain $\mathbf{\Omega}$. Those points make the present method fit with adaptive refinements where only certain regions of $\mathbf{\Omega}$ are successively refined several times. Throughout this document, the next chromatic decomposition is assumed on every level $\ell = 0, 1, ..., L$. One needs at most B colors to group the patches of the mesh $\mathcal{M}^{(\ell)}$ where $B \neq B(\ell, L)$. Since the

patches are centered at nodes, one uses the nodes as subset indices

$$X_k^{(\ell)} := \Big\{ n \in \mathcal{J}^{(\ell)} : \quad \omega_n^{(\ell)} \text{ has chromatic index } k \Big\} \quad k = 1, ..., B \tag{4.88}$$

$$X_{k_1}^{(\ell)} \cap X_{k_2}^{(\ell)} = \emptyset \quad \forall\, k_1, k_2 = 1, ..., B, \quad k_1 \neq k_2 \tag{4.89}$$

$$\overline{\omega}_{n_1}^{(\ell)} \cap \overline{\omega}_{n_2}^{(\ell)} = \emptyset \quad \forall\, n_1, n_2 \in X_k^{(\ell)}, \quad n_1 \neq n_2. \tag{4.90}$$

The last property means that patches admitting the same chromatic index k form disjoint subsets. Note that, the collection of all patches in all $X_k^{(\ell)}$, $k = 1, ..., B$ covers the whole domain $\mathbf{\Omega}$. Introduce for each level ℓ the space

$$\mathcal{V}(X_k^{(\ell)}) := \text{span}\Big\{ \phi_n^{(\ell)} : \quad n \in X_k^{(\ell)} \Big\} \quad k = 1, ..., B. \tag{4.91}$$

Define the nonnegative integer

$$\mu(\omega_n^{(\ell)}) := \Big\lfloor -\log_2 \text{diam}(\omega_n^{(\ell)}) \Big\rfloor \quad \text{for} \quad n \in \Lambda^{(\ell)}, \quad \ell = 0, ..., L, \tag{4.92}$$

$$\text{diam}(\omega_n^{(\ell)}) = \mathcal{O}\left(2^{-\mu(\omega_n^{(\ell)})}\right). \tag{4.93}$$

From the collection of all patches $\omega_n^{(\ell)}$ where $n \in \Lambda^{(\ell)}$, $\ell = 0, ..., L$, there is a certain maximal level M which is generally different from L such that $\mu(\omega_n^{(\ell)}) = 0, ..., M$. For any level $m = 0, ..., M$, define

$$W^{(m)} := \left\{ \omega_n^{(\ell)} \text{ patch} \;\middle|\; \begin{array}{l} n \in \Lambda^{(\ell)} \quad \ell = 0, ..., L \\ \mu(\omega_n^{(\ell)}) = m \end{array} \right\} \tag{4.94}$$

corresponding to patches of diameter $\mathcal{O}(2^{-m})$. Note that, although two identical patches could appear on two meshes $\mathcal{M}^{(\ell_1)} \subset \mathcal{M}^{(\ell_2)}$, each patch in $W^{(m)}$ does not appear more than once because in the set $\Lambda^{(\ell)}$, only newly created nodes and their nearby nodes are considered. The set of indices on level $m = 0, ..., M$ is

$$\mathcal{R}^{(m)} := \big\{ \mathbf{r} = (\ell, n) : \quad \omega_n^{(\ell)} \text{ patch in } W^{(m)} \big\}. \tag{4.95}$$

For a patch and its corresponding linear shape function w.r.t. $W^{(m)}$, we use the shorthand

$$\omega_{\mathbf{r}} \equiv \omega_n^{(\ell)} \quad \text{and} \quad \phi_{\mathbf{r}} \equiv \phi_n^{(\ell)} \qquad \mathbf{r} = (\ell, n) \in \mathcal{R}^{(m)} \tag{4.96}$$

where $\mathbf{r}$ is a bold letter, or as a reminder that it is on level $m = 0, ..., M$ we write $\omega_{\mathbf{r}}^{(m)}$, $\phi_{\mathbf{r}}^{(m)}$. For every level $m = 0, ..., M$, define the subspace

$$\mathcal{U}_{\mathbf{r}}^{(m)} := \text{span}\{\phi_{\mathbf{r}}\} \quad \text{where} \quad \mathbf{r} = (\ell, n) \in \mathcal{R}^{(m)} \tag{4.97}$$

so that $\text{diam}\big[\text{supp}(u)\big] = \mathcal{O}(2^{-m})$ for $u \in \mathcal{U}_{\mathbf{r}}^{(m)}$. Define also the space

$$\mathcal{U}^{(m)} := \text{span}\big\{ \phi_{\mathbf{r}},\, \mathbf{r} \in \mathcal{R}^{(m)} \big\}. \tag{4.98}$$

By introducing

$$\mathcal{Z}^{(m)} := \mathcal{U}^{(0)} + \cdots + \mathcal{U}^{(m)} \quad \text{for} \quad m = 0, ..., M, \tag{4.99}$$

one has the diametric nestedness

$$\mathcal{Z}^{(0)} \subset \cdots \subset \mathcal{Z}^{(m)} \subset \cdots \subset \mathcal{Z}^{(M)}. \tag{4.100}$$

We will use both nestedness (2.16) and (4.100) and we note that $\mathcal{V}^{(L)} \equiv \mathcal{Z}^{(M)}$. The orthogonal projections $Q^{(m)}u \in \mathcal{Z}^{(m)}$ and $Q^{(m)}_{\mathbf{r}}u \in \mathcal{U}^{(m)}_{\mathbf{r}}$ with respect to $a(\bullet, \bullet)$ satisfy

$$a(Q^{(m)}u, v) = a(u, v) \quad \forall v \in \mathcal{Z}^{(m)} \quad \text{for} \quad m = 0, ..., M \tag{4.101}$$

$$a(Q^{(m)}_{\mathbf{r}}u, v) = a(u, v) \quad \forall v \in \mathcal{U}^{(m)}_{\mathbf{r}} \quad \text{for} \quad m = 0, ..., M, \quad \mathbf{r} \in \mathcal{R}^{(m)}. \tag{4.102}$$

We deduce immediately from the definitions that

$$\mathcal{V}^{(L)} = \bigoplus_{\ell=0}^{L} \bigoplus_{n \in \Lambda^{(\ell)}} \mathcal{V}^{(\ell)}_n = \bigoplus_{m=0}^{M} \bigoplus_{\mathbf{r} \in \mathcal{R}^{(m)}} \mathcal{U}^{(m)}_{\mathbf{r}} = \mathcal{Z}^{(M)}, \tag{4.103}$$

$$P^{\mathbf{HPL}} = \sum_{\ell=0}^{L} \sum_{n \in \Lambda^{(\ell)}} P^{(\ell)}_n = \sum_{m=0}^{M} \sum_{\mathbf{r} \in \mathcal{R}^{(m)}} Q^{(m)}_{\mathbf{r}}. \tag{4.104}$$

Consider a fixed level $m = 0, ..., M$. We assume a similar chromatic decomposition as (4.88)–(4.90)

$$\mathcal{C}^{(m)}_s := \Big\{ \mathbf{r} \in \mathcal{R}^{(m)} : \quad \omega_{\mathbf{r}} \text{ has chromatic index } s \Big\} \quad s = 1, ..., B \tag{4.105}$$

$$\mathcal{C}^{(m)}_{s_1} \cap \mathcal{C}^{(m)}_{s_2} = \emptyset \quad \forall\, s_1, s_2 = 1, ..., B, \quad s_1 \neq s_2 \tag{4.106}$$

$$\overline{\omega}^{(m)}_{\mathbf{r}_1} \cap \overline{\omega}^{(m)}_{\mathbf{r}_2} = \emptyset \quad \forall\, \mathbf{r}_1, \mathbf{r}_2 \in \mathcal{C}^{(m)}_s, \quad \mathbf{r}_1 \neq \mathbf{r}_2. \tag{4.107}$$

We define also

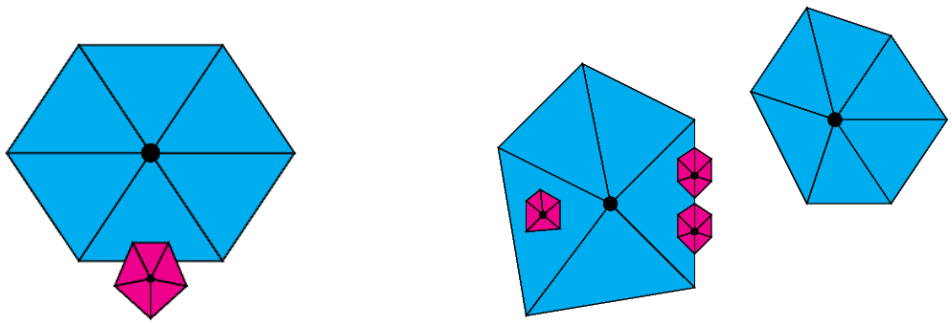
$$Q_{\mathcal{C}^{(m)}_s} := \sum_{\mathbf{r} \in \mathcal{C}^{(m)}_s} Q_{\mathbf{r}}. \text{ Thus,} \quad Q_{\mathcal{C}^{(m)}_s} u \in \mathcal{U}^{(m)} \subset \mathcal{Z}^{(m)}. \tag{4.108}$$

4.2 Multilevel strengthened Cauchy inequality

With regard to the former patch reordering, we want to investigate now a multi-level strengthened Cauchy-Schwarz inequality.

LEMMA. *For $\mu = 0, ..., M$, consider a function $u \in \mathcal{Z}^{(\mu)}$ from the diametrical nestedness (4.100) and let m be such that $\mu \leq m \leq M$. One has for each $s = 1, ..., B$*

$$a(Q_{\mathcal{C}^{(m)}_s} u, u) \precsim \Big(\frac{2^\mu}{2^m}\Big) a\big(Q_{\mathcal{C}^{(m)}_s} u, Q_{\mathcal{C}^{(m)}_s} u\big)^{1/2} a(u, u)^{1/2}. \tag{4.109}$$



(a) (b)

Figure 4: (a)Impossible: some fine elements are partly inside and partly outside a coarse element, (b)Typical situation and chromatic intersections.

PROOF. All patches corresponding to $\mathcal{Z}^{(\mu)}$ have diameters larger than $\mathcal{O}(2^{-\mu})$ according to (4.92) and (4.93). Hence, due to the hierarchical refinements (2.16), there is some $\mathcal{M}^{(\ell)}$ whose smallest patch has diameter $\mathcal{O}(2^{-\mu})$ and all patches corresponding to $\mathcal{Z}^{(\mu)}$ are union of elements of $\mathcal{M}^{(\ell)}$. Thus, $u \in \mathcal{Z}^{(\mu)} \subset \mathcal{V}^{(\ell)}$.

Part 1. Consider first the case $u \in \mathcal{V}(X_k^{(\ell)})$ for some fixed $k = 1, ..., B$ such that $u = \sum_{n \in X_k^{(\ell)}} u_n$. Define $J := a(Q_{\mathcal{C}_s^{(m)}} u, u) = \sum_{n_1, n_2 \in X_k^{(\ell)}} a(Q_{\mathcal{C}_s^{(m)}} u_{n_1}, u_{n_2})$. A fine element of a patch corresponding to $\mathcal{Z}^{(m)}$ is either completely inside or completely outside a coarse element of $\mathcal{M}^{(\ell)}$ (illustrated in Fig. 4). Hence,

$$\operatorname{int}\big[\operatorname{supp}(Q_{\mathcal{C}_s^{(m)}} u_{n_1})\big] \cap \operatorname{int}\big[\operatorname{supp}(u_{n_2})\big] = \operatorname{int}\big[\operatorname{supp}(Q_{\mathcal{C}_s^{(m)}} u_{n_1})\big] \cap \operatorname{int}(\omega_{n_2}^{(\ell)}) = \emptyset. \quad (4.110)$$

As a consequence, $J = \sum_{n \in X_k^{(\ell)}} a(Q_{\mathcal{C}_s^{(m)}} u_n, u_n)$. According to (4.108), one has

$$I_n := a(Q_{\mathcal{C}_s^{(m)}} u_n, u_n) = \sum_{\mathbf{r} \in \mathcal{C}_s^{(m)}} a(Q_{\mathbf{r}}^{(m)} u_n, u_n) \qquad \forall n \in X_k^{(\ell)}. \quad (4.111)$$

Consider the patch $\omega_n^{(\ell)}$. For a patch $\omega_{\mathbf{r}}^{(m)}$ which is inside an element T of $\omega_n^{(\ell)}$, one has $a(Q_{\mathbf{r}}^{(m)} u_n, u_n) = a_T(Q_{\mathbf{r}}^{(m)} u_n, u_n) = 0$ because ∇u_n is constant in T. Hence,

$$I_n = \sum_{\mathbf{r} \in \mathcal{F}_s^{(m)}} a(Q_{\mathbf{r}}^{(m)} u_n, u_n) = a(Q_{\mathcal{F}_s^{(m)}} u_n, u_n) \quad (4.112)$$

where $\mathcal{F}_s^{(m)}$ consists of the indices of the patches in $\mathcal{C}_s^{(m)}$ which intersect the faces (edges in 2D and triangles in 3D) of $\omega_n^{(\ell)}$. Thus, define $F := (\cup_{\mathbf{r}\in\mathcal{F}_s^{(m)}}\omega_{\mathbf{r}}^{(m)}) \cap \omega_n^{(\ell)}$. Deduce

$$\begin{aligned} I_n &= a_F(Q_{\mathcal{F}_s^{(m)}}u_n, u_n) && (4.113)\\ &\leq a_F(Q_{\mathcal{F}_s^{(m)}}u_n, Q_{\mathcal{F}_s^{(m)}}u_n)^{1/2} a_F(u_n,u_n)^{1/2} && (4.114)\\ &\leq a(Q_{\mathcal{F}_s^{(m)}}u_n, Q_{\mathcal{F}_s^{(m)}}u_n)^{1/2} a_F(u_n,u_n)^{1/2} && (4.115)\\ &= a(Q_{\mathcal{F}_s^{(m)}}u_n, u_n)^{1/2} a_F(u_n,u_n)^{1/2} && (4.116)\\ &= a(Q_{\mathcal{C}_s^{(m)}}u_n, u_n)^{1/2} a_F(u_n,u_n)^{1/2}. && (4.117)\end{aligned}$$

Hence, $I_n^{1/2} = a(Q_{\mathcal{C}_s^{(m)}}u_n,u_n)^{1/2} \leq a_F(u_n,u_n)^{1/2}$. As a result,

$$\begin{aligned} I_n &\leq a_F(u_n,u_n) \leq \frac{\mathrm{vol}(F)}{\mathrm{vol}(\omega_n^{(\ell)})} a(u_n,u_n) \leq \frac{\mathrm{vol}[\cup_{\mathbf{r}\in\mathcal{F}_s^{(m)}}\omega_{\mathbf{r}}]}{\mathrm{vol}(\omega_n^{(\ell)})} a(u_n,u_n) && (4.118)\\ &\leq \sum_{\mathbf{r}\in\mathcal{F}_s^{(m)}} \frac{\mathrm{vol}(\omega_{\mathbf{r}})}{\mathrm{vol}(\omega_n^{(\ell)})} a(u_n,u_n) \leq \mathrm{card}(\mathcal{F}_s^{(m)}) \frac{2^{-md}}{2^{-\mu d}} a(u_n,u_n). && (4.119)\end{aligned}$$

