Hierarchical multigrid approaches for the finite cell method on uniform and multi-level hp-refined grids

J. Jomo*[1], O. Oztoprak[1], F. de Prenter[2], N. Zander[1], S. Kollmannsberger[1], and E. Rank[1]

[1]Chair of Computational Modeling and Simulation, Technische Universität München
[2]Research Develpoment Netherlands, Hengelo, Netherlands

Abstract

This contribution presents a hierarchical multigrid approach for the solution of large-scale finite cell problems on both uniform grids and multi-level hp-discretizations. The proposed scheme leverages the hierarchical nature of the basis functions utilized in the finite cell method and the multi-level hp-method, which is attributed to the use of high-order integrated Legendre basis functions and overlay meshes, to yield a simple and elegant multigrid scheme. This simplicity is reflected in the fact that all restriction and prolongation operators reduce to binary matrices that do not need to be explicitly constructed. The coarse spaces are constructed over the different polynomial orders and refinement levels of the immersed discretization. Elementwise and patchwise additive Schwarz smoothing techniques are used to mitigate the influence of the cut cells leading to convergence rates that are independent of the cut configuration, mesh size and in certain scenarios even the polynomial order. The multigrid approach is applied to second-order problems arising from the Poisson equation and linear elasticity. A series of numerical examples demonstrate the applicability of the scheme for solving large immersed systems with multiple millions and even billions of unknowns on massively parallel machines.

Keywords: immersed methods, finite cell method, multigrid, iterative solvers, hp-refinement, parallel computing

1 Introduction

Immersed finite element methods allow numerical simulations to be easily performed on domains with a complex shape by employing a discretization that does not need to capture the boundaries of the body under analysis. The suitability of these methods to handle a variety of geometrical input-data, such as CAD files, scan images, implicitly-defined geometries and oriented point clouds, has led to the emergence of several immersed schemes. Noteworthy immersed methods include the finite cell method (FCM) [1, 2], cutFEM [3, 4], the aggregated unfitted method (AgFEM) [5], the Cartesian grid finite element method [6, 7] and the shifted boundary method [8, 9].

The widespread use of immersed methods for solving problems of engineering relevance was for a long time inhibited by the conditioning problems associated with cut elements, i.e. elements that are intersected by the boundary of the original body. In [10] the underlying cause of ill-conditioning

*Corresponding author: john.jomo@tum.de

in immersed methods is systematically analyzed for uniform grids and shown to be the occurrence of basis functions that are not only small but also almost linear dependent. Over the past few years, different approaches have been devised to address the conditioning problems of immersed methods such as different preconditioning strategies e.g. [11–13] and methods based on basis function manipulation, removal or aggregation, e.g. [5, 14–16]. These techniques enable iterative solvers to be applied to immersed systems and have opened the door for large-scale immersed finite element analysis.

One possibility of further improving the convergence properties of (immersed) linear systems and the efficiency of the solution process itself is by the use of multigrid solution techniques. The central idea behind multigrid techniques is to accelerate the convergence of a fine problem by incorporating correction terms generated from a hierarchy of coarser problems. These terms approximate the smooth components of the error, hereby leading to fast convergence of smooth modes that generally converge slowly when standard iterative solvers such as the Conjugate Gradient (CG) method are used. The high-frequency errors are commonly treated in a smoothing process within the multigrid algorithm. There is a vast amount of literature on multigrid methods. Classical works with a more theoretical perspective include [17, 18], while [19–21] provide a more implementation-oriented introduction into the subject. Multigrid techniques are generally classified as either geometric or algebraic methods. Geometric multigrid methods generate coarse problems using geometrical information. In finite element analysis, this translates to the generation of coarse problems based on a sequence of discretizations with varying element sizes (h-multigrid), polynomial orders (p-multigrid) or both (hp-multigrid). Algebraic multigrid methods generate their coarse problems solely from the fine system and are usually applied in scenarios where geometrical information is not available or cannot be easily obtained [22].

Multigrid methods have been successfully applied to boundary-conforming finite element methods based on the p-version of the finite element method e.g. [23, 24] and isogeometric analysis e.g. [25–27], yielding convergence rates that are independent of the grid-size h. Convergence rates that are independent of the polynomial order p are, however, harder to achieve and are dependent on the type of smoother employed, as well as the problem type. In most cases, standard smoothers such as the Jacobi or Gauss-Seidel methods result in convergence rates that deteriorate with increasing values of p, as shown in [26]. In [27], it is shown that using an ILUT smoother in an isogeometric p-multigrid framework results in p-independent convergence rates for the problem classes considered therein. Likewise, smoothers based on the multiplicative Schwarz algorithm have been shown to yield convergence rates independent of p for two-dimensional problems in isogeometric analysis [28]. The use of different smoothers in the context of high-order isogeometric analysis is discussed in [29–31].

Approaches based on multigrid cycles have also been utilized to solve linear systems derived from immersed finite element methods. An algebraic multigrid solver is utilized in [32] to solve large problems with up to 300 million unknowns using AgFEM on meshes consisting of linear finite elements. It is also used in [33, 34] to solve systems with up to 480 million unknowns involving local h-refinements using an adaptive h-AgFEM framework. Reference [35] employs an algebraic multigrid method for the solution of cutFEM discretizations of the electroencephalography (EEG) forward problem. In [36] an h-multigrid approach that employs a multiplicative Schwarz smoother is presented and convergence rates independent of the cut configuration and the element size are reported for Lagrange and B-spline bases in problems with up to 10 million unknowns. Reference [37] employs a geometric multigrid approach to solve finite cell systems arising from the Poisson equation on a square domain. This paper shows promising developments for the solution of large FCM systems but only reports results of two-dimensional problems with linear elements on which local h-refinement is performed.

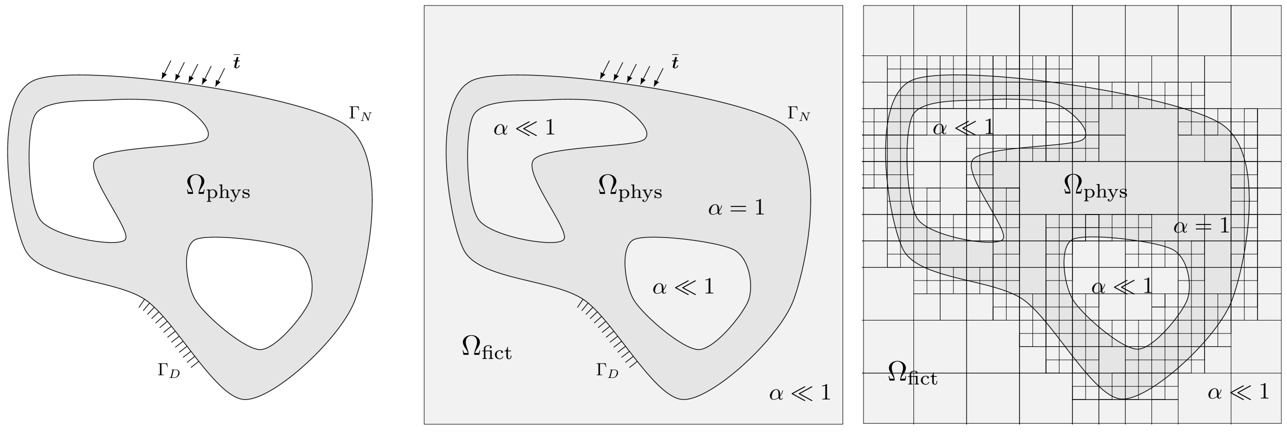
As previously mentioned, it is possible to construct the nested spaces of the multigrid algorithm in high-order and hp-adaptive finite element methods based on the values of the polynomial order p and the different refinement levels of the mesh. In a p-multigrid, the nested subspace of a certain polynomial order is simply spanned by the basis functions up to and including this order. p-multigrid methods were first developed in the context of the p-version of the finite element method [23] and the spectral element method [38]. Since then p-multigrid approaches have been extensively studied, see e.g. [23, 24, 39–41] and applied to different problem classes. Most of these methods define the multigrid spaces by using an arithmetic sequence where the polynomial order is successively lowered until $p = 1$, see [42]. It is, however, also possible to define the V-cycles by using a geometrical sequence of p, as suggested in [24], by choosing either only odd or only even polynomial orders. hp-multigrid methods can be applied on both uniform and refined grids, and generally define the hierarchy of coarse spaces by first performing a p-coarsening of the mesh until the $p = 1$ level before applying an h-coarsening. In the case of uniform meshes, h-coarsening is usually applied using a standard h-multigrid approach, with the coarsest multigrid level comprising a few large elements e.g. [43]. Hierarchical hp-multigrid approaches are commonly applied on refined grids and often replace the standard h-refined linear nodal basis with a hierarchical basis, see e.g. [42, 44]. Mitchel et al. apply a hierarchical hp-approach to refined triangular meshes in [42] and show that the computational complexity of a single V-cycle is $\mathcal{O}(M/p)$, where M is the number of nonzero entries in the stiffness matrix.

The main objective of this contribution is to present a hierarchical multigrid method for the solution of linear systems arising from the finite cell method on both uniform grids and multi-level hp-discretizations. The proposed scheme takes advantage of the hierarchical nature of the basis functions in the finite cell method [1, 2] and the multi-level hp-scheme [45, 46]. The FCM implementation used in this work, is based on the p-version of the finite element method (p-FEM) [47] and utilizes high-order integrated Legendre shape functions that are hierarchical in nature, i.e. the set of basis functions of order p contains all basis functions from order 1 up to p. Likewise, the superposition principle used to perform spatial refinement in the multi-level hp-method results in a hierarchical basis. The suggested multigrid scheme is therefore elegant and efficient, simple in implementation and not strongly interwoven with the code, since all restriction and prolongation operators reduce to binary matrices which do not need to be explicitly constructed. Furthermore, our scheme makes use of additive Schwarz smoothing, and builds upon the work done on additive Schwarz preconditioning for uniform and multi-level hp-refined finite cell systems in [13]. The overall algorithm results in convergence rates that are independent of the mesh resolution, refinement level and, under certain conditions, the polynomial order. The multigrid technique put forward in this contribution is used to solve large finite cell systems with up to 3.2 billion degrees of freedom (DOFs) and shown to be suitable for large-scale finite cell analyses on massively parallel machines.