By using mesh shape regularity, the number of fine patches $\omega_{\mathbf{r}}$ of diameters $\mathcal{O}(2^{-m})$ on the faces of a coarse patch $\omega_n^{(\ell)}$ of minimal diameter $\mathcal{O}(2^{-\mu})$ is of order $\mathrm{card}(\mathcal{F}_s^{(m)}) = \mathcal{O}(2^{-\mu(d-1)}/2^{-m(d-1)})$ in dimension $d = 2, 3$. Hence,

$$I_n \precsim \frac{2^\mu}{2^m} a(u_n,u_n) \quad \forall n \in X_k^{(\ell)}. \tag{4.120}$$

By taking the sum over $X_k^{(\ell)}$ whose patches are mutually disjoint,

$$J = a(Q_{\mathcal{C}_s^{(m)}}u, u) \precsim \sum_{n\in X_k^{(\ell)}} \frac{2^\mu}{2^m} a(u_n,u_n) = \frac{2^\mu}{2^m} a(u,u) \qquad \forall u \in \mathcal{V}(X_k^{(\ell)}). \tag{4.121}$$

Part 2. Now suppose $u \in \mathcal{Z}^{(\mu)} \subset \mathcal{V}^{(\ell)}$ such that $u = \sum_{k=1}^B u_k$ where $u_k \in \mathcal{V}(X_k^{(\ell)})$. Since B is finite, deduce from (4.121)

$$\begin{aligned} a(Q_{\mathcal{C}_s^{(m)}}u, u) &= a(Q_{\mathcal{C}_s^{(m)}}u, Q_{\mathcal{C}_s^{(m)}}u) = \Big| Q_{\mathcal{C}_s^{(m)}} \Big(\sum_{k=1}^B u_k \Big) \Big|^2_{\mathbf{H}^1(\mathbf{\Omega})} && (4.122)\\ &\precsim \sum_{k=1}^B |Q_{\mathcal{C}_s^{(m)}}(u_k)|^2_{\mathbf{H}^1(\mathbf{\Omega})} = \sum_{k=1}^B a(Q_{\mathcal{C}_s^{(m)}}u_k, u_k) && (4.123)\\ &\precsim \frac{2^\mu}{2^m} \sum_{k=1}^B |u_k|^2_{\mathbf{H}^1(\mathbf{\Omega})} \leq \frac{2^\mu}{2^m} |u|^2_{\mathbf{H}^1(\mathbf{\Omega})} && (4.124)\end{aligned}$$

because the reunion of the patches of $X_k^{(\ell)}$ covers the entire $\mathbf{\Omega}$.

■

4.3 Multilevel higher order estimate

In the next description, we want to analyze the upper bound from (2.8).

THEOREM. *For a polynomial degree $p \geq 1$ and a function $u_{hp} \in S_h^p(\mathbf{\Omega})$, the local higher order operator $P_p^{\mathbf{LHO}}$ and the hierarchical piecewise linear operator $P_p^{\mathbf{HPL}}$ satisfy*

$$a((P_p^{\mathbf{LHO}} + P^{\mathbf{HPL}})u_{hp}, u_{hp}) \leq C_1\Big[a(P^{\mathbf{HPL}}[P^{(L)}u_{hp}], P^{(L)}u_{hp}) + a(u_{hp}, u_{hp})\Big] \tag{4.125}$$

$$a(P^{\mathbf{HPL}}[P^{(L)}u_{hp}], P^{(L)}u_{hp}) \leq C_2 a(P^{(L)}u_{hp}, P^{(L)}u_{hp}) \tag{4.126}$$

and hence

$$a((P_p^{\mathbf{LHO}} + P^{\mathbf{HPL}})u_{hp}, u_{hp}) \leq C_3 a(u_{hp}, u_{hp}) \tag{4.127}$$

where C_1, C_2, C_3 are constants independent of the mesh sizes $h^{(\ell)}$, the polynomial degree p and the maximal level L.

PROOF. On the one hand, we have

$$a\big((P_p^{\mathbf{LHO}} + P^{\mathbf{HPL}})u_{hp}, u_{hp}\big) = a(P_p^{\mathbf{LHO}}u_{hp}, u_{hp}) + a\big(\sum_{\ell=0}^{L}\sum_{n\in\Lambda^{(\ell)}} P_n^{(\ell)}u_{hp}, u_{hp}\big). \tag{4.128}$$

$$I := a\big(\sum_{\ell=0}^{L}\sum_{n\in\Lambda^{(\ell)}} P_n^{(\ell)}u_{hp}, u_{hp}\big) \tag{4.129}$$

$$= \sum_{\ell=0}^{L}\sum_{n\in\Lambda^{(\ell)}} a(P_n^{(\ell)}u_{hp}, u_{hp}) \tag{4.130}$$

$$= \sum_{\ell=0}^{L}\sum_{n\in\Lambda^{(\ell)}} a\big(P_n^{(\ell)}u_{hp}, P^{(L)}u_{hp}\big) \qquad (\text{because } P_n^{(\ell)}u_{hp} \in \mathcal{V}^{(L)}) \tag{4.131}$$

$$= \sum_{\ell=0}^{L}\sum_{n\in\Lambda^{(\ell)}} a\big(P_n^{(\ell)}u_{hp}, P_n^{(\ell)}[P^{(L)}u_{hp}]\big) \qquad (\text{because } P_n^{(\ell)}u_{hp} \in \mathcal{V}_n^{(\ell)}) \tag{4.132}$$

$$= \sum_{\ell=0}^{L}\sum_{n\in\Lambda^{(\ell)}} a(u_{hp}, P_n^{(\ell)}[P^{(L)}u_{hp}]) \qquad (\text{because } P_n^{(\ell)}[P^{(L)}u_{hp}] \in \mathcal{V}_n^{(\ell)}) \tag{4.133}$$

$$= \sum_{\ell=0}^{L}\sum_{n\in\Lambda^{(\ell)}} a(P^{(L)}u_{hp}, P_n^{(\ell)}[P^{(L)}u_{hp}]) \qquad (\text{because } P_n^{(\ell)}[P^{(L)}u_{hp}] \in \mathcal{V}^{(L)}) \tag{4.134}$$

$$= a(P^{\mathbf{HPL}}[P^{(L)}u_{hp}], P^{(L)}u_{hp}). \tag{4.135}$$

On the other hand, from the fact that $\mathrm{supp}(P_{(p,n)}u_{hp}) = \omega_n^{(L)}$, obtain

$$a(P_p^{\mathbf{LHO}}u_{hp}, u_{hp}) = \sum_{n\in\mathcal{J}^{(L)}} a_{\omega_n^{(L)}}(P_{(p,n)}u_{hp}, u_{hp}), \tag{4.136}$$

$$a_{\omega_n^{(L)}}(P_{(p,n)}u_{hp}, u_{hp}) = a_{\omega_n^{(L)}}(P_{(p,n)}u_{hp}, P_{(p,n)}u_{hp}). \tag{4.137}$$

As a consequence,

$$\begin{aligned} J &:= a_{\omega_n^{(L)}}(u_{hp}, u_{hp}) - a_{\omega_n^{(L)}}(P_{(p,n)}u_{hp}, u_{hp}) && (4.138)\\ &= a_{\omega_n^{(L)}}(P_{(p,n)}u_{hp}, P_{(p,n)}u_{hp}) + a_{\omega_n^{(L)}}(u_{hp}, u_{hp}) - 2a_{\omega_n^{(L)}}(P_{(p,n)}u_{hp}, u_{hp}) && (4.139)\\ &= a_{\omega_n^{(L)}}([P_{(p,n)}u_{hp}] - u_{hp}, [P_{(p,n)}u_{hp}] - u_{hp}) && (4.140)\\ &= |P_{(p,n)}u_{hp} - u_{hp}|^2_{\mathbf{H}^1(\omega_n^{(L)})} \geq 0. && (4.141) \end{aligned}$$

Hence,

$$a_{\omega_n^{(L)}}(P_{(p,n)}u_{hp}, u_{hp}) \leq a_{\omega_n^{(L)}}(u_{hp}, u_{hp}). \tag{4.142}$$

By combining that with (4.136), obtain

$$a(P_p^{\mathbf{LHO}}u_{hp}, u_{hp}) \leq \sum_{n\in\mathcal{J}^{(L)}} a_{\omega_n^{(L)}}(u_{hp}, u_{hp}) = \sum_{n\in\mathcal{J}^{(L)}} |u_{hp}|^2_{\omega_n^{(L)}}. \tag{4.143}$$

Since the patches intersect at most a finite number of times due to mesh shape regularities and the reunion of $\omega_n^{(L)}$ for $n \in \mathcal{J}^{(L)}$ covers the whole domain $\mathbf{\Omega}$, we have

$$a(P_p^{\mathbf{LHO}}u_{hp}, u_{hp}) \leq |u_{hp}|^2_{\mathbf{H}^1(\mathbf{\Omega})}. \tag{4.144}$$

A combination of (4.128), (4.135) and (4.144) yields (4.125). Define $u_h := P^{(L)}u_{hp} \in \mathcal{Z}^{(M)} \equiv \mathcal{V}^{(L)}$ and

$$\begin{aligned} u^\mu &:= [Q^{(\mu)} - Q^{(\mu-1)}]u_h \in \mathcal{Z}^{(\mu)} \quad \text{if} \quad \mu \geq 1 && (4.145)\\ u^0 &:= Q^{(0)}u_h \in \mathcal{Z}^{(0)}. && (4.146) \end{aligned}$$

Thus, one has $Q^{(m)}u_h = \sum_{\mu=0}^m u^\mu$. Use (4.108) and the fact that $\mathrm{supp}(Q^{(m)}_{\mathbf{r}_1}u_h) \cap \mathrm{supp}(Q^{(m)}_{\mathbf{r}_2}u_h) = \emptyset$ for $\mathbf{r}_1, \mathbf{r}_2 \in \mathcal{C}_s^{(m)}$ and $\mathbf{r}_1 \neq \mathbf{r}_2$ to obtain:

$$\begin{aligned} K &:= a(Q_{\mathcal{C}_s^{(m)}}u_h, u_h) = a(Q^{(m)}u_h, Q_{\mathcal{C}_s^{(m)}}u_h) && (4.147)\\ &= a(Q_{\mathcal{C}_s^{(m)}}Q^{(m)}u_h, Q^{(m)}u_h) = a(Q_{\mathcal{C}_s^{(m)}}Q^{(m)}u_h, Q_{\mathcal{C}_s^{(m)}}Q^{(m)}u_h) && (4.148)\\ &= |Q_{\mathcal{C}_s^{(m)}}Q^{(m)}u_h|^2_{\mathbf{H}^1(\mathbf{\Omega})} \precsim \Big(\sum_{\mu=0}^m |Q_{\mathcal{C}_s^{(m)}}u^\mu|_{\mathbf{H}^1(\mathbf{\Omega})}\Big)^2 && (4.149)\\ &= \Big(\sum_{\mu=0}^m a(Q_{\mathcal{C}_s^{(m)}}u^\mu, u^\mu)^{1/2}\Big)^2. && (4.150) \end{aligned}$$

Since $u^\mu \in \mathcal{Z}^{(\mu)}$, use the strengthened Cauchy inequality (4.109) to obtain

$$K \precsim \Big[\sum_{\mu=0}^m \sqrt{\frac{2^\mu}{2^m}}a(u^\mu, u^\mu)^{1/2}\Big]^2 \leq \Big[\sum_{\mu=0}^m \sqrt{\frac{2^\mu}{2^m}}\Big]\Big[\sum_{\mu=0}^m \sqrt{\frac{2^\mu}{2^m}}a(u^\mu, u^\mu)\Big], \tag{4.151}$$

$$\sum_{\mu=0}^{m}\sqrt{\frac{2^\mu}{2^m}}=\Big(\frac{1}{2^m}\Big)^{1/2}\Big(\frac{1-(\sqrt{2})^{m+1}}{1-\sqrt{2}}\Big)=\frac{1}{2^{m/2}}\mathcal{O}(2^{(m+1)/2}-1)=\mathcal{O}(1). \tag{4.152}$$

As a result,

$$a(Q_{\mathcal{C}_s^{(m)}}u_h,u_h)\precsim\sum_{\mu=0}^{m}\sqrt{\frac{2^\mu}{2^m}}a(u^\mu,u^\mu). \tag{4.153}$$

Take the sum over s and use

$$\sum_{\mathbf{r}\in\mathcal{R}^{(m)}}Q_{\mathbf{r}}^{(m)}=\sum_{s=1}^{B}Q_{\mathcal{C}_s^{(m)}} \tag{4.154}$$

to obtain

$$\sum_{\mathbf{r}\in\mathcal{R}^{(m)}}a(Q_{\mathbf{r}}^{(m)}u_h,u_h)\precsim\sum_{s=1}^{B}\sum_{\mu=1}^{m}\sqrt{\frac{2^\mu}{2^m}}a(u^\mu,u^\mu)=B\sum_{\mu=1}^{m}\sqrt{\frac{2^\mu}{2^m}}a(u^\mu,u^\mu). \tag{4.155}$$

From (4.104), deduce

$$a(P^{\mathbf{HPL}}u_h,u_h)=\sum_{m=0}^{M}\sum_{\mathbf{r}\in\mathcal{R}^{(m)}}a(Q_{\mathbf{r}}^{(m)}u_h,u_h)\precsim\sum_{m=0}^{M}\sum_{\mu=1}^{m}\sqrt{\frac{2^\mu}{2^m}}a(u^\mu,u^\mu). \tag{4.156}$$

The remaining part is processed as above by noting from the orthogonality that $a(u_h,u_h)=\sum_{\mu=0}^{M}a(u^\mu,u^\mu)$ to obtain $a(P^{\mathbf{HPL}}u_h,u_h)\precsim a(u_h,u_h)$. That is,

$$a(\sum_{\ell=0}^{L}\sum_{n\in\Lambda^{(\ell)}}P_n^{(\ell)}\big[P^{(L)}u_{hp}\big],P^{(L)}u_{hp})\precsim a(P^{(L)}u_{hp},P^{(L)}u_{hp}) \tag{4.157}$$

which provides (4.126). Proceed as in (4.137)– (4.142) to obtain $a(P^{(L)}u_{hp},u_{hp})\leq a(u_{hp},u_{hp})$. By combining that with (4.144) and (4.157), deduce

$$a((P_p^{\mathbf{LHO}}+P^{\mathbf{HPL}})u_{hp},u_{hp})\precsim a(u_{hp},u_{hp}). \tag{4.158}$$

■

REMARK. *The assumptions (4.88)–(4.90) are natural for quasi-uniform meshes. Indeed, they are based on graph theory in the context of chromatic problems (see [16] and the references there). One defines a graph $\mathcal{G}$ where one associates a graph vertex to a patch. Two graph vertices are connected by a graph edge if the closure of their corresponding patches intersect. The degree of the graph vertex v is the number of graph edges emanating from v. The maximum vertex degree Δ is bounded due to mesh regularity. Hence, the chromatic decomposition is derived from graph theoretic arguments. The above number B is any number larger than the chromatic number $\chi(\mathcal{G})$ which is either Δ or $(\Delta+1)$ due to Brook's theorem.*

■

5 Higher order FEM for elliptic systems

In this section, we want to carry the previous method over to elliptic systems. Throughout, vector values are supposed to be column vectors. Thus, the scalar product of $\mathbf{a}$ and $\mathbf{b}$ is denoted by $\mathbf{a}^{\mathrm{T}}\mathbf{b}$ while the norm is denoted by $|\mathbf{a}| = \sqrt{\mathbf{a}^{\mathrm{T}}\mathbf{a}}$. The vector/matrix valued Sobolev spaces are component-wise of the scalar valued Sobolev spaces. A partial differential operator acting on $\mathbf{u} : \mathbf{\Omega} \subset \mathbb{R}^d \longrightarrow \mathbb{R}^m$ which is a vector valued function is

$$\mathcal{L}\mathbf{u} = -\sum_{i=1}^{d}\sum_{j=1}^{d}\partial_i\big(\mathbf{A}_{ij}(\mathbf{x})\partial_j\mathbf{u}\big) + \sum_{i=1}^{d}\mathbf{B}_i(\mathbf{x})\partial_i\mathbf{u} + \mathbf{C}(\mathbf{x})\mathbf{u} \tag{5.159}$$

such that $\mathcal{L}\mathbf{u} : \mathbf{\Omega} \longrightarrow \mathbb{R}^m$. The coefficients are matrix valued functions $\mathbf{A}_{ij}(\mathbf{x}) = [a_{ij}^{kl}(\mathbf{x})]_{k,l=1}^{m}$, $\mathbf{B}_i(\mathbf{x}) = [b_i^{kl}(\mathbf{x})]_{k,l=1}^{m}$, $\mathbf{C}(\mathbf{x}) = [c^{kl}(\mathbf{x})]_{k,l=1}^{m}$ such that $a_{ij}^{kl} : \mathbf{\Omega} \longrightarrow \mathbb{R}^m$, $b_i^{kl} : \mathbf{\Omega} \longrightarrow \mathbb{R}^m$, $c^{kl} : \mathbf{\Omega} \longrightarrow \mathbb{R}^m$. The associated bilinear form is

$$\mathcal{A}(\mathbf{u},\mathbf{v}) = \int_{\mathbf{\Omega}}\Big(\sum_{i=1}^{d}\sum_{j=1}^{d}\big(\mathbf{A}_{ij}(\mathbf{x})\partial_j\mathbf{u}\big)^{\mathrm{T}}\partial_i\mathbf{v} + \sum_{i=1}^{d}(\mathbf{B}_i(\mathbf{x})\partial_i\mathbf{u})^{\mathrm{T}}\mathbf{v} + (\mathbf{C}(\mathbf{x})\mathbf{u})^{\mathrm{T}}\mathbf{v}\Big)d\mathbf{x}. \tag{5.160}$$

The coefficients of $\mathcal{L}$ are assumed to have adequate properties such that $\mathcal{A}$ is continuous and coercive with respect to $[\mathbf{H}^1(\mathbf{\Omega})]^m$. They are related by

$$(\mathcal{L}\mathbf{u},\mathbf{v})_{\mathbf{\Omega}} = \mathcal{A}(\mathbf{u},\mathbf{v}) - (\mathcal{B}_{\mathbf{n}}\mathbf{u},\mathrm{Trace}(\mathbf{v}))_{\mathbf{\Gamma}} \tag{5.161}$$

where $\mathcal{B}_{\mathbf{n}}\mathbf{u}$ is the conormal derivative

$$\mathcal{B}_{\mathbf{n}}\mathbf{u} = \sum_{i=1}^{d} n_i \mathrm{Trace}\big[\sum_{j=1}^{d}\mathbf{A}_{ij}\partial_j\mathbf{u}\big] \quad \text{on} \quad \mathbf{\Gamma} = \partial\mathbf{\Omega} \tag{5.162}$$

with $\mathbf{n} = (n_1, ..., n_d)$ being the outward unit normal vector. We consider only the Poisson and the elasticity cases. In the former case, one has $d = 2, 3$ and $m = 1$ while the coefficients in the principal part of (5.159) are $\mathbf{A}_{ij} = \delta_{i,j}$ and the other coefficients vanish. In the case of elasticity, the domain $\mathbf{\Omega}$ is an elastic model which admits the elastic properties that are specified by the Lamé coefficients (λ, μ) whose relation with the Young's modulus E and the Poisson ratio ν is:

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \qquad \mu = \frac{E}{2(1+\nu)}. \tag{5.163}$$

When the model $\mathbf{\Omega}$ is subject to a body force and prescribed boundary conditions, it has the displacement function $\mathbf{u} : \mathbf{\Omega} \longrightarrow \mathbb{R}^d$ whose strain tensor is defined as

$$\varepsilon_{ij}(\mathbf{u}(\mathbf{x})) := \frac{1}{2}\big(\partial_j u_i(\mathbf{x}) + \partial_i u_j(\mathbf{x})\big) \quad \text{for} \quad \mathbf{x} \in \mathbf{\Omega}. \tag{5.164}$$

According to the Hook's law for the stress-strain relation, the stress tensor is given by

$$\begin{aligned} \sigma_{ij}(\mathbf{u}) &= \mu(\partial_j u_i + \partial_i u_j) + \lambda \sum_k (\partial_k u_k)\delta_{i,j} && (5.165)\\ &= 2\mu\varepsilon_{ij}(\mathbf{u}) + \lambda(\operatorname{div}\mathbf{u})\delta_{i,j}. && (5.166)\end{aligned}$$

In term of tensors where $\nabla\mathbf{u}$ denotes the matrix having $\partial_i u_j$ as coefficients, we have

$$\boldsymbol{\varepsilon}(\mathbf{u}) = \big[\varepsilon_{ij}(\mathbf{u})\big]_{i,j} = 0.5\big[(\nabla\mathbf{u}) + (\nabla\mathbf{u})^{\mathrm{T}}\big] \qquad \text{and} \qquad \boldsymbol{\sigma}(\mathbf{u}) = \big[\sigma_{ij}(\mathbf{u})\big]_{i,j}, \tag{5.167}$$

We consider the interior Navier-Lamé equation with Dirichlet boundary condition:

$$\begin{cases} -\operatorname{div}\big[\boldsymbol{\sigma}\big(\mathbf{u}(\mathbf{x})\big)\big] &= \mathbf{f}(\mathbf{x}) & \text{for} \quad \mathbf{x}\in\boldsymbol{\Omega}\\ \boldsymbol{\sigma}\big(\mathbf{u}(\mathbf{x})\big) &= 2\mu\boldsymbol{\varepsilon}\big(\mathbf{u}(\mathbf{x})\big) + \lambda\,\mathrm{Trace}\big[\boldsymbol{\varepsilon}\big(\mathbf{u}(\mathbf{x})\big)\big]\mathbf{I}_d & \text{for} \quad \mathbf{x}\in\boldsymbol{\Omega}\\ \mathbf{u}(\mathbf{x}) &= \mathbf{0} & \text{for} \quad \mathbf{x}\in\boldsymbol{\Gamma}=\partial\boldsymbol{\Omega}\end{cases} \tag{5.168}$$

in which $\mathbf{f}$ represents a given body/volume force function. The above Navier-Lamé equation corresponds to the partial differential operator $\mathcal{L}$ coinciding with its principal part $\big(\mathbf{B}_i(\mathbf{x}) = \mathbf{C}(\mathbf{x}) = \mathbf{0}\big)$ such that $m \equiv d$ and the coefficients in 2D and 3D are respectively

$$\begin{cases} \mathbf{A}_{11} = \begin{bmatrix} \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} & 0 \\ 0 & \frac{E}{2(1+\nu)} \end{bmatrix}, \mathbf{A}_{12} = \begin{bmatrix} 0 & \frac{E}{4(1+\nu)(1-2\nu)} \\ \frac{E}{4(1+\nu)(1-2\nu)} & 0 \end{bmatrix},\\ \mathbf{A}_{21} = \begin{bmatrix} 0 & \frac{E}{4(1+\nu)(1-2\nu)} \\ \frac{E}{4(1+\nu)(1-2\nu)} & 0 \end{bmatrix}, \mathbf{A}_{22} = \begin{bmatrix} \frac{E}{2(1+\nu)} & 0 \\ 0 & \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \end{bmatrix}, \end{cases} \tag{5.169}$$

$$\begin{cases} \mathbf{A}_{11} = \begin{bmatrix} \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} & 0 & 0 \\ 0 & \frac{E}{2(1+\nu)} & 0 \\ 0 & 0 & \frac{E}{2(1+\nu)} \end{bmatrix}, \mathbf{A}_{12} = \begin{bmatrix} 0 & \frac{E\nu}{(1+\nu)(1-2\nu)} & 0 \\ \frac{E}{2(1+\nu)} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},\\ \mathbf{A}_{13} = \begin{bmatrix} 0 & 0 & \frac{E\nu}{(1+\nu)(1-2\nu)} \\ 0 & 0 & 0 \\ \frac{E}{2(1+\nu)} & 0 & 0 \end{bmatrix}, \mathbf{A}_{21} = \begin{bmatrix} 0 & \frac{E}{2(1+\nu)} & 0 \\ \frac{E\nu}{(1+\nu)(1-2\nu)} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},\\ \mathbf{A}_{22} = \begin{bmatrix} \frac{E}{2(1+\nu)} & 0 & 0 \\ 0 & \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} & 0 \\ 0 & 0 & \frac{E}{2(1+\nu)} \end{bmatrix}, \mathbf{A}_{23} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{E\nu}{(1+\nu)(1-2\nu)} \\ 0 & \frac{E}{2(1+\nu)} & 0 \end{bmatrix},\\ \mathbf{A}_{31} = \begin{bmatrix} 0 & 0 & \frac{E}{2(1+\nu)} \\ 0 & 0 & 0 \\ \frac{E\nu}{(1+\nu)(1-2\nu)} & 0 & 0 \end{bmatrix}, \mathbf{A}_{32} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{E}{2(1+\nu)} \\ 0 & \frac{E\nu}{(1+\nu)(1-2\nu)} & 0 \end{bmatrix},\\ \mathbf{A}_{33} = \begin{bmatrix} \frac{E}{2(1+\nu)} & 0 & 0 \\ 0 & \frac{E}{2(1+\nu)} & 0 \\ 0 & 0 & \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \end{bmatrix}. \end{cases} \tag{5.170}$$

The differential operator $\mathcal{L}$ is associated to the bilinear form

$$\mathcal{A}(\mathbf{u},\mathbf{v}) = \int_{\mathbf{\Omega}} \sum_{i,j} 2\mu\varepsilon_{ij}\big(\mathbf{u}(\mathbf{x})\big)\varepsilon_{ij}\big(\mathbf{v}(\mathbf{x})\big) + \lambda \mathrm{div}\big(\mathbf{u}(\mathbf{x})\big)\mathrm{div}\big(\mathbf{v}(\mathbf{x})\big) d\mathbf{x} \quad (5.171)$$

$$= \int_{\mathbf{\Omega}} \boldsymbol{\varepsilon}(\mathbf{u}(\mathbf{x})) : \boldsymbol{\sigma}(\mathbf{v}(\mathbf{x}))\, d\mathbf{x} \quad (5.172)$$

where $A : B := \sum_{i,j} A_{ij}B_{ij}$. The resulting energy norm $\| \bullet \|_E := \mathcal{A}(\bullet,\bullet)^{1/2}$ is

$$\|\mathbf{u}\|_E^2 = 2\mu \big\|\boldsymbol{\varepsilon}(\mathbf{u})\big\|^2_{[\mathbf{L}_2(\mathbf{\Omega})]^{d\times d}} + \lambda \big\|\mathrm{div}(\mathbf{u})\big\|^2_{\mathbf{L}_2(\mathbf{\Omega})}. \quad (5.173)$$

The conormal derivative (5.162) of a function $\mathbf{u}$ is given component-wise by

$$[\mathcal{B}_{\mathbf{n}}\mathbf{u}]_i(\mathbf{x}) = \sum_{j=1}^{d} n_j(\mathbf{x})\sigma_{ij}[\mathbf{u}(\mathbf{x})] \quad (5.174)$$

which coincides with the traction in engineering applications. The Galerkin variational formulation is

$$\int_{\mathbf{\Omega}} \boldsymbol{\varepsilon}[\mathbf{u}(\mathbf{x})] : \boldsymbol{\sigma}[\mathbf{v}(\mathbf{x})] d\mathbf{x} = \int_{\mathbf{\Omega}} [\mathbf{f}(\mathbf{x})]^{\mathrm{T}}\mathbf{v}(\mathbf{x}) d\mathbf{x} \qquad \forall\, \mathbf{v} \in \big[\mathbf{H}^1_0(\mathbf{\Omega})\big]^d. \quad (5.175)$$

The approximated variational formulation consists in searching for $\mathbf{u}_h \in \mathcal{S}^p_h$ such that

$$\int_{\mathbf{\Omega}} \boldsymbol{\varepsilon}[\mathbf{u}_h(\mathbf{x})] : \boldsymbol{\sigma}[\mathbf{v}_h(\mathbf{x})] d\mathbf{x} = \int_{\mathbf{\Omega}} [\mathbf{f}(\mathbf{x})]^{\mathrm{T}}\mathbf{v}_h(\mathbf{x}) d\mathbf{x} \qquad \forall\, \mathbf{v}_h \in \mathcal{S}^p_h(\mathbf{\Omega}) \quad (5.176)$$

where

$$\mathcal{S}^p_h(\mathbf{\Omega}) := \left\{ \mathbf{W} \in \Big[\mathcal{C}^0(\mathbf{\Omega}) \cap \mathbf{H}^1_0(\mathbf{\Omega})\Big]^d \;\middle|\; \begin{array}{l} \mathbf{W} = (W_1, ..., W_d), \\ W_i \in \mathcal{P}_p(T) \quad \forall\, T \in \mathcal{M},\, \forall\, i = 1, ..., d \end{array} \right\}.$$

Under some adequate assumptions on the Lamé coefficients, there exists some constant $C > 0$ such that

$$C \sum_{j,l=1}^{d} (\eta^l_j)^2 \leq \sum_{i,j,k,l=1}^{d} a^{kl}_{ij}\eta^l_j\eta^k_i. \quad (5.177)$$

In addition, the continuity of $\mathcal{A}(\bullet,\bullet)$ with respect to $\big[\mathbf{H}^1(\mathbf{\Omega})\big]^d$ is obvious because the coefficients $\mathbf{A}_{i,j} \in \big[\mathbf{L}^\infty(\mathbf{\Omega})\big]^{d\times d}$. Thus, the description from the former sections can be repeated for elliptic systems where $\mathcal{A}(\bullet,\bullet)$ plays the role of $a(\bullet,\bullet)$ and the Sobolev norm of $\big[\mathbf{H}^1(\mathbf{\Omega})\big]^d$ is considered component-wise. We briefly expose now some a-posteriori error estimates because we will need it to conduct adaptivity in the next description. We define the following mutually disjoint subsets of faces (edges for $d = 2$ and triangular faces for $d = 3$)

$$\begin{aligned} \mathbb{E}^{\mathbf{0}}_h &:= \text{ set of faces of } \mathcal{M} \text{ on the boundary } \mathbf{\Gamma} = \partial\mathbf{\Omega}, \\ \mathbb{E}^{\mathbf{int}}_h &:= \text{ set of faces of } \mathcal{M} \text{ which are not included in } \mathbf{\Gamma}. \end{aligned}$$

Note that a face of $\mathbb{E}_h^{\mathbf{int}}$ may have an endpoint in $\mathbf{\Gamma}$. We introduce in addition the set of all faces

$$\mathbb{E}_h := \mathbb{E}_h^{\mathbf{0}} \cup \mathbb{E}_h^{\mathbf{int}}. \tag{5.178}$$

For a face $e \in \mathbb{E}_h$, we denote

$$\begin{aligned} h(e) &:= \text{diameter}(e) = \sup\Big\{|\mathbf{x}-\mathbf{y}|,\, \mathbf{x},\mathbf{y} \in e\Big\}, \\ \mathcal{N}(e) &:= \text{set of elements of } \mathcal{M} \text{ having } e \text{ as a side}, \\ \mathbf{n}(e) &:= \text{unit normal vector orthogonal to } e. \end{aligned}$$

The direction of the normal vector $\mathbf{n}(e)$ is pointed toward the exterior of $\mathbf{\Omega}$ if the face $e \in \mathbb{E}_h^{\mathbf{0}}$. For the interior faces in $\mathbb{E}_h^{\mathbf{int}}$, the normal vectors $\mathbf{n}(e)$ are pointed in an arbitrary but fixed orientation. For an element $T = \{(x_0, y_0), ..., (x_d, y_d)\}$, we denote the barycentric coordinates by $\lambda_{i,T}$ for $i = 0, ..., d$. The barycentric weight function is expressed as

$$\omega_T(\mathbf{x}) = \lambda_{0,T}(\mathbf{x}) \cdots \lambda_{d,T}(\mathbf{x}) \qquad \text{for} \quad \mathbf{x} \in T \tag{5.179}$$

which takes zero values at the boundary of the element T. It is a generalization of the uni-dimensional Gegebauer weight (also known as Jacobi weight) which is defined on the unit interval $[0, 1]$ as

$$\omega_{[0,1]}(t) := t(1-t) \qquad \forall\, t \in [0,1]. \tag{5.180}$$

In [31, 32], the following weight function has been used

$$\omega_T(\mathbf{x}) = \text{distance}(\mathbf{x}, \partial T) \qquad \forall\, \mathbf{x} \in T \tag{5.181}$$

which is a bubble function vanishing at the boundary ∂T. The advantage of the Gegenbauer weight (5.179) is that it is not expensive to evaluate in practical computations. We have described in [34] the following result related to inverse estimates in weighted Sobolev spaces.