The paper at hand is organized into five sections. The core features of the finite cell method and multi-level hp-refinement are highlighted in Section 2. Section 3 presents a hierarchical multigrid framework for FCM and the multi-level hp-method focusing on the construction of the coarse spaces and the choice of smoothers to guarantee robustness in the immersed setting. A series of numerical examples are handled in Section 4. These examples are comprised of simple benchmark test cases as well as large-scale applications that attest the suitability of the proposed solution scheme for large-scale finite cell analyses. Section 5 concludes the paper, providing a summary and an outlook.

2 The finite cell method and multi-level hp-refinement

The following section presents the fundamental ideas behind the finite cell method and the multi-level hp-method. These discretization techniques are used in tandem to perform numerical simulations on domains with complex geometries. Noteworthy application areas include the analysis of bone-implant systems [48], the simulation of metal additive manufacturing processes [49] and the modeling of crack growth in brittle structures by means of a phase-field approach [50].



(a) Domain of computation. (b) Fictitious domain extension. (c) Structured *hp*-refined FCM mesh.

Figure 1: Illustration of the main idea behind the finite cell method.

2.1 Ingredients of the finite cell method

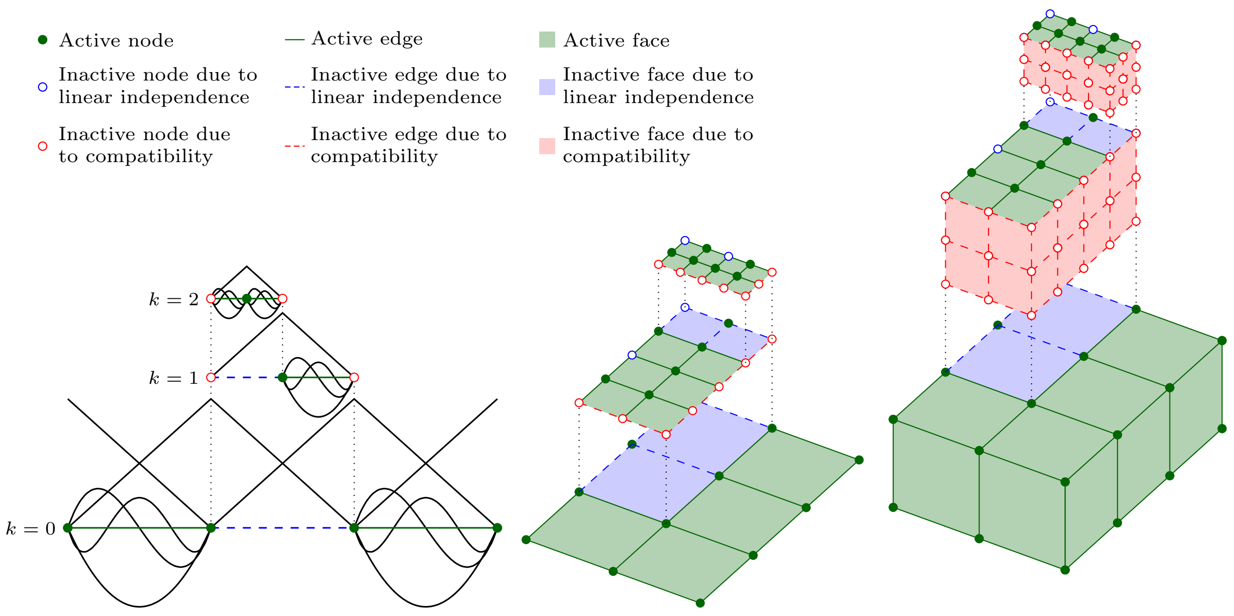
The finite cell method (FCM) is an immersed finite element method that combines a fictitious domain approach with high-order finite elements making it suitable for performing FE-analysis on bodies with very complex geometries. Consider a body defined by the physical domain Ω_{phys} and boundary Γ with $\Gamma = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$. Γ_D represents the part of the boundary where Dirichlet boundary conditions g_D are applied while Γ_N represents the Neumann boundary with applied traction g_N. The main idea of the finite cell method is to place Ω_{phys} in an immersing (fictitious) domain Ω_{fict} yielding an overall computational domain Ω of simple shape that can be easily discretized by regular finite elements as illustrated in Figure 1. An indicator function α is commonly used to distinguish between points lying within Ω_{phys}, which are associated with the value $\alpha = 1$, and points lying in Ω_{fict}, where alpha is chosen as a small constant $\alpha = \epsilon$ with $\epsilon \ll 1$. This indicator function is applied to the weak form in the FCM as shown in Equation 1 and can be interpreted as a penalization of the fictitious domain that results in an asymptotically consistent method recovering the original weak form when $\alpha \to 0$.

$$
\begin{aligned}
a(\mathbf{u}_h, \mathbf{v}_h) &= \int (\mathbf{B}\mathbf{v}_h)^T \alpha(\boldsymbol{x}) \mathbf{C}\mathbf{B}\mathbf{u}_h \, \mathrm{d}\Omega + \int \beta \, (\mathbf{N}\mathbf{v}_h)^T \mathbf{N}\mathbf{u}_h \, \mathrm{d}\Gamma_D \\
b(\mathbf{v}_h) &= \int \alpha(\boldsymbol{x}) (\mathbf{N}\mathbf{v}_h)^T \mathbf{f} \, \mathrm{d}\Omega + \int (\mathbf{N}\mathbf{v}_h)^T g_N \, \mathrm{d}\Gamma_N + \int \beta \, (\mathbf{N}\mathbf{v}_h)^T g_D \, \mathrm{d}\Gamma_D
\end{aligned} \tag{1}
$$