THEOREM. *Given α, β such that $-1 < \alpha < \beta$ and some $\delta \in [0, 1]$. By using the generalized Gegenbauer weight*

$$\omega_{\widehat{T}}(\mathbf{x}) = \lambda_{0,\widehat{T}}(\mathbf{x}) \cdots \lambda_{d,\widehat{T}}(\mathbf{x}) \tag{5.182}$$

for $\mathbf{x} = (x_1, ..., x_d)$ in the reference element

$$\widehat{T} = \Big\{\mathbf{x} = (x_1, ..., x_d) \in \mathbb{R}^d : 0 \le x_i \le 1,\ i = 1, ..., d,\ 0 \le x_1 + \cdots + x_d \le 1\Big\}, \tag{5.183}$$

one has for every bivariate polynomial π_p *of degree* $p \geq 1$

$$\int_{\widehat{T}} \big[\pi_p(\mathbf{x})\big]^2 \omega_{\widehat{T}}^{\alpha}(\mathbf{x}) d\mathbf{x} \leq C_1 p^{2(\beta-\alpha)} \int_{\widehat{T}} \big[\pi_p(\mathbf{x})\big]^2 \omega_{\widehat{T}}^{\beta}(\mathbf{x}) d\mathbf{x} \tag{5.184}$$

$$\int_{\widehat{T}} \big|\nabla \pi_p(\mathbf{x})\big|^2 \omega_{\widehat{T}}^{2\delta}(\mathbf{x}) d\mathbf{x} \leq C_2 p^{2(2-\delta)} \int_{\widehat{T}} \big[\pi_p(\mathbf{x})\big]^2 \omega_{\widehat{T}}^{\delta}(\mathbf{x}) d\mathbf{x} \tag{5.185}$$

$$\int_{\widehat{T}} \big|\nabla \pi_p(\mathbf{x})\big|^2 \omega_{\widehat{T}}(\mathbf{x}) d\mathbf{x} \leq C_3 p^2 \int_{\widehat{T}} \big[\pi_p(\mathbf{x})\big]^2 \omega_{\widehat{T}}(\mathbf{x}) d\mathbf{x}. \tag{5.186}$$

The constants are $C_1 = C_1(\alpha, \beta)$, $C_2 = C_2(\delta)$, C_3 *which do not depend on* p.

For the case of Poisson, we apply the APEE from [31, 32] with the exception that we use the barycentric weights (5.179) instead of (5.181). As for the elasticity, we suppose that we dispose of the approximated solution $\mathbf{u}_h$ and our purpose is to estimate the error $\|\mathbf{u} - \mathbf{u}_h\|_{[\mathbf{H}^1(\mathbf{\Omega})]^d}$. For an element $T \in \mathcal{M}$, the interior estimator is defined as

$$\eta_{\alpha,T}^{\mathbf{inter}} := \frac{h(T)}{p} \Big\| \big(\mathbf{f}_T + \mu \Delta \mathbf{u}_h + (\mu + \lambda) \mathbf{grad}(\operatorname{div} \mathbf{u}_h)\big) \omega_T^{\alpha/2} \Big\|_{\left[\mathbf{L}^2(T)\right]^d} \tag{5.187}$$

where $\mathbf{f}_T$ designates the $\mathbf{L}_2(T)$-projection of the body force $\mathbf{f}$ onto the element T. The estimator for an interior face $e \in \mathbb{E}_h^{\mathbf{int}}$ having a normal vector $\mathbf{n}(e)$ is defined by means of the stress tensor $\boldsymbol{\sigma}(\mathbf{u}_h)$ as

$$\eta_{\alpha,e}^{\mathbf{face}} := \sqrt{\frac{h(e)}{2p}} \Big\| \big(\boldsymbol{\sigma}(\mathbf{u}_h)_{|_{T_{(1,e)}}} \mathbf{n}(e) - \boldsymbol{\sigma}(\mathbf{u}_h)_{|_{T_{(2,e)}}} \mathbf{n}(e)\big) \omega_e^{\alpha/2} \Big\|_{\left[\mathbf{L}^2(e)\right]^d} \tag{5.188}$$

where $T_{(1,e)}$ and $T_{(2,e)}$ are the elements incident upon the face e. Since one needs computable local estimators for an element-by-element computation, an interior element $T \in \mathcal{M}$ is introduced

$$\eta_{\alpha,T}^{\mathbf{loc}} := \Big[(\eta_{\alpha,T}^{\mathbf{inter}})^2 + \sum_{e \subset \partial T,\, e \in \mathbb{E}_h^{\mathbf{int}}} (\eta_{\alpha,e}^{\mathbf{face}})^2 \Big]^{1/2} \tag{5.189}$$

The local estimators add up to the global estimator:

$$\eta_\alpha := \sqrt{\sum_{T \in \mathcal{M}} (\eta_{\alpha,T}^{\mathbf{loc}})^2}. \tag{5.190}$$

By using the above theorem the APEE η_α is reliable and efficient with respect to the exact error. The principal role of an a-posteriori error estimator is twofold. For one, it serves as gaining some idea of whether to continue or to abort a simulation. The computation is aborted when the desired precision is provided by the estimator. A further purpose of the error estimator is to identify the regions within the domain $\mathbf{\Omega}$ where the precision is unsatisfactory. Mesh refinements are therefore applied at those regions to improve the local precision.

6 Numerical experiments

We want now to present some results of the former method which was implemented in C functions together with C++ classes. RABOOL is our in-house software which treats h, p and hp FEM implementation. Its parallelization is based on message passing carried out with MPI. The current hardware for the execution of this program is very moderate in term of parallel computing but the program is designed to work on large number of CPUs. We concentrate for now on optimizing the programming task (load balancing, enhancing the local computing tasks, data compression and minimization of inter-process communication), rather than on increasing the CPU counts. Porting this program to a more powerful hardware is a matter of a little or no modification. The computations have been executed on a computing equipment possessing a total processing power of 4.1 GHz, a total memory 32 GB RAM and 8 CPUs. We have already presented a partial result in [37] on a hardware admitting a larger number of computing units (no more than 100 CPUs in general). Since then, the programming has considerably developed in performance and in efficiency.

6.1 Numerical convergence

Before presenting our results about the higher order linear solver, we want to display in this section some results about the convergence. Because a fast linear solver does not make sense if the FEM-convergence is not even obtained. The scope of this section is a little larger than the former description.

6.1.1 h-performance

Our first test consists in studying the h-performance in 2D and 3D where the polynomial degree is fixed and the mesh size h is variable. The precisions of the solutions to the Poisson and the Navier-Lamé equations are measured with respect to the $\mathbf{H}^1$-accuracies $|u-u_h|_{\mathbf{H}^1(\mathbf{\Omega})}$, $\left|\mathbf{u}-\mathbf{u}_h\right|_{[\mathbf{H}^1(\mathbf{\Omega})]^d}$. Concerning $d=2$, the exact solution is $u(\mathbf{x})=\sin(2\pi x_1)$ $\sin(2\pi x_2)$ for the Poisson equation while it is $\mathbf{u}(\mathbf{x})=\big[\sin(2\pi x_1)\ \sin(2\pi x_2)$,$\sin(2\pi x_1)$ $\sin(2\pi x_2)\big]^{\mathrm{T}}$ for the elasticity on the domain $\mathbf{\Omega}$ which the unit square. Similar exact solutions are used in the 3D case inside the unit cube. The expressions of the right hand side are computed according to the Laplace operator and the Navier-Lamé operator (5.168). One level incrementation amounts to reducing the mesh size from h to $h/2$ and we consider the polynomial degrees $p=1,2,3,4$. For each polynomial degree, the FEM-level ranges from one to six in the 2D case, while it is from one to five in the

3D case. For the elasticity simulation, the Lamé coefficients are $(\lambda, \mu) = (2, 1)$ in both 2 and 3 dimensions. The ratio ρ_p between the accuracy on level ℓ and the accuracy on the preceding level $\ell - 1$ for the polynomial degree p will be termed *contraction ratio*. The accuracy results as well as the contraction ratios are collected in Table 1. For the linear case where the polynomial degree $p = 1$, one has the contraction ratios of $\rho_1 \approx 2$. Since the elastic parameter λ is small, the contraction ratio for $p = 1$ coincides with the expectation. When the elastic material approaches incompressibility, i.e. the value of the Poisson ratio ν is very close to $1/2$ (or equivalently λ is very large), the piecewise linear setting is not theoretically guaranteed to converge. That inconvenience is better known as *locking phenomenon* [8, 45]. Known methods for avoiding the locking phenomenon include: (1)increase of the polynomial degree as done here where the displacement formulation is used, (2)using mixed formulation. The contraction ratios for the quadratic case are $\rho_2 \approx 4$, while those for the cases $p = 3, 4$ are $\rho_3 \approx 8$, $\rho_4 \approx 16$. Those contraction ratios are not well perceived when the level is low. But as the levels grow, they approach those ideal values for both equations and for both dimensions.

6.1.2 p-performance

Our next test consists in investigating the software for the case of increasing polynomial degree which ranges from 1 to 13. That means, the whole mesh is kept intact from beginning till the end of the simulation in both 2D and 3D. The corresponding results are collected in Figure 5 where we consider the $\mathbf{H}^1$-accuracy in function of the polynomial degrees. In the case of elasticity, the Lamé coefficients are again $(\lambda, \mu) = (2, 1)$. Since the vertical axis is logarithmically scaled, this experiment highlights that all the accuracies decrease exponentially proportional with the polynomial degree. For both the Poisson and the elasticity equations, the slope of the curves are linear with respect to the polynomial degree. The slope should somehow depend on the material properties. But this should be robust against the locking phenomenon even when the elastic material approaches the incompressiblity.

6.1.3 hp-performance

The next tests related to hp-simulation need a reliable and efficient a-posteriori error estimator. In our former investigation [34, 35], we conducted a comparison between the $\mathbf{H}^1$-error and the error estimator η_α. We observed that the proposed estimators capture the exact precision up to some constant factors. In fact, the decrease of the exact and the estimated precisions follow the same pace as the FEM-level increases.

	edge	$p=1$		$p=2$		$p=3$		$p=4$	
		$\mathbf{H}^1$	ratio	$\mathbf{H}^1$	ratio	$\mathbf{H}^1$	ratio	$\mathbf{H}^1$	ratio
P2	h	1.499157E+00	—	2.058470E-01	—	1.847078E-02	—	1.404930E-03	—
	$h/2$	7.687939E-01	1.950012	5.287170E-02	3.893330	2.311419E-03	7.991100	8.908065E-05	15.771439
	$h/4$	3.868795E-01	1.987166	1.331219E-02	3.971676	2.884402E-04	8.013512	5.586588E-06	15.945448
	$h/8$	1.937528E-01	1.996769	3.334088E-03	3.992753	3.601128E-05	8.009718	3.493854E-07	15.989758
	$h/16$	9.691559E-02	1.999191	8.339036E-04	3.998170	4.498358E-06	8.005428	2.183746E-08	15.999361
	$h/32$	4.846270E-02	1.999798	2.084998E-04	3.999541	5.620954E-07	8.002837	1.364771E-09	16.000824
P3	h	9.240404E-01	—	1.821920E-01	—	2.604170E-02	—	2.970825E-03	—
	$h/2$	4.719950E-01	1.957733	4.835779E-02	3.767583	3.293234E-03	7.907637	1.930138E-04	15.391775
	$h/4$	2.372405E-01	1.989521	1.229169E-02	3.934186	4.121108E-04	7.991137	1.219399E-05	15.828601
	$h/8$	1.187766E-01	1.997367	3.085998E-03	3.983052	5.152107E-05	7.998879	7.642363E-07	15.955785
	$h/16$	5.940785E-02	1.999342	7.723235E-04	3.995732	6.440302E-06	7.999791	4.779921E-08	15.988471
E2	h	2.649210E+00	—	3.595189E-01	—	3.193651E-02	—	2.524193E-03	—
	$h/2$	1.318714E+00	2.008934	8.993155E-02	3.997695	3.927106E-03	8.132327	1.610897E-04	15.669487
	$h/4$	6.531989E-01	2.018855	2.241408E-02	4.012279	4.871867E-04	8.060782	1.012826E-05	15.904973
	$h/8$	3.255080E-01	2.006706	5.597219E-03	4.004503	6.070882E-05	8.024974	6.340125E-07	15.974859
	$h/16$	1.626051E-01	2.001831	1.398866E-03	4.001255	7.578446E-06	8.010721	3.964223E-08	15.993361
	$h/32$	8.128354E-02	2.000468	3.496879E-04	4.000327	9.467285E-07	8.004878	2.481653E-09	15.974123
E3	h	1.794166E+00	—	1.033938E+00	—	3.444999E-01	—	7.299233E-02	—
	$h/2$	1.641967E+00	1.092693	3.300826E-01	3.132361	4.712105E-02	7.310956	5.367026E-03	13.600145
	$h/4$	8.555156E-01	1.919272	8.813140E-02	3.745346	5.973199E-03	7.888746	3.505492E-04	15.310336
	$h/8$	4.332360E-01	1.974710	2.246687E-02	3.922727	7.486531E-04	7.978594	2.219947E-05	15.790881
	$h/16$	2.176447E-01	1.990565	5.648322E-03	3.977618	9.368404E-05	7.991256	1.392836E-06	15.938323

Table 1: h-performance for fixed p: P2 (Poisson 2D), P3 (Poisson 3D), E2 (Elasticity 2D), E3 (Elasticity 3D).