Equation (1) shows the discretized weak form of a finite cell problem in the case of linear elasticity with a symmetric bilinear form $a(\mathbf{u}_h, \mathbf{v}_h)$ and linear functional $b(\mathbf{v}_h)$. The term $\mathbf{B}$ denotes the linear strain operator, $\mathbf{N}$ the shape function vector, $\mathbf{C}$ the elasticity tensor and $\boldsymbol{f}$ a prescribed volumetric force. The unknown displacement field vector and test functions are represented by $\mathbf{u}_h$ and $\mathbf{v}_h$, respectively. The penalty method [51] is used in (1) to impose the Dirichlet boundary conditions and β denotes the penalty parameter.

Various kinds of embedding meshes can be employed in the finite cell method. We use a mesh based on *p*-FEM that utilizes integrated Legendre polynomials. The hierarchical nature of these basis functions can be exploited to construct a multigrid algorithm as outlined in Section 3.

The fictitious domain approach employed in the finite cell method has two main benefits. First, it significantly simplifies the meshing process, as the generation of (high-order) body-fitting meshes can be especially challenging for complex geometries in three dimensions. Secondly, it yields a versatile discretizational framework that can be applied in a straightforward manner to different types of


Active node
Inactive node due to linear independence
Inactive node due to compatibility
Active edge
Inactive edge due to linear independence
Inactive edge due to compatibility
Active face
Inactive face due to linear independence
Inactive face due to compatibility
$k = 2$
$k = 1$
$k = 0$

(a) One-dimensional case (b) Two-dimensional case (c) Three-dimensional case

Figure 2: Illustration of the multi-level *hp*-refinement scheme with two refinement levels, $k = 2$, in different spatial dimensions. The deactivation of specific topological components following a simple rule-set ensures compatibility and linear independence of the basis functions [46].

geometric models such as those stemming from CAD systems, constructive solid geometry, scan images and even oriented point clouds. The fictitious extension, however, has several implications that make the simulation pipeline in FCM, and immersed methods in general, differ from that of conventional body-fitted methods. Three core aspects that require special treatment in immersed methods include *i)* the numerical integration on cut elements, *ii)* the application of Dirichlet boundary conditions and *iii)* the conditioning of the system. A comprehensive description of each of these areas is beyond the scope of this article, as it focuses only on the conditioning and solution of the resulting linear systems. The interested reader is directed to the review articles on the finite cell method [52, 53] for a comprehensive overview of cut cell integration techniques and the weak imposition of Dirichlet boundary conditions.

2.2 The multi-level *hp*-method

The multi-level *hp*-method presented in [45, 46] is a novel *hp*-method that performs spatial refinement based on superposition. The scheme was developed to circumvent the challenges associated with constraining arbitrary levels of hanging nodes, while at the same time preserving the desirable characteristics of classical *hp*-formulations. It employs high-order overlay meshes which are placed over coarse elements in critical areas of the mesh. The original formulation of the *hp*-scheme is based on the *p*-version of the finite element method and makes use of the direct association of the topological components — nodes, edges, faces and solids — with the degrees of freedom to ensure the compatibility and linear independence of the basis functions. The routines used to construct the basis are intuitive and make the method suitable for dealing with complex refinement scenarios in multiple dimensions. This contribution only considers refinement patterns that are driven by a priori geometrical information. The use of error estimators to guide refinement in the scheme is treated in [54]. A recent publication presents a novel *hp*-adaptive strategy for multi-level discretiztations [55]. Figure 2 illustrates the manner in which topological components can be activated and deactivated in order to construct the multi-level *hp*-basis.

A multi-level *hp*-mesh comprises elements with different refinement levels as shown in Figure 2. The letter k is used to denote the refinement level/depth of an element. Elements on the lowest refinement level with $k = 0$ are termed *base elements.* Spatial refinement is performed by superposing a base element with subelements formed by uniform bisections. This procedure is performed recursively until the desired refinement depth, resulting in a refinement tree for every refined base element. The term *leaf elements* is used to refer to the set of all elements in the mesh that do not have subelements. An in-depth description of the properties of the multi-level *hp*-basis and the steps needed for its construction can be found in [46, 56].

3 Hierarchical multigrid for uniform and multi-level hp-grids

The following sections handle the use of multigrid algorithms for the solution of finite cell systems on uniform meshes and multi-level *hp*-refined grids. It begins with a brief summary of the multigrid method, introducing the notation and terminology that is employed in this contribution. It thereafter expounds on how an *hp*-multigrid method can be devised for finite cell systems involving multi-level *hp*-refinement and elaborates on suitable smoothing techniques for achieving convergence rates that are independent of the cut configuration, mesh size and in special cases, even the polynomial order.

3.1 The multigrid algorithm

Multigrid methods aim at improving the convergence of a linear system by incorporating information obtained from a hierarchy of coarse discretizations. Consider a sequence of function spaces $\mathcal{V}_0, \mathcal{V}_1 \ldots \mathcal{V}_{\ell-1}, \mathcal{V}_\ell$, where the index ℓ denotes the level-number and $\ell \in [0, \ell_{max}]$. $\mathcal{V}_{\ell_{max}}$ denotes the finest space which contains the solution of the original problem, whereas $\mathcal{V}_0$ denotes the coarsest space. Each multigrid level is associated with a system of linear equations

$$\mathbf{A}_\ell \mathbf{x}_\ell = \mathbf{b}_\ell. \tag{2}$$

In this work, only sequences of nested spaces are considered, with the consequence that basis functions in a coarse space can be expressed as a linear combination of basis functions from a finer space. This makes it possible to define restriction operators $\mathbf{R}_\ell$ that map basis functions from the fine space of level ℓ into the next coarse space of level $\ell - 1$. Conversely, the prolongation operators $\mathbf{R}_\ell^T$ can be introduced that map a vector in the coarse space $\ell - 1$ to a vector in the fine space ℓ which corresponds to the same function. These operators allow quantities such as vectors and matrices to be transferred from one level to another e.g. $\mathbf{A}_{\ell-1} = \mathbf{R}_\ell \mathbf{A}_\ell \mathbf{R}_\ell^T$ and $\mathbf{r}_{\ell-1} = \mathbf{R}_\ell \mathbf{r}_\ell$, where r_ℓ and $r_{\ell-1}$ are the residuals on level ℓ and $\ell - 1$, respectively.

The two main steps in a multigrid iteration are a smoothing process and the coarse-grid correction. These steps are considered complementary since they act on different components of the error. The error in each multigrid level can be expressed as the difference between the exact solution $\mathbf{x}_\ell$ and the current approximation $\tilde{\mathbf{x}}_\ell$, i.e.

$$\mathbf{e}_\ell = \mathbf{x}_\ell - \tilde{\mathbf{x}}_\ell \tag{3}$$

The smoothing process, or smoothing in short, efficiently reduces the oscillatory components of the error that are associated with large eigenvalues in level ℓ. Smoothing is performed by the repeated application of linear iterative methods such as fixed-point iterations. Standard smoothers include the weighted Jacobi method and the Gauss-Seidel method, but it is also possible to use fixed-point smoothing based on additive Schwarz (AS) techniques, multiplicative Schwarz techniques and incomplete LU factorizations. In this work, the smoothing process is based on additive Schwarz techniques since they can be readily applied in massively parallel settings and their construction can be done in a fully parallel manner without the need for synchronization. This is in contrast to techniques such as Gauss-Seidel and multiplicative Schwarz, which often require coordinated application in parallel settings such as multicoloring algorithms, see e.g. [57]. The application of the smoother is carried out per the formula

$$\tilde{\mathbf{x}}_\ell = \tilde{\mathbf{x}}_\ell + \omega \mathbf{M}_\ell^{-1} \mathbf{r}_\ell, \tag{4}$$

where ω denotes the relaxation parameter and $\mathbf{M}^{-1}$ the smoother. The choice of the relaxation parameter ω for the various FCM meshes applied in this manuscript is elaborated in Section 3.3. The coarse-grid correction accelerates the convergence of the smooth components of the error that are associated with small eigenvalues in level ℓ. By taking advantage of the relation between the residual $\mathbf{r}_\ell$ and the error $\mathbf{e}_\ell$

$$\mathbf{A}_\ell \mathbf{e}_\ell = \mathbf{r}_\ell \tag{5}$$

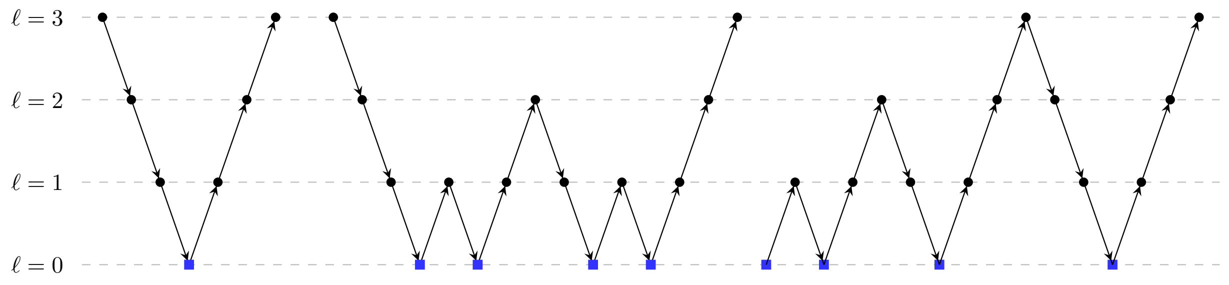
it is possible to approximate the smooth components of the error $\mathbf{e}_\ell$ using a coarser discretization (grid), where $\mathbf{e}_\ell \approx \mathbf{R}_\ell^T \mathbf{e}_{\ell-1}$, by solving the coarse-grid system

$$\mathbf{A}_{\ell-1} \mathbf{e}_{\ell-1} = \mathbf{r}_{\ell-1}. \tag{6}$$

The righthand side vector of this coarse system is formed by restricting the residual at level ℓ to level $\ell - 1$, i.e. $\mathbf{r}_{\ell-1} = \mathbf{R}_\ell \mathbf{r}_\ell$. Once this system has been solved, its solution, termed the *coarse-grid correction*, can be transferred to level ℓ through the expression

$$\tilde{\mathbf{x}}_\ell = \tilde{\mathbf{x}}_\ell + \mathbf{R}_\ell^T \mathbf{e}_{\ell-1}. \tag{7}$$