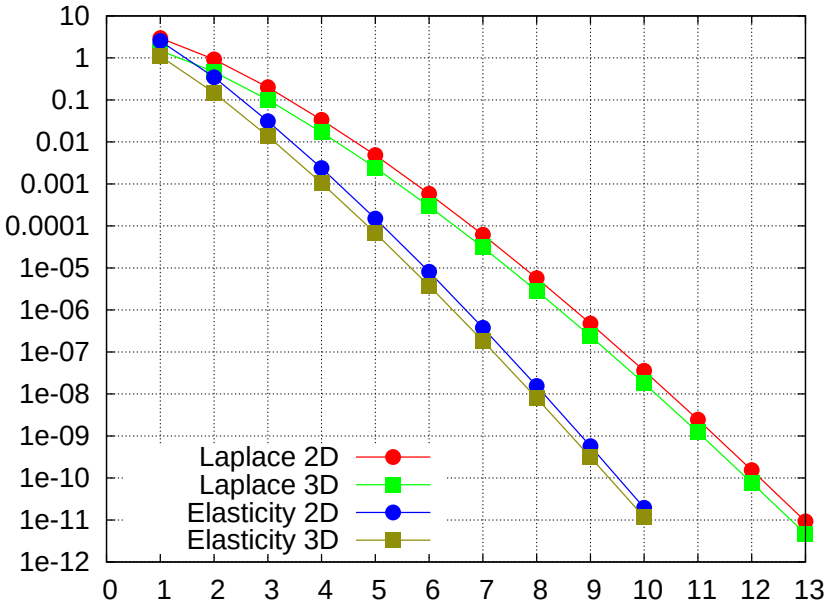


Figure 5: p-performance for fixed h: $\mathbf{H}^1$-accuracy in function of the polynomial degrees.

We made also some tests consisting in considering several values of the parameter α. For all values of α, the results highlight that the estimator η_α provides an efficient estimation of the exact precision as predicted theoretically. In addition, the decrease of the exact precision agrees well with the decrease of the error estimator η_α. In fact, the estimations by η_α are somewhat influenced by the values of the chosen α. Nonetheless, the values of η_α are comparatively of the same order up to some constant scaling factors.

For the choice between h-refinement (local mesh refinement) and p-refinement (elevation of the local polynomial degree), the algorithm of Melenk and Wohlmuth in [32] is adopted. One considers the mean value $(\eta^{\mathbf{mean}})^2 = (1/NEL)\sum_{T\in\mathcal{M}}(\eta^{\mathbf{loc}}_{\alpha,T})^2$ and the elements which are marked for refinements are those such that $(\eta^{\mathbf{loc}}_{\alpha,T})^2 \geq \sigma(\eta^{\mathbf{mean}})^2$ where $\sigma \in]0,1[$. The objective is to obtain errors which are almost evenly distributed over the whole mesh. It estimates some error prediction $\eta^{\mathbf{pred}}_T$ of the local error based on the current a-posteriori error $\eta^{\mathbf{loc}}_{\alpha,T}$ and the local polynomial degree $p(T)$. In the case the a-posteriori error $\eta^{\mathbf{loc}}_{\alpha,T}$ is larger than the error prediction $\eta^{\mathbf{pred}}_T$, one applies an h-refinement. Otherwise a p-refinement is applied. The prediction $\eta^{\mathbf{pred}}_T$ is initialized to be zero for each element T of the coarsest mesh. The prediction is updated after each level by using some user defined parameters γ_h, γ_p and γ_n. Those parameters specify the predicted errors from the a-posteriori errors if an h-refinement, p-refinement or no refinement is applied. We adjusted that algorithm [32] for the present approach because the original algorithm in 2D subdivides each square element into 4

LEV	2D-Poisson			2D-Navier-Lamé		
	DOF	$\mathbf{H}^1$-precision	$e^{-b\sqrt[3]{DOF}}$	DOF	$\mathbf{H}^1$-precision	$e^{-b\sqrt[3]{DOF}}$
0	4	1.107624E-01	2.649138E-01	8	8.815498E-02	2.648880E-01
1	5	1.103790E-01	2.390882E-01	10	8.809492E-02	2.390632E-01
2	13	1.077094E-01	1.397896E-01	20	8.770824E-02	1.648068E-01
3	29	1.073813E-01	7.646354E-02	36	8.706460E-02	1.115570E-01
4	53	9.971684E-02	4.314053E-02	50	8.562632E-02	8.655094E-02
5	89	9.144426E-02	2.384604E-02	68	6.639722E-02	6.646144E-02
6	145	7.870403E-02	1.232349E-02	94	5.511038E-02	4.879548E-02
7	201	6.185181E-02	7.432974E-03	164	5.029478E-02	2.636375E-02
8	257	4.598861E-02	4.891222E-03	206	4.415346E-02	1.978428E-02
9	285	4.036611E-02	4.058640E-03	256	3.949101E-02	1.473456E-02
10	349	3.054567E-02	2.762501E-03	444	3.336900E-02	6.299483E-03
11	389	1.299518E-02	2.223630E-03	546	2.808065E-02	4.387905E-03
12	437	4.660595E-03	1.746417E-03	620	2.048220E-02	3.469132E-03
13	493	3.832219E-03	1.346005E-03	896	1.124385E-02	1.655756E-03
14	557	1.572809E-03	1.022898E-03	984	7.033895E-03	1.351387E-03
15	629	4.042444E-04	7.694428E-04	1,202	4.888993E-03	8.567602E-04
16	953	1.034520E-04	2.652396E-04	1,550	3.338474E-03	4.587395E-04
17	1,277	1.137615E-05	1.140711E-04	2,022	8.960010E-04	2.250312E-04
18	1,757	7.360366E-06	4.117962E-05	2,510	2.331511E-04	1.201331E-04
19	2,157	4.458395E-06	2.015441E-05	3,302	6.283955E-05	5.062823E-05
20	2,249	2.433081E-06	1.731962E-05	4,902	2.325025E-06	1.258035E-05
21	2,841	1.161687E-06	7.125928E-06	7,618	4.822623E-07	2.108407E-06
22	3,901	4.347448E-07	1.901241E-06	10,016	1.798561E-07	6.050828E-07
23	4,357	1.545851E-07	1.159540E-06	12,474	4.996987E-08	2.041158E-07
24	5,717	2.251104E-08	3.174668E-07	16,060	1.132655E-08	5.271890E-08
25	7,449	1.006277E-08	7.987819E-08	19,030	3.372471E-09	1.988127E-08
26	8,513	3.355966E-09	3.796796E-08	22,562	8.000827E-10	7.058949E-09
27	9,937	1.445934E-09	1.537449E-08	27,408	4.071905E-10	2.007069E-09
28	10,973	8.442681E-10	8.398452E-09	31,972	2.080706E-10	6.988078E-10
29	13,029	4.828502E-10	2.808004E-09	36,566	9.953700E-11	2.662628E-10
30	16,669	2.018127E-10	5.205461E-10	41,452	4.906904E-11	1.038840E-10

Table 2: Two dimensional *hp*-convergence by using $p_{\max} = 9$ for the Poisson and the Navier-Lamé admitting the material properties $(\lambda, \mu) = (2, 1)$.

LEV	3D-Poisson			3D-Navier-Lamé		
	DOF	$\mathbf{H}^1$	$e^{-b\sqrt[3]{DOF}}$	DOF	$\mathbf{H}^1$	$e^{-b\sqrt[3]{DOF}}$
0	21	3.81048E-03	5.68031E-01	63	6.59995E-03	5.50538E-01
1	27	3.80946E-03	5.40641E-01	81	6.59819E-03	5.22562E-01
2	35	3.79708E-03	5.11418E-01	267	6.59219E-03	3.80644E-01
3	133	3.78926E-03	3.51194E-01	357	6.57491E-03	3.45040E-01
4	205	3.77641E-03	2.98569E-01	543	6.56485E-03	2.94126E-01
5	423	3.70100E-03	2.14624E-01	1,209	6.47061E-03	2.02308E-01
6	633	3.63861E-03	1.72015E-01	2,076	6.30864E-03	1.47560E-01
7	1,019	3.35010E-03	1.27085E-01	2,790	5.90435E-03	1.21036E-01
8	1,531	3.11533E-03	9.41638E-02	4,278	4.43860E-03	8.75949E-02
9	2,133	2.51039E-03	7.14422E-02	6,789	3.53437E-03	5.84098E-02
10	3,153	1.81730E-03	4.94871E-02	9,540	2.35681E-03	4.15308E-02
11	4,019	1.36151E-03	3.84147E-02	12,282	1.99425E-03	3.14035E-02
12	5,735	1.19859E-03	2.54911E-02	20,298	1.75844E-03	1.67100E-02
13	6,913	1.07537E-03	2.01370E-02	34,179	1.20732E-03	7.68926E-03
14	8,451	8.21117E-04	1.53653E-02	54,213	5.09351E-04	3.42351E-03
15	10,397	4.95044E-04	1.13984E-02	112,845	1.51172E-04	7.11194E-04
16	16,229	2.37751E-04	5.57087E-03	252,216	2.64908E-05	7.65746E-05
17	22,097	1.32811E-04	3.17448E-03	454,446	5.55873E-06	9.80740E-06
18	27,911	6.60301E-05	1.99242E-03	832,266	1.61573E-06	7.45276E-07
19	33,743	1.46060E-05	1.32754E-03	1,273,572	1.22724E-07	8.68401E-08
20	39,611	3.10167E-06	9.22798E-04	1,670,292	1.63081E-08	1.86513E-08
21	45,425	8.89079E-07	6.65805E-04	—	—	—
22	51,257	8.69060E-08	4.92986E-04	—	—	—
23	184,613	1.33682E-08	8.52260E-06	—	—	—
24	300,971	5.29646E-09	1.08042E-06	—	—	—
25	417,611	1.00433E-09	2.21460E-07	—	—	—
26	528,527	2.61980E-10	6.33498E-08	—	—	—
27	692,291	7.41567E-11	1.33071E-08	—	—	—
28	1,054,325	2.30800E-11	8.68117E-10	—	—	—
29	1,323,029	8.49995E-12	1.68364E-10	—	—	—
30	2,022,995	3.58131E-12	5.49605E-12	—	—	—

Table 3: Three dimensional *hp*-convergence by using $p_{\max} = 9$ for the Poisson and the Navier-Lamé admitting the material properties $(\lambda, \mu) = (2, 1)$.

sub-squares of the same size where hanging nodes could appear. We use 2D and 3D bisections where a few elements could be refined without being marked [6, 7]. The bisection refinement could spread locally in the vicinity of the marked elements but the spreading is known to be finite and no domino effect could occur. In the case that an element T has not been marked for refinement but it is anyway refined and it is flagged to obtain a p-refinement, we modify the flag so that it has only an h-refinement. The goal is to avoid an over-refinement where an element is both p-refined and h-refined simultaneously. We used it for both the Poisson equation and the Navier-Lamé equation where the elastic properties are $(\lambda, \mu) = (2, 1)$. The parameters taken for the Poisson problem are $\sigma = 0.75$, $\gamma_h = 0.4$, $\gamma_p = 5.0$, $\gamma_n = 1.0$ whereas those for the Navier-Lamé equation are $\sigma = 0.75$, $\gamma_h = 0.6$, $\gamma_p = 9.0$, $\gamma_n = 1.0$. By using $p_{\max} = 9$, the convergence related to the 2D hp-adaptivity is collected in Table 2 where the $\mathbf{H}^1$-accuracies $e_{\mathbf{H}^1}$ are displayed alongside the value of $\exp[-b(DOF)^{1/3}]$. The exact solution is taken as $u(x, y) = x(x-1)y(y-1)\exp[(x-0.5)^2 + (y-0.5)^2]$ for the 2D-Poisson equation whereas we take that function componentwise for the 2D-Elasticity. We consider refinements in the range level$=0, ..., 30$ and the expected accuracy of $e_{\mathbf{H}^1} \leq C\exp[-b(DOF)^{1/3}]$ is achieved. The used value in the $2D$ case amounts to $b \approx 0.83$ and $b \approx 0.66$ for the Poisson and the Navier-Lamé equations respectively. A similar computing has been done in the 3D case whose results are shown in Table 3. The values of the parameters γ_h, γ_p and γ_n remain as in the 2D case. The exact solution is $u(x, y, z) = x(x-1)y(y-1)z(z-1)\exp[(x-0.5)^2 + (y-0.5)^2 + (z-0.5)^2]$ for the 3D-Poisson and it is used componentwise in the 3D-elasticity. We have also in the 3D case the same exponential convergence $\exp[-b(DOF)^{1/3}]$ where we use $b \approx 0.21$ and $b \approx 0.15$ for the Poisson and elasticity respectively. The 3D-case of the Navier-Lamé equation is aborted at level 20, on account of memory capacity. For the polynomial degree $p_{\max} = 9$, the matrix for the 3D case is rather dense because each tetrahedron contains $\mathcal{O}(p^3)$ local DOF for each of the three components. We want now to show the benefit of hp-adaptivity over other refinements where we consider the exact solution in both coordinates $x(x-1)y(y-1)\exp\big[-\beta((x-0.25)^2 + (y-0.75)^2)\big]$ in which $\beta = 1.0E+4$ and the elastic material properties are $(\lambda, \mu) = (5, 1.25)$. We take the estimator $\eta_{0.5}$ and we consider the polynomial degrees $p = 1, 2, 3$. Our next purpose is to compare the uniform refinements and the adaptive ones. The degree of freedom is used as a common reference measurement for all the computational tests. The outcome of the simulation is plotted in Fig. 6 which displays the absolute $\mathbf{H}^1$-accuracy in term of the degree of freedom. The starting mesh consists of a tensor product grid having 18 elements. The advantage of adaptivity over the uniform refinements is as follows. For the polynomial degrees $p = 1, 2, 3$, we need 588290, 588290, 1324802 DOFs

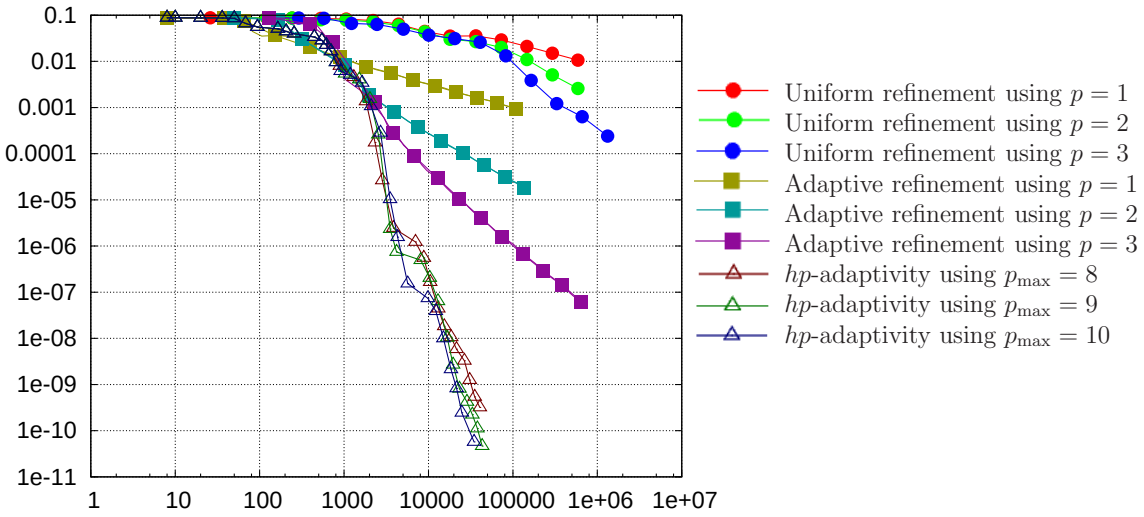


Figure 6: $\mathbf{H}^1$-accuracy as function of DOF for various refinements and polynomial degrees. Elastic properties $(\lambda, \mu) = (5, 1.25)$.

to achieve the accuracies of $1.063621E-02$, $2.591986E-03$, $2.404479E-04$ respectively in the case of uniform refinements. In contrast, the adaptive case requires only 108738, 136882, 645146 DOFs to produce more precise accuracies $9.490625E-04$, $1.852298E-05$, $6.224150E-08$. In addition, we perceive that for the elastic material property $(\lambda, \mu) = (5, 1.25)$, the use of higher order FEM ($p \geq 2$) already proves more advantageous than the piecewise linear ($p = 1$) setting although that material can still be categorized as absolutely compressible. Indeed, for comparable DOF$\approx$ 100000, the adaptive simulations using $p = 1, 2, 3$ yield respectively the accuracies of $9.490625E-04$ (DOF=108738), $2.576346E-05$ (DOF=99282) and $1.039906E-06$ (DOF=104444). As observed in Fig. 6, that advantage of higher order computation over the piecewise linear case is also plainly perceived in the uniform case. We consider the maximal degrees $p_{\max} = 8, 9, 10$ for the hp-computations. All three outperform the adaptive simulations using fixed polynomial degrees. For the case $p_{\max} = 8$, we obtained for $DOF = 40590$ the accuracy of $3.193302E-10$. The value $DOF = 43214$ for $p_{\max} = 9$ yields the accuracy $4.630825E-11$. As for $p_{\max} = 10$, an accuracy of $5.695637E-11$ is obtained from $DOF = 34642$. In the current implementation, we need to fix the $p_{\max}$ because our program constructs some set-up phase to accelerate the computations. Extending the program to the case where $p_{\max}$ is not specified is a work on progress.