A single multigrid iteration or cycle consists of the application of the smoothing and coarse-grid correction steps on the different multigrid levels. An exception is, however, made on the coarsest level $\ell = 0$. Here, the coarse system is solved with either a direct solver, if the system is small in size, or with an iterative solver before prolongating its solution to level $\ell = 1$. There are different ways of traversing the levels within a multigrid cycle, the most prevalent of which are the V-cycle, W-cycle and the full multigrid cycle (FMG). A schematic representation of each of these cycles is given in Figure 3. V-cycles are applied in this work and a summary of the multigrid algorithm for this iteration type is given in Algorithm 1.



(a) V-cycle. (b) W-cycle. (c) FMG-cycle.

Figure 3: Illustration of three different types of multigrid iterations. The circular dots represent n_s smoothing steps performed on every level with $\ell \neq 0$, while the square dots represent an "exact" solve performed on $\ell = 0$. Downwards pointing arrows represent restriction operations, while upwards pointing arrows represent prolongations.

Algorithm 1: $\tilde{\mathbf{x}}_\ell = \text{performVcycle}(\tilde{\mathbf{x}}_\ell, \mathbf{r}_\ell, \ell)$

```
if $l \neq 0$ then
    # perform $n_s$ pre-smoothing steps
    for $i \in n_s$ do
        $\tilde{\mathbf{x}}_\ell \leftarrow \tilde{\mathbf{x}}_\ell + \omega \mathbf{M}_\ell^{-1} \mathbf{r}_\ell$
    end
    # update residual
    $\mathbf{r}_\ell = \mathbf{b}_\ell - \mathbf{A}_\ell \tilde{\mathbf{x}}_\ell$
    # coarse grid correction
    $\mathbf{r}_{\ell-1} = \mathbf{R}_\ell \mathbf{r}_\ell$
    $\tilde{\mathbf{x}}_{\ell-1} = \text{performVcycle}(\mathbf{0}, \mathbf{r}_{\ell-1}, \ell-1)$
    $\tilde{\mathbf{x}}_\ell = \tilde{\mathbf{x}}_\ell + \mathbf{R}_\ell^T \tilde{\mathbf{x}}_{\ell-1}$
    $\mathbf{r}_\ell = \mathbf{b}_\ell - \mathbf{A}_\ell \tilde{\mathbf{x}}_\ell$
    # perform $n_s$ post-smoothing steps
    for $i \in n_s$ do
        $\tilde{\mathbf{x}}_\ell \leftarrow \tilde{\mathbf{x}}_\ell + \omega \mathbf{M}_\ell^{-1} \mathbf{r}_\ell$
    end
else
    # solve the coarse system
    $\tilde{\mathbf{x}}_\ell = \text{solve}(\mathbf{A}_\ell, \mathbf{r}_\ell)$
end
```

3.2 Multigrid for hierarchical bases

It is possible to take advantage of the hierarchical nature of the shape functions in the finite cell method and multi-level *hp*-refinement to devise a hierarchical multigrid method. Let $\mathbf{N}_\ell$ denote the basis functions that belong to the multigrid level ℓ. For each level in a *p*-multigrid (for uniform meshes) or *hp*-multigrid (for multi-level *hp*-meshes) the following relation holds

$$\mathbf{N}_0 \subset \mathbf{N}_1 \ldots \mathbf{N}_{\ell-1} \subset \mathbf{N}_\ell. \tag{8}$$

The hierarchical structure in (8) is also reflected in the degree of freedom vector $\mathbf{u}$. The vector of DOFs on level ℓ, denoted by $\mathbf{u}_\ell$, is made up of coefficients from level $\ell - 1$, represented by $\mathbf{u}_{\ell-1}$ and entries solely on level ℓ, denoted by w_ℓ, i.e.

$$\mathbf{u}_\ell = \begin{bmatrix} \mathbf{u}_{\ell-1} \\ \mathbf{w}_\ell \end{bmatrix}. \tag{9}$$

From (8) it follows that the basis functions spanning the (coarse) nested subspace can not only be constructed by a linear combination of basis functions in the fine space, which is done in restriction and prolongation, but are explicitly contained in the basis functions of the fine space. This leads to an elegant and efficient multigrid framework since all restriction and prolongation operators reduce to binary matrices which do not need to be explicitly applied. Transitioning from one multigrid level to another is done easily by either leaving out specific basis functions or re-introducing them back into the system. Equation (10) illustrates how the DOFs on level ℓ can be "trimmed" to obtain the DOFs on level $\ell - 1$. $\mathbf{I}$ represents an identity matrix while the term $\mathbf{0}$ denotes a matrix in which all entries are zero.

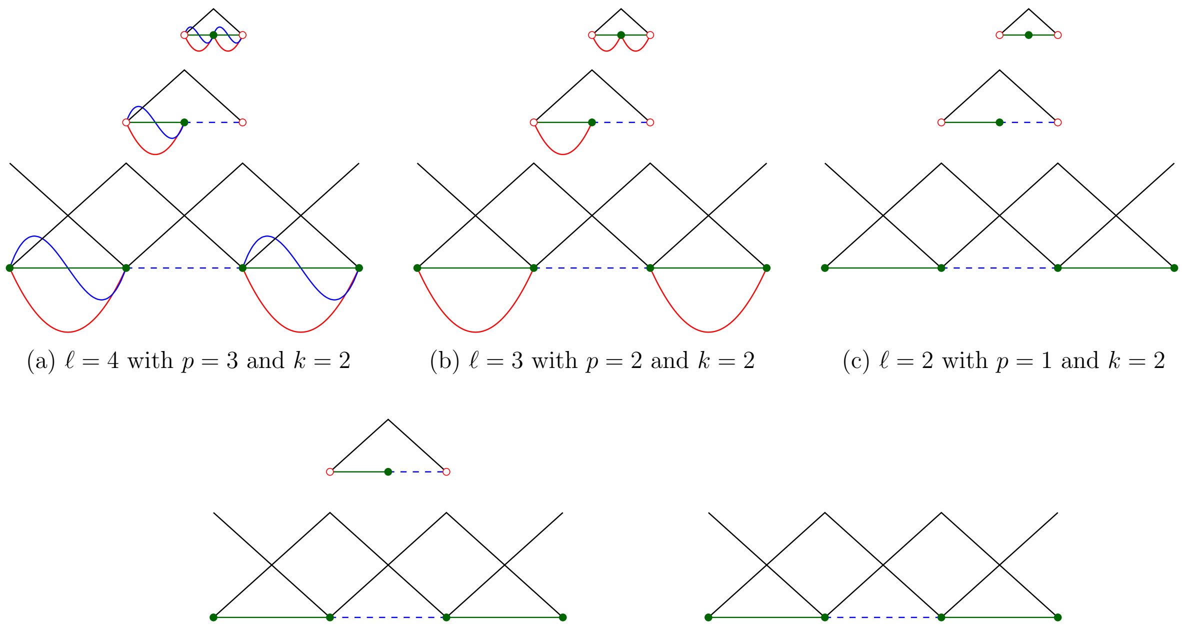
$$\mathbf{u}_{\ell-1} = [\mathbf{I},\ \mathbf{0}] \begin{bmatrix} \mathbf{u}_{\ell-1} \\ \mathbf{w}_{\ell} \end{bmatrix}. \tag{10}$$

For uniform grids with high-order elements, an *arithmetic p-sequence* is used to generate the coarse subspaces, i.e. the value of p is progressively reduced until $p = 1$, see [42]. In the case of multi-level hp-grids, it is required to first reduce the polynomial order p in the overlay meshes, before reducing the levels of refinement. This procedure ensures that the resulting coarse spaces are subspaces.

The hierarchical nature of the FE-basis in FCM and the multi-level hp-scheme is also reflected in the system matrices. Consequently, computation of lower-level matrices is not needed as this information is readily available. Equation (11) shows the structure of a hierarchical matrix of level ℓ, which consists of entries $\tilde{\mathbf{A}}_{\ell}$ belonging to basis functions contained solely in the highest level, a term $\tilde{\mathbf{A}}_{\ell-1}$ that contains entries of all lower levels up to $\ell - 1$ and a term $\tilde{\mathbf{A}}_{\ell,\ell-1}$ that couples DOFs on level ℓ with all other DOFs.

$$\mathbf{A}_{\ell} = \begin{bmatrix} \tilde{\mathbf{A}}_{\ell} & \tilde{\mathbf{A}}_{\ell,\ell-1} \\ \tilde{\mathbf{A}}_{\ell,\ell-1}^{T} & \mathbf{A}_{\ell-1} \end{bmatrix} \tag{11}$$

No distinction will be made from this point on in the manuscript between the p-multigrid scheme for uniform meshes and the hp-multigrid approach of the multi-level hp-discretizations, since the p-multigrid can be regarded as a special case of the hp-multigrid in which $k = 0$. Figure 4 shows the hp-multigrid levels used for a one-dimensional multi-level hp-mesh. A reduction in the p-level is performed first, followed by a reduction in the levels of refinement until the lowest multigrid level with $p = 1$ and $k = 0$ is reached.


(a) $\ell = 4$ with $p = 3$ and $k = 2$ (b) $\ell = 3$ with $p = 2$ and $k = 2$ (c) $\ell = 2$ with $p = 1$ and $k = 2$

(d) $\ell = 1$ with $p = 1$ and $k = 1$ (e) $\ell = 0$ with $p = 1$ and $k = 0$

Figure 4: Multigrid levels in a one-dimensional mesh with two levels of multi-level hp-refinement ($k = 2$) and a polynomial order of $p = 3$.