6.2 Data distribution and parallel processing

We survey here the implementation infrastructure which is needed to carry out the parallel multilevel p-FEM solver. The features of our parallel mesh processing are as follows:

- Simplices other than nodes and elements are involved for the p-FEM while the program operates in parallel for 2D and 3D.
- It supports parallel adaptive or uniform refinements where load balanced meshes are distributed among the processors. Ghost tightening and parallel mesh redistributions are supported.
- Element attributes are used for a domain $\mathbf{\Omega}$ which is composed of subdomains (e.g. internal/external subdomains in the case of interface problems). Those attributes serve as identification of the belonging of an element to a subregion. They are also needed if one assigns elastic material properties to the subdomains.
- Boundary conditions are represented as 1D/2D manifold pieces. The mapping of a manifold element (segment in 2D/triangle in 3D) onto a volume element (triangle in 2D/tetrahedron in 3D) is updated during parallel mesh refinements.
- For a mesh $\mathcal{M}^{(\ell)}$ refined into a mesh $\mathcal{M}^{(\ell+1)}$, cascading is used to exactly represent higher order FE-functions in $\mathcal{M}^{(\ell)}$ as functions in $\mathcal{M}^{(\ell+1)}$ in any degree p.
- We have scalar/vector valued FE-functions in an arbitrary polynomial degree p for the cascading process which is updated during the parallel mesh refinements.

Next, we want to survey those features briefly.

6.2.1 Load balancing

For a piecewise linear FEM setting, only node and tetrahedral information are required. Additional information are needed for higher order FEM setting: nodes (0-simplex), edges (1-simplex), triangular faces (2-simplex), tetrahedral cells (3-simplex), in order to ensure global continuity of the p-FEM shape functions at incident elements. The mesh $\mathcal{M}$ is decomposed into N_p sub-meshes where N_p is the number of processors so that every processor generates, stores and updates its own information concerning its own simplices. That is, $\mathcal{M} = \bigcup_{i=1}^{N_p} \mathcal{M}_i^{\text{loc}}$ such that $\mathcal{M}_i^{\text{loc}} \cap \mathcal{M}_j^{\text{loc}}$ is not always completely disjoint. As an illustration, we have in Fig. 1 a 3D situation where some meshes are

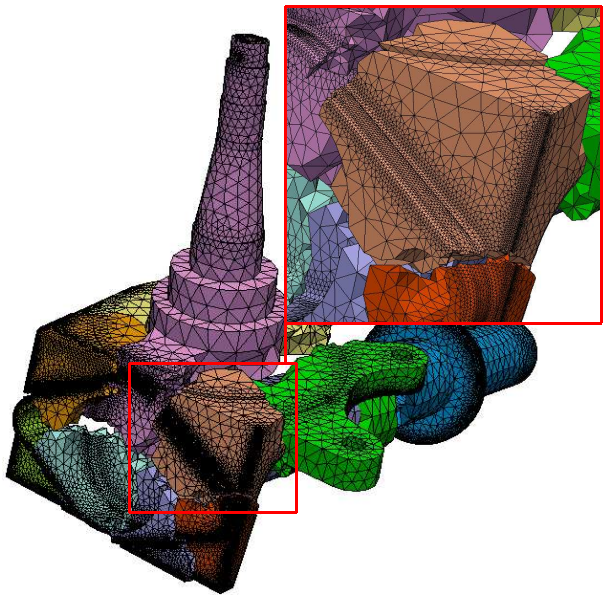


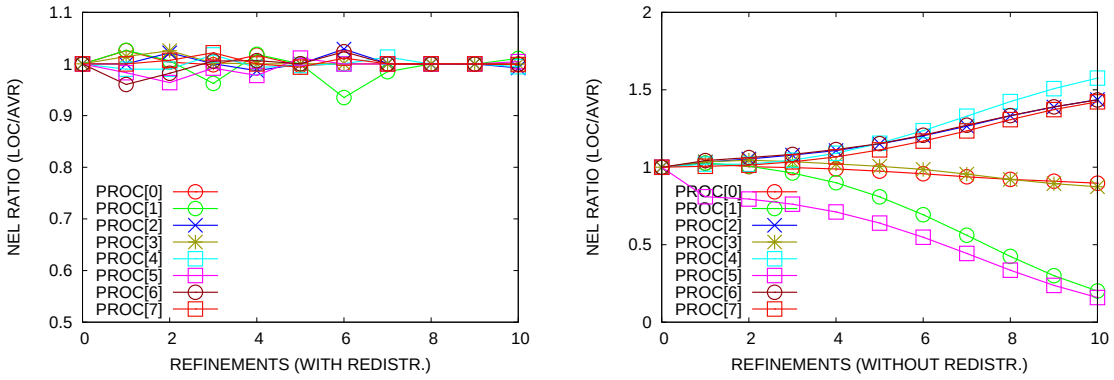
Figure 7: Mesh distribution among the processors

decomposed onto the processors. We organize synchronous indexation of the simplices at the inter-domain where some σ-simplices are shared by different processors. For a FEM simulation using polynomials of degree p, every processor manages: (1)the σ-simplices which are strictly members of its subdomain where $\sigma \in \mathcal{J}_p$,(2)the few shared duplicated σ-simplices related to the neighboring subdomain for all $\sigma \in \mathcal{J}_p$, (3)the local σ-simplices incident upon every local tetrahedron. The reason for having such a non-disjoint mesh decomposition is that during the subsequent FEM matrix assembly, each unknown FEM-variable on a local processor requires all information related to the tetrahedra where its corresponding FEM shape function is supported. Only the initial coarse mesh is stored by the root processor. All subsequent tasks are performed in a distributed manner: refinements, generation and storing of the geometric simplices. The size of the parallel data is only limited by the available computer memory and the processor count. The objective of the distribution is to assign to all processors comparable amount of computational tasks. The complete mesh $\mathcal{M}$ is decomposed into N_p submeshes admitting two properties. First, the submeshes are approximately of similar size. Second, the measures of the area shared by adjacent submeshes are as small as possible. As a consequence, the FEM-tasks assigned to the processors are balanced: matrix assembly, numerical quadrature for the right hand side, the linear solver and the a-posteriori estimators. Additionally, the communications between the adjacent processors are minimal. The way of obtaining the distributed decomposition is to first split $\mathcal{M}$ into $\mathcal{M} = \bigcup_{i=1}^{N_p} \widetilde{\mathcal{M}}_i^{\text{loc}}$ such that the submeshes $\widetilde{\mathcal{M}}_i^{\text{loc}}$ are mutually

Proc.	$\sigma = 0$		$\sigma = 1$		$\sigma = 2$		$\sigma = 3$	
	count	ratio	count	ratio	count	ratio	count	ratio
0	130,767	1.0044	843,290	1.0396	1,410,500	1.0395	697,976	1.0394
1	128,687	0.9885	811,763	1.0008	1,357,966	1.0008	672,126	1.0009
2	131,004	1.0062	822,910	1.0145	1,376,150	1.0142	680,923	1.0140
3	128,860	0.9898	799,451	0.9856	1,337,560	0.9858	662,115	0.9860
4	130,968	1.0060	822,638	1.0142	1,375,831	1.0140	680,825	1.0138
5	128,853	0.9897	798,000	0.9838	1,335,909	0.9846	661,552	0.9851
6	132,085	1.0146	807,937	0.9960	1,350,355	0.9952	667,976	0.9947
7	130,300	1.0008	783,179	0.9655	1,310,582	0.9659	648,882	0.9662
All	1,041,524		6,489,168		10,854,853		5,372,375	

Table 4: Load-balancing of the data among 8 parallel processors.

disjoint. Thereafter, each submesh $\mathcal{M}_i^{\text{loc}}$ is deduced by enlarging $\widetilde{\mathcal{M}}_i^{\text{loc}}$. Every σ-simplex $\widetilde{\mathcal{M}}_i^{\text{loc}}$ is assigned to one and only one supporting processor P_i. Those simplices will be termed the *adherent* simplices with respect to their corresponding processor P_i or submeshes $\widetilde{\mathcal{M}}_i^{\text{loc}}$, $\mathcal{M}_i^{\text{loc}}$. A σ-simplex s is a *ghost* simplex w.r.t. a processor P_i if s is adherent to another processor P_j and if s is incident upon a tetrahedron having an adherent simplex belonging to P_i. The enlargement margin between $\widetilde{\mathcal{M}}_i^{\text{loc}}$ and $\mathcal{M}_i^{\text{loc}}$ is composed of the ghost simplices. Every processor stores its own adherent simplices and duplications of ghost simplices from incident submeshes. One employs a graph which is assembled by using a nodal distribution method. It utilizes a graph $\mathcal{G}$ based on the element-element incidence of the mesh. One applies to the resulting graph $\mathcal{G}$ a partitioning algorithm which consists in decomposing the graph $\mathcal{G}$ into N_p subgraphs $\mathcal{G}_i^{\text{loc}}$ of similar size where the number of graph-edges to be cut is minimal. For the implementation, we have used METIS to achieve that task about graph partitioning. Inter-processor information dispatching is unavoidable in order that the global mapping is coherent. Since the number of graph-edges to be cut from the graph $\mathcal{G}$ has been made minimal, the inter-process data transfer is also kept minimal. We examine now the sizes of the distributed data among the processors. An efficient FEM computation requires approximately the same amount of simplices in all processors. In Table 4, we gather the distribution of the data in which 8 processors indexed from 0 to 7 are used. In that table, we investigate the difference between the ideal number of σ-simplices and their actual number in the submeshes $\mathcal{M}_i^{\text{loc}}$. That is quantified by the ratio between the average number of the σ-simplices and the number of the ones stored and generated



(a) (b)

Figure 8: Load balancing during refinements, ratio between local and average NEL for each processor: (a)With redistribution, (b)Without redistribution.

by each processor. Although it is impossible to obtain the ideal ratio of unity for all simplices, it can be observed that the proposed method enables a good balance on the processors because all ratios approximate the unity in all σ-simplices. In fact, the ranges of the ratio are $[0.9897, 1.0146]$, $[0.9655, 1.0396]$, $[0.9659, 1.0395]$, $[0.9662, 1.0394]$ for the nodes, edges, triangles, and tetrahedra respectively.

6.2.2 Refinement redistribution

Most mesh refinements are based on two major methods: the red-green refinement [11] and the bisection [7, 6]. We have used in this document the bisection approach because it is simpler to ensure the nestedness (2.16) of the meshes. We adopt the method in [6, 7] in which a bisection refinement is proposed where the aspect ratios are guaranteed to be bounded. That method ensures the quasi-uniformity (2.17) of the sequence of meshes $\mathcal{M}^{(\ell)}$. For local refinements, not only the marked elements by the error estimator are refined, but also a few nearby elements. The method is guaranteed to terminate when that local refinement spreading is applied. While the parallel implementation of the bisection is not complicated except at the ghost region shared by the processors, the procedure for load balancing is very involved. In fact, depending on the solution of the PDE and the a-posteriori values, it is possible that the elements marked by the estimator are accumulated on some processors while the meshes in other processors are almost left unchanged. Therefore, we need a redistribution which consists of two stages: (1)computing the new position of each element, (2)transferring

Nb. all tetra.	Average nel	Min local nel	Max local nel
17,252	2,156.5	2,156	2,157
106,626	13,328.25	13,154	13,571
177,866	22,233.25	21,654	22,852
488,807	61,100.88	60,036	62,135
1,339,353	167,419.13	164,121	169,853
3,744,518	468,064.75	452,012	485,463
10,582,995	1,322,874.38	1,281,907	1,364,748
30,348,046	3,793,505.75	3,401,494	3,880,868

Table 5: Global and local number of tetrahedra among processors after refinements WITH redistribution.

the required mesh entities and updating the ghost region. That involves the assembly of the parallel graph corresponding to the element-element adjacency of the global mesh. After a parallel graph partitioning using PARMETIS format, the new positions of the local mesh entities are determined. In the typical case for the refinement $\mathcal{M}^{(\ell)}$ to $\mathcal{M}^{(\ell+1)}$, only a few local mesh entities need to be migrated while a large part of the local mesh is kept unchanged. An efficient inter-processor procedure is used to transfer as few information as possible. In addition, in each local mesh, local tasks are minimized to incorporate only the new mesh entities and to discard the obsolete ones. During the mesh refinements, we update also the boundary manifolds and the cascading procedure which are described in the next sections. In order to show the importance of redistributions, we start from the same number of tetrahedra in Tab. 5 and Tab. 6 where the former uses redistribution while the latter omit mesh redistributions. For both tests, the whole tetrahedra are distributed among 8 processors. As the refinement proceeds, it is observed in Tab. 5 that the minimal/maximal numbers of tetrahedra in each processor do not differ too much from the desired average number of elements. That implies that the elements are well balanced among the processors. In contrast, the difference between the minimal and maximal numbers of local elements is considerable in Tab. 6. When the number of global tetrahedral reaches $30,348,046$, the maximal number of local elements $(1,594,221)$ is almost 3 times as large as the minimal one $(5,574,153)$. That ratio is highlighted more clearly in Fig. 8 where we use 10 levels of refinements. The ratios between the local number of elements and the average number of elements are almost constantly unity when redistributions are utilized. As the refinement level grows, omitting the redistribution process leads to a very unbalanced load.

Nb. all tetra.	Average nel	Min local nel	Max local nel
17,252	2,156.5	2,156	2,157
106,626	13,328.25	7,343	16,292
177,866	22,233.25	11,074	27,910
488,807	61,100.88	26,914	83,736
1,339,353	167,419.13	71,574	236,144
3,744,518	468,064.75	197,558	676,460
10,582,995	1,322,874.38	556,057	1,934,980
30,348,046	3,793,505.75	1,594,221	5,574,153

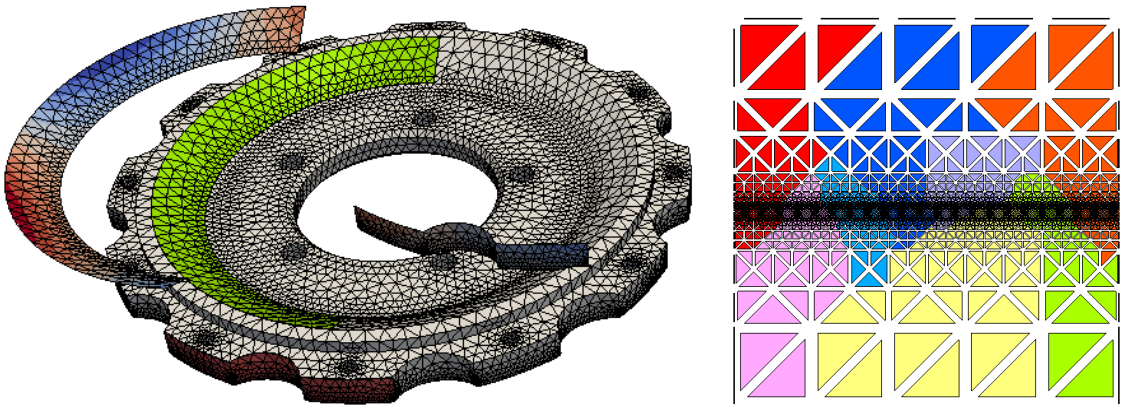
Table 6: Global and local number of tetrahedra among processors after refinements WITHOUT redistribution.

6.2.3 Manifolds for boundary conditions

Our structure represents boundary conditions in form of manifold meshes on some part of the boundary $\partial\mathbf{\Omega}$ as illustrated in Fig. 9. It enables different types of boundary conditions on various manifolds. It supports a mapping of a manifold element (segment in 2D/triangle in 3D) onto a volume element (triangle in 2D/tetrahedron in 3D). Some faces of $\mathbf{\Omega}$ which is represented as a B-rep model can be chosen where different faces can be combined to form a single manifold. In the case of 2D/3D problems, we have 1D/2D manifolds respectively. We have also some mapping μ of the p-basis functions $\phi_i^{\mathbf{\Gamma}}$ on the manifold mesh toward the p-basis functions $\phi_{\mu[i]}$ in the volume mesh. That functionality is useful when one wants to handle some functions on the boundary manifold only. That applies also in the case that some functions defined on the boundary have to be carried toward the volume mesh of the domain $\mathbf{\Omega}$. That is the case when considering nonhomogeneous Dirichlet or Neumann boundary conditions. An instance is the computation of the load function in the case of Neumann boundary condition:

$$I_i^{\mathbf{\Gamma}} = \int_{\mathbf{\Gamma}} F(\mathbf{x})\phi_i^{\mathbf{\Gamma}} d_{\mathbf{\Gamma}}(\mathbf{x}), \qquad R(\mu[i]) = \int_{\mathbf{\Omega}} R(\mathbf{x})\phi_{\mu[i]} d\mathbf{x} - I_i^{\mathbf{\Gamma}}. \tag{6.191}$$

The right hand side using the volume integral has to be substracted from some integral defined on a boundary manifold. Another instance is the computation of the a-posteriori error estimator where some values on the manifolds are needed. It simplifies also the interpolation or the projection of any function onto the manifold piece as piecewise p-polynomials. In our current implementation, the processor which contains the volume block equally contains the corresponding manifold piece which maps onto



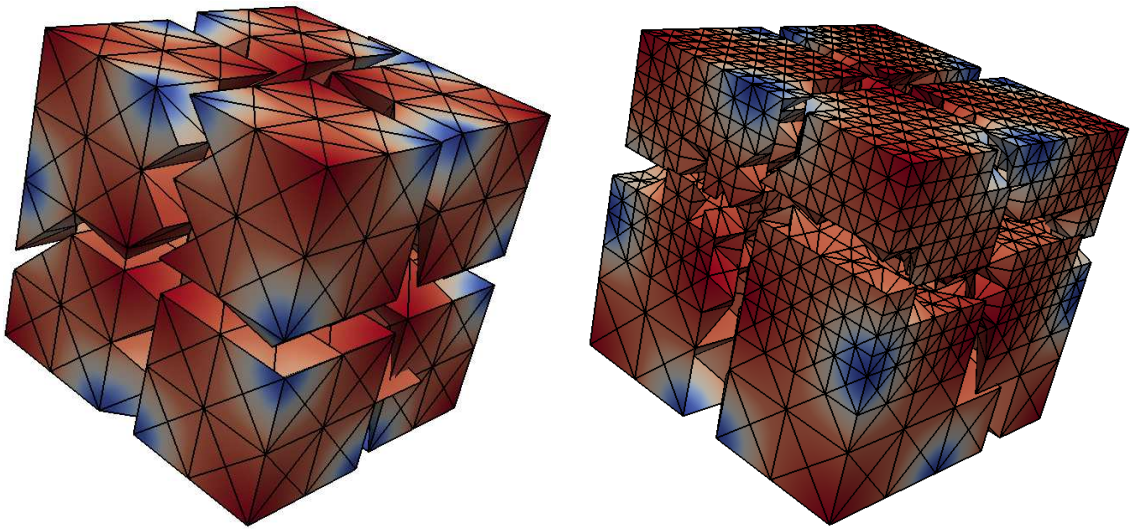
(a) (b)

Figure 9: Boundary condition: (a)Facial manifold mesh mapped onto volume mesh, (b)Manifolds are updated during volume mesh refinement and redistribution among 8 processors.

that block. The manifolds are updated as one modifies the volume mesh: ghost tightening, refinements, redistributions as illustrated in Fig. 9(b). The backward mapping is not assembled because some volume elements are unmapped and different manifold elements may map onto the same volume element. That situation typically occurs in the case of corners (see the top-left triangle in the unit square of Fig. 9(b)). Just like the volume meshes, the manifold meshes can contain also some attribute information in order to distinguish which manifold piece corresponds to which type of boundary condition.

6.2.4 Scalar/vector valued cascading

The importance of cascading is in the determination of the initial guess of the CG-iteration. Instead of starting from zero initial guess or a random initial guess on $\mathcal{M}^{(L)}$, it is advantageous to use the last solution from the previous mesh $\mathcal{M}^{(L-1)}$. That reduces the total number of CG-iterations significantly although the error reduction between two consecutive CG-iterates remains unchanged. Sometimes, two CG-iterations are sufficient to achieve the desired accuracy when starting from a good initial guess. In order to achieve that cascading procedure, it is important to have a functionality of exactly expressing a p-FEM function in $\mathcal{M}^{(\ell)}$ in term of a function in $\mathcal{M}^{(\ell+1)}$ during the refinement tasks. We have scalar/vector valued FE-functions using arbitrary polynomial degrees for the cascading process. In the case of scalar-valued PDE such as the



(a) (b)

Figure 10: Cascading for 8 processors: (a)A piecewise polynomial of degree $p = 3$ on a coarse mesh $\mathcal{M}^{(\ell_1)}$, (b)The same function expressed on a finer mesh $\mathcal{M}^{(\ell_2)} \supset \mathcal{M}^{(\ell_1)}$. The mesh distributions are load-balanced.

Laplace equation, a scalar-valued cascading suffices. For the case of vector-valued one such as the Navier-Lamé equation, the component functions need to be treated as well. The cascading procedure is performed together with the mesh processing: refinement, ghost tightening, redistribution. That is, when there is some redistribution of the mesh elements, requiring some data transfer, the FEM-values have to be shifted also by using inter-processor communications. For the piecewise linear setup ($p = 1$), the local cascading is easily done because the values at the nodes completely determine the FE-values. Thus, the value at the midpoint is the average of the two edge endpoint values during each edge bisection. That is, the complicated tasks are those related to mesh load balancing between the processors. As for the higher order setting ($p \geq 2$), one has to locally determine the new values in each element when one performs a bisection task. We do the bisections on the reference element, and make a transformation toward the elements to be bisected. Some look-up tables are precomputed on the reference element. For a polynomial degree p in dimension d, two local matrices of size $\binom{p+d}{p}$ are exactly inverted by using an exact solver provided by some LAPACK routines. In Fig. 10, an instance of cascading for the scalar-valued case is displayed. We consider there a piecewise polynomial u of degree $p = 3$ in the unit cube. The function in Fig. 10(a) corresponds to the projection of $\sin\big[2\pi(x + 0.25)\big] \sin\big[2\pi(y + 0.25)\big] \sin\big[2\pi(z + 0.25)\big]$

on the piecewise polynomial space on a coarse mesh. We apply a parallel mesh refinement and redistribution using 8 processors where the upper part above the half plane of the cube is refined. The same piecewise function is shown on the finer mesh on Fig. 10(b). The load is balanced:(1)since the local meshes on the upper blocks are denser, it can be observed in Fig. 10(b) that the mesh volumes on the upper part are smaller than those on the lower part, (2)the local numbers of elements for all processors remain comparable.

6.3 Iteration counts

In this section, we would like to provide some numerical results related to the number of iterations. We consider the Poisson and the elasticity equations in both 2D and 3D. We apply here a cascading approach, i.e. the initial guess of the CG-iteration on the current level is derived from the last CG-iterate of the previous level. The abortion criterion of the CG is when the CG-residual is dropped below 10^{-6}. We consider uniform polynomial degrees p ranging from 1 to 4. NEL stands for the number of elements. By denoting by u^{INI}, u^{FEM}, u^{PDE} the initial guess, the FEM-solution targeted by the CG-iteration and the PDE-solution respectively, we have $\|u^{\mathrm{FEM}} - u^{\mathrm{INI}}\| \leq \|u^{\mathrm{FEM}} - u^{\mathrm{PDE}}\| + \|u^{\mathrm{PDE}} - u^{\mathrm{INI}}\|$ which shows the importance of having a good initial guess. As the FE-solution gets closer to the PDE-solution, the number of required CG-iterations becomes small. That is more closely observed in Table 11. For the special case of polynomial degree $p = 4$ in 3D simulation, the number of local DOF is already very large. The size of the ASM block per node n is of order $(m/6)(p+1)(p+2)(p+3)$ where m denotes the number of tetrahedral elements admitting n as an apex. Due to its large size, we decompose the blocks further into sub-blocks having much lower local DOF. We use than the sub-blocks instead of the original ASM blocks. For the other cases (the 2D-case and the 3D-case where $p \leq 3$), that method using sub-blocks does not seem to be necessary. Our experience about the *hp*-simulation is that it requires a very precise accuracy of the CG-solver. In contrast to the fixed polynomial setting, the CG-residual should be dropped to the order between 10^{-12} and 10^{-15} in order to achieve the Galerkin accuracy of the *hp*-FEM computation. The proposed theory estimates only the number of required iterations which is not the only important factor when considering a preconditioner. Other important practical factors in choosing a preconditioner is a tight memory requirements, (2)speed of each iteration. The presented preconditioner $P^{\mathbf{HPL}}$ needs only to store $A(\phi_i^{(\ell)}, \phi_i^{(\ell)})$ for every level ℓ. That means that an extra-storage of $\mathcal{O}(N)$ is adequate to store the preconditioner where N denotes the number of nodes on the finest level. The cost of computation is equivalent to that storage.

Level	NEL	$p=1$		$p=2$		$p=3$		$p=4$	
		DOF	#ITER	DOF	#ITER	DOF	#ITER	DOF	#ITER
1	18	4	2	25	7	64	8	121	9
2	36	13	6	61	8	145	8	265	8
3	72	25	7	121	7	289	7	529	6
4	144	61	10	265	11	613	7	1,105	6
5	288	121	11	529	9	1,225	6	2,209	3
6	576	265	13	1,105	11	2,521	6	4,513	3
7	1,152	529	14	2,209	8	5,041	5	9,025	2
8	2,304	1,105	17	4,513	9	10,225	5	18,241	2
9	4,608	2,209	15	9,025	6	20,449	4	36,481	1
10	9,216	4,513	18	18,241	8	41,185	4	73,345	1
11	18,432	9,025	16	36,481	5	82,369	3	146,689	1
12	36,864	18,241	19	73,345	5	165,313	3	294,145	1
13	73,728	36,481	15	146,689	5	330,625	2	588,289	1
14	147,456	73,345	18	294,145	5	662,401	1	1,178,113	1
15	294,912	146,689	14	588,289	4	1,324,801	1	2,356,225	1
16	589,824	294,145	17	1,178,113	4	2,651,905	1	4,715,521	1
17	1,179,648	588,289	14	2,356,225	3	5,303,809	1	9,431,041	1
18	2,359,296	1,178,113	15	4,715,521	3	10,612,225	1	18,868,225	3
19	4,718,592	2,356,225	12	9,431,041	3	21,224,449	1	37,736,449	1
20	9,437,184	4,715,521	14	18,868,225	2	42,458,113	3	—	—
21	18,874,368	9,431,041	11	37,736,449	3	—	—	—	—
22	37,748,736	18,868,225	12	—	—	—	—	—	—

Table 7: 2D-Laplace: PCG-iteration counts by using cascading and bisection as refinement.

Level	NEL	$p=1$		$p=2$		$p=3$		$p=4$	
		DOF	#ITER	DOF	#ITER	DOF	#ITER	DOF	#ITER
1	18	8	4	50	10	128	11	242	11
2	36	26	9	122	11	290	11	530	10
3	72	50	11	242	12	578	12	1,058	8
4	144	122	15	530	16	1,226	12	2,210	10
5	288	242	16	1,058	14	2,450	11	4,418	6
6	576	530	21	2,210	18	5,042	13	9,026	6
7	1,152	1,058	21	4,418	15	10,082	10	18,050	3
8	2,304	2,210	26	9,026	17	20,450	10	36,482	4
9	4,608	4,418	25	18,050	12	40,898	6	72,962	2
10	9,216	9,026	28	36,482	14	82,370	6	146,690	2
11	18,432	18,050	26	72,962	9	164,738	5	293,378	1
12	36,864	36,482	29	146,690	10	330,626	5	588,290	1
13	73,728	72,962	26	293,378	7	661,250	3	1,176,578	1
14	147,456	146,690	28	588,290	7	1,324,802	4	2,356,226	1
15	294,912	293,378	25	1,176,578	6	2,649,602	2	4,712,450	1
16	589,824	588,290	27	2,356,226	6	5,303,810	3	9,431,042	1
17	1,179,648	1,176,578	24	4,712,450	6	10,607,618	1	18,862,082	4
18	2,359,296	2,356,226	26	9,431,042	6	21,224,450	2	—	—
19	4,718,592	4,712,450	24	18,862,082	5	—	—	—	—
20	9,437,184	9,431,042	25	37,736,450	5	—	—	—	—
21	18,874,368	18,862,082	23	—	—	—	—	—	—

Table 8: 2D-Elasticity: PCG-iteration counts by using cascading and bisection as refinement.

Level	NEL	$p=1$		$p=2$		$p=3$		$p=4$	
		DOF	#ITER	DOF	#ITER	DOF	#ITER	DOF	#ITER
1	192	21	1	187	5	689	7	1,719	8
2	384	27	1	343	5	1,331	7	3,375	7
3	768	91	3	855	7	3,059	8	7,471	8
4	1,536	235	5	1,815	8	6,275	8	15,151	7
5	3,072	343	6	3,375	9	12,167	8	29,791	5
6	6,144	855	9	7,471	11	25,991	9	62,559	8
7	12,288	2,199	12	15,919	12	53,447	8	127,071	6
8	24,576	3,375	9	29,791	9	103,823	7	250,047	3
9	49,152	7,471	12	62,559	12	214,415	8	512,191	4
10	98,304	18,991	14	133,215	11	440,975	7	1,040,575	3
11	196,608	29,791	11	250,047	7	857,375	5	2,048,383	1
12	393,216	62,559	16	512,191	11	1,742,111	7	4,145,535	3
13	786,432	157,791	17	1,089,727	11	3,582,239	5	8,421,759	2
14	1,572,864	250,047	13	2,048,383	6	6,967,871	3	16,581,375	1
15	3,145,728	512,191	19	4,145,535	7	14,045,759	5	—	—
16	6,291,456	1,286,335	21	8,814,975	8	—	—	—	—
17	12,582,912	2,048,383	16	16,581,375	6	—	—	—	—
18	25,165,824	4,145,535	23	—	—	—	—	—	—

Table 9: 3D-Laplace: PCG-iteration counts by using cascading and bisection as refinement.

Level	NEL	$p=1$		$p=2$		$p=3$		$p=4$	
		DOF	#ITER	DOF	#ITER	DOF	#ITER	DOF	#ITER
1	24	3	1	45	3	195	7	525	13
2	48	3	1	81	4	375	9	1,029	12
3	96	27	2	273	8	1,023	11	2,565	15
4	192	63	1	561	10	2,067	10	5,157	15
5	384	81	1	1,029	11	3,993	10	10,125	17
6	768	273	9	2,565	15	9,177	15	22,413	21
7	1,536	705	13	5,445	17	18,825	15	45,453	23
8	3,072	1,029	14	10,125	16	36,501	15	89,373	15
9	6,144	2,565	18	22,413	21	77,973	16	187,677	20
10	12,288	6,597	25	47,757	24	160,341	15	381,213	17
11	24,576	10,125	21	89,373	18	311,469	13	750,141	9
12	49,152	22,413	28	187,677	24	643,245	12	1,536,573	12
13	98,304	56,973	36	399,645	26	1,322,925	13	3,121,725	9
14	196,608	89,373	28	750,141	17	2,572,125	9	6,145,149	5
15	393,216	187,677	37	1,536,573	23	5,226,333	10	—	—
16	786,432	473,373	43	3,269,181	24	10,746,717	9	—	—
17	1,572,864	750,141	31	6,145,149	14	—	—	—	—
18	3,145,728	1,536,573	41	12,436,605	19	—	—	—	—
19	6,291,456	3,859,005	47	—	—	—	—	—	—
20	12,582,912	6,145,149	36	—	—	—	—	—	—
21	25,165,824	12,436,605	43	—	—	—	—	—	—

Table 10: 3D-Elasticity: PCG-iteration counts by using cascading and bisection as refinement.

Level	NEL	DOF	#ITER		$\mathbf{H}^1$-accuracy	$\mathbf{L}_2$-accuracy
			p-MDS	ParaSail		
1	18	25	7	6	1.483362E+00	7.804007E-02
2	72	121	10	10	4.442698E-01	1.019997E-02
3	288	529	13	17	1.176018E-01	1.294636E-03
4	1,152	2,209	14	32	2.986315E-02	1.627913E-04
5	4,608	9,025	13	60	7.496013E-03	2.039017E-05
6	18,432	36,481	9	115	1.875926E-03	2.565685E-06
7	73,728	146,689	5	210	4.691064E-04	3.846821E-07
8	294,912	588,289	4	417	1.172908E-04	1.496870E-07
9	1,179,648	2,356,225	6	808	2.932161E-05	1.256920E-08
10	4,718,592	9,431,041	3	1,413	7.331019E-06	8.361510E-09
11	18,874,368	37,736,449	3	2,918	1.833412E-06	6.159913E-09

Table 11: Comparison with ParaSail solver for $p = 2$.

The contribution of the local higher order term $P^{\mathbf{LHO}}$ is computed on the fly as it applies only locally on the highest level. More results for the hp-simulation will be defered in a subsequent article because the cascading process is not yet available in that case.

ParaSail is a general purpose linear solver which is not restricted to PDE applications. It treats sparse matrices and it functions in parallel using an MPI-implementation. It uses a sparse approximate inverse. That is achieved by using an approximation in the least-square sense [17] in term of the Froebenius norm. It does not exploit the hierarchy in the nested mesh refinement. It only treats the final mesh and it constructs its own hierarchy internally. We adopt only the default values of the parameters. In term of higher order FEM, ParaSail appears to perform better in 3D than in 2D for a reason that we do not know. The ParaSail implementation is described in [18]. We use it only as a black-box solver by converting our data to its structure. For both tests, the abortion criterion of the CG is 10^{-6}. Our own experience concerning the parallel use of ParaSail for the 3D higher order FEM is documented in [37] which includes the Poisson-Boltzmann equation admitting highly discontinuous coefficients.

Conclusion and outlook

We have exposed a linear solver for higher order FEM which is applied to elliptic systems. We do not assume $\mathbf{H}^2$ smoothness of the solution and the preconditioner lower and upper bounds have been thoroughly analyzed. Concerning the memory consumption, the preconditioner is light-weight, indeed it necessitates only $\mathcal{O}(N)$ extra-storage where N is the number of nodes. The numerical results corroborate the efficiency of the solver for both the Laplace and the Navier-Lamé equations. The solver has been successfully compared with a black-box solver. We investigated the numerical convergence and the number of iterations in 2 and 3 dimensions.

As an outlook, the presented preconditioner can be used for other purposes. For instance, in the case of solving generalized eigenvalue problems originating from PDE, say $Ax = \lambda Mx$ where A is the stiffness matrix and M is the mass matrix. The LOBPCG (Locally Optimal Block Preconditioned Conjugate Gradient) is a standard iterative method for searching for eigenvalues and their respective eigenvectors. In every iteration of LOBPCG, it is needed to provide a preconditioner $P \approx (A - \lambda M)^{-1}$. For a very small value of λ, it is typical to just use a preconditioner for A, whereas larger eigenvalues are more difficult to approximate by LOBPCG using $P \approx A^{-1}$. Thus, the use of a black-box preconditioner does not appear to be optimal in such a case. With that regard, the advantage of the current preconditioner over some other ones is that it only accesses $A(\phi_i^{(\ell)}, \phi_i^{(\ell)})$ and $M(\phi_i^{(\ell)}, \phi_i^{(\ell)})$ on every level ℓ. Thus, for a certain iterate λ_k provided by the LOBPCG algorithm, accessing $A(\phi_i^{(\ell)}, \phi_i^{(\ell)}) - \lambda_k M(\phi_i^{(\ell)}, \phi_i^{(\ell)})$ is adequate to construct a preconditioner of $(A - \lambda M)$, at least when λ_k is close to convergence (implied from the LOBPCG residuals). The initial guess for the eigenvectors can be deduced from the results of the previous level by using cascading as presented in this document. We already have an implementation of LOBPCG but it has not yet been coupled with this preconditioner. A combination of cascading and the presented preconditioner would provide an efficient multilevel LOBPCG performance. Another possible utility of this preconditioner is in the solving of saddle-point problems whose system is composed of blocks including a major block for elliptic systems. Using the standard Bramble-Pasciak-CG to solve such a system would benefit from this preconditioner for the elliptic block.

References

[1] V. ADAMS, A. ASKENAZI, *Building better products with finite element analysis*, OnWord Press, 1999.

[2] M. AINSWORTH, *Pyramid algorithms for Bernstein-Bézier finite elements of high, non-uniform order in any dimension*, SIAM J. Sci. Comput. **36**, No. 2, pp. 543–569, 2014.

[3] M. AINSWORTH, G. ANDRIAMARO, O. DAVYDOV, *Bernstein-Bézier finite elements of arbitrary order and optimal assembly procedures*, SIAM J. Sci. Comput. **33**, No. 6, 3087–3109, 2011.

[4] M. AINSWORTH, O. DAVYDOV, L. SCHUMAKER, *Bernstein-Bézier finite elements on tetrahedral-hexahedral-pyramidal partitions*, Computer Methods in Applied Mathematics and Engineering **304**, pp. 140–170, 2016.

[5] M. AINSWORTH, T. ODEN, *A Posteriori error estimators for the Stokes and Oseen equations*, SIAM J. Numer. Anal. **34**, pp. 228–245, 1997.

[6] D. ARNOLD, A. MUKHERJEE, *Tetrahedral bisection and adaptive finite elements*, In: Grid generation and adaptive algorithms (Minneapolis, MN). Springer, Editors: M. Bern, J. Flahery and M. Luskin, 1997.

[7] D. ARNOLD, A. MUKHERJEE, L. POULY, *Locally adapted tetrahedral meshes using bisections*, SIAM J. Sci. Comput. **22**, No. 2, pp. 431–448, 2000.

[8] I. BABUSKA, M. SURI, *Locking effects in the finite element approximation of elasticity problems*, Numer. Math. **62**, No. 1, pp. 439–463, 1992.

[9] I. BABUSKA, M. SURI, *The hp-version of the finite element method with quasiuniform meshes*, ESAIM: Mathematical Modelling and Numerical Analysis **21**, No. 2, pp. 199–238, 1987.

[10] I. BABUSKA, *On the h, p and $h-p$ version of the finite element method*, Tatra Mountains Math. Publ. **4**, No. 5–18, 1994.

[11] R. BANK, A. SHERMAN, A. WEISER, *Some refinement algorithms and data structures for regular local mesh refinement*, In Scientific Computing, R. Stepleman et al. **44**, IMACS North-Holland, Amsterdam. pp. 3–17, 1983.

[12] C. BERNARDI, *Indicateurs d'erreur en h-N version des éléments spectraux*, Modélisation Mathématique et Analyse Numérique **30**, No. 1, pp. 1–38, 1996.

[13] C. BERNARDI, M. DAUGE, Y. MADAY, *Polynomials in the Sobolev world*, Preprint of the Laboratoire Jacques-Louis Lions, No. R03038, 2003.

[14] C. BERNARDI, Y. MADAY, *Polynomial interpolation results in Sobolev spaces*, Journal of Computational and Applied Mathematics **43**, No. 1–2, pp. 53–80, 1992.

[15] J. BRAMBLE, J. PASCIAK, J. XU, *Parallel Multilevel Preconditioners*, Math. Comp. **55**, pp. 1–22, 1990.

[16] R. BRUALDI, J. QUINN MASSEY, *Incidence and strong edge colorings of graphs*, Discrete Mathematics **122**, pp. 51–58, 1993.

[17] E. CHOW, *Parallel implementation and performance characteristics of least squares sparse approximate inverse preconditioners*, Tech. Report UCRL-MA-137863, Lawrence Livermore Nat. Lab., 2000.

[18] E. CHOW, *A priori sparsity patterns for parallel sparse approximate inverse preconditioners*, SIAM J. Sci. Comput. **21**, pp. 1804–1822, 2000.

[19] E. CREUSÉ, G. KUNERT, S. NICAISE, *A posteriori error estimation for the Stokes problem: anisotropic and isotropic discretizations*, Mathematical Models and Methods in Applied Sciences **14**, No. 9, pp. 1297–1341, 2004.

[20] P. DANIEL, A. ERN, I. SMEARS, M. VOHRAL, *An adaptive hp-refinement strategy with computable guaranteed bound on the error reduction factor*, 2018.

[21] L. DEMKOWICZ, W. RACHOWICZ, P. DEVLOO, *A fully automatic hp-adaptivity*, J. Scientific Computing **17**, No. 1–4, pp. 117-142, 2002.

[22] C. DIEDRICH, D. DIJKSTRA, J. HAMAEKERS, B. HENNINGER, M. RANDRIANARIVONY, *A finite element study on the effect of curvature on the reinforcement of matrices by randomly distributed and curved nanotubes*, Journal of Computational and Theoretical Nanoscience **12**, pp. 2108–2116, 2015.

[23] T. EIBNER, J. MELENK, *Multilevel preconditioning for the boundary concentrated hp-FEM*, Comp. Math. Mech. Eng. **196**, pp. 3713–3725, 2007.

[24] T. EIBNER, J. MELENK, *An adaptive strategy for hp-FEM based on testing for analyticity*, Comput. Mech. **39**, No. 5, pp. 575–595, 2007.

[25] S. FUNKEN, D. PRAETORIUS, P. WISSGOTT, *Efficient implementation of adaptive P1-FEM in MATLAB*, Comput. Methods Appl. Math. **11**, No. 4, pp. 460–490, 2011.

[26] H. HARBRECHT, M. RANDRIANARIVONY, *From Computer Aided Design to wavelet BEM*, Comput. Vis. Sci. **13**, No. 2, pp. 69–82, 2010.

[27] H. HARBRECHT, P. ZASPEL, *A scalable H-matrix approach for the solution of boundary integral equations on multi-GPU clusters*, 2018.

[28] R. KIRBY, *Efficient discontinuous Galerkin finite element methods via Bernstein polynomials*, Arxiv Preprint.

[29] R. KIRBY, *Fast simplicial finite element algorithms using Bernstein polynomials*, Numer. Math. **117**, No. 4, pp. 631–652, 2011.

[30] Y. MADAY, *Analysis of spectral projectors in one-dimensional domains*, Math. Comp. **55**, No. 192, pp. 537–562, 1990.

[31] J. MELENK, *hp-interpolation of non-smooth functions*, SIAM J. Numer. Anal. **43**, pp. 127–155, 2005.

[32] J. MELENK, B. WOHLMUTH, *On residual-based a posteriori error estimation in hp-FEM*, Adv. Comput. Math. **15**, No. 1–4, pp. 311-331, 2002.

[33] J. ODEN, A. PATRA, Y. FENG, *An hp adaptive strategy*, In: Adaptive Multilevel, and Hierarchical Computational Strategies. ASME **157**, pp. 23–46, 1992

[34] M. RANDRIANARIVONY, *Finite Element polynomial inverse estimates and elastostatic higher order estimator*, preprint 2018.

[35] M. RANDRIANARIVONY, *Adaptive mesh for elasticity under traction boundary condition using higher order FEM*, preprint 2018.

[36] M. RANDRIANARIVONY, *Anisotropic finite elements for the Stokes problem: a-posteriori error estimator and adaptive mesh*, Journal of Computational and Applied Mathematics **169**, No. 2, pp. 255–275, 2004.

[37] M. RANDRIANARIVONY, *Parallel processing of analytical Poisson-Boltzmann using higher order FEM*, ACTA Press, No. 455434, pp. 1–6, 2013.

[38] M. RANDRIANARIVONY, *Stability of mixed finite element methods on anisotropic meshes*, Master's thesis, Faculty of mathematics, Technische Universität Chemnitz, 2001.

[39] J. SCHOEBERL, J. MELENK, C. PECHSTEIN, S. ZAGLMAYR, *Additive Schwarz preconditioning for p-version triangular and tetrahedral finite elements*, IMA. J. Numer. Anal. **28**, pp. 1–24, 2008.

[40] L. SCOTT, S. ZHANG, *Finite element interpolation of nonsmooth functions satisfying boundary conditions*, Math. Comp. **54**, No. 190, pp. 483–493, 1990.

[41] S. SHERWIN, G. KARNIADAKIS, *A new triangular and tetrahedral basis for high-order (hp) finite element methods*, Internat. J. Numer. Meths. Engrg. **38**, pp. 3775–3802, 1995.

[42] S. SHERWIN, G. KARNIADAKIS, *Tetrahedral hp finite elements: Algorithms and flow simulations*, J. Comput. Phys. **124**, pp. 14–45, 1996.

[43] M. SURI, *The p-version of the finite element method for elliptic equations of order* $2l$, Modélisation mathématique et analyse numérique **24**, No. 2, pp. 265–304, 1990.

[44] M. SURI, *The p and hp finite element method for problems on thin domains*, Journal of Computational and Applied Mathematics **128**, pp. 235-260, 2001.

[45] M. VOGELIUS, *An analysis of the p-version of the finite element method for nearly incompressible materials. Uniformly valid, optimal error estimates*, Numer. Math. **41**, No. 1, pp. 39–53, 1983.

[46] X. ZHANG, *Multilevel Schwarz methods*, Numer. Math. **63**, pp. 521–539, 1992.