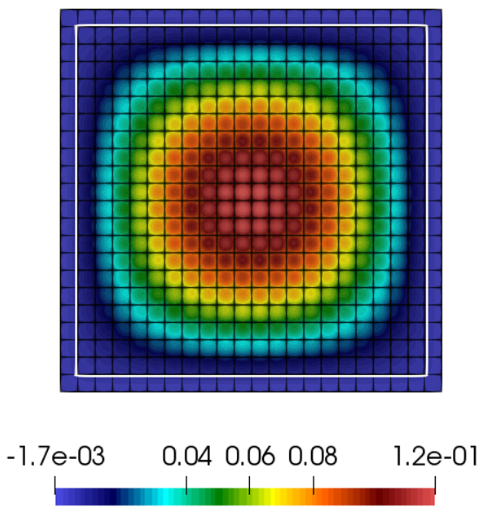
Remark 1 *The $p = 1$ and $k = 0$ level is chosen as the coarsest level in the proposed hp-multigrid approach. This allows a simple yet elegant parallel implementation, since only the DOFs on the finest problem need to be distributed over the MPI tasks. All coarse sub-problems in the multigrid hierarchy are able to reuse the parallel data structures that have already been set up for the fine problem and do* *not require any additional steps such as redistribution or partitioning of the DOFs. It should be noted, however, that the $p = 1$, $k = 0$ problem can be coarsened further using a standard geometric multigrid algorithm. This procedure can improve the convergence of the coarse problem and is beneficial in applications where the solution of the coarse problem is slow.*

3.3 Selection of suitable smoothing strategies

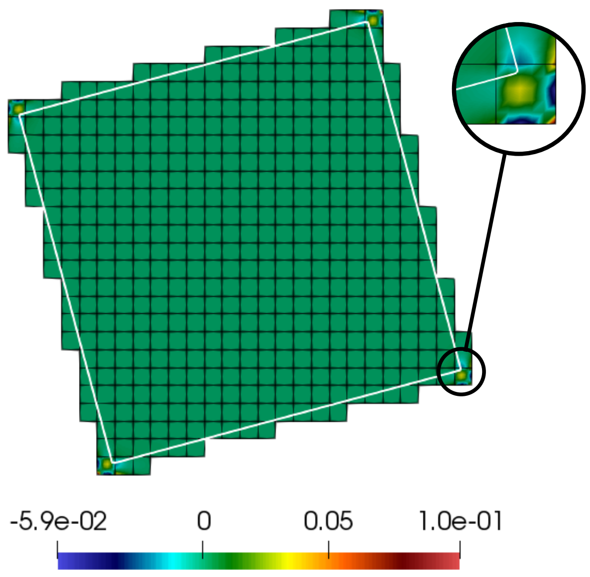
The choice of an appropriate smoother in a multigrid algorithm is fundamental in achieving convergence rates that are independent of the mesh parameter h. In specific problems, it is even possible to achieve convergence rates that are independent of the element polynomial order p when a suitable smoother is chosen, see Section 4.1.1.

3.3.1 Additive Schwarz smoothing

Smoothing techniques such as the Jacobi and Gauss-Seidel methods are commonly used in multigrid algorithms for boundary-conforming finite element methods. These techniques are, however, not suitable for FCM problems as they fail to resolve the conditioning problems associated with cut cells. To illustrate this point, we consider a system arising from the Poisson equation posed on a square domain and compare two scenarios; a case in which the domain is discretized in a boundary-conforming manner and a scenario where an immersed grid is utilized. Figure 5 visualizes the eigenmodes associated with the smallest eigenvalue (further denoted as “smallest eigenmode”) of the system matrix when Jacobi preconditioning is applied. In the boundary-conforming case, the smallest eigenmode corresponds to a smooth mode that spans the entire domain as shown in Figure 5a. This mode can be sufficiently approximated on a coarse space. In the immersed case, the smallest eigenmode is restricted to a few cut elements, see Figure 5b. The occurrence of such small modes in immersed methods impedes the convergence of iterative solvers and makes conventional smoothers such as the Jacobi and Gauss-Seidel methods unsuitable for immersed grids.



(a) Smallest eigenmode of a boundary-conforming system with Jacobi preconditioning.



(b) Smallest eigenmode of an immersed system with Jacobi preconditioning.

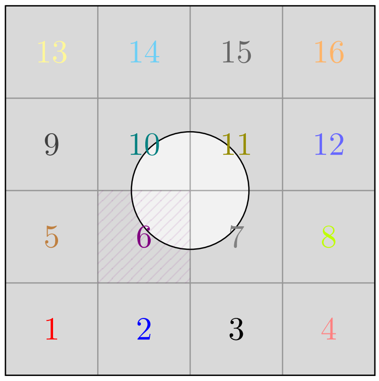
Figure 5: Illustration of the smallest eigenmodes of a Jacobi preconditioned system arising from the Poisson equation on a boundary-conforming grid and an immersed grid.

Reference [36] shows that smoothers based on the additive and multiplicative Schwarz lemmas are better suited for multigrid methods involving cut cells and presents results of multiplicative Schwarz smoothing in different linear elastic examples. As mentioned in the previous section, an additive Schwarz approach is used in this work, since it can be easily applied in parallel and has been successfully used for preconditioning FCM systems involving multi-level *hp*-refinement, see [13]. The core idea behind additive Schwarz smoothing is to group the basis functions in a mesh into *blocks/groups* and thereafter construct a smoother by inverting and summing sub-matrices of $\mathbf{A}$ devised from the chosen blocks as per the expression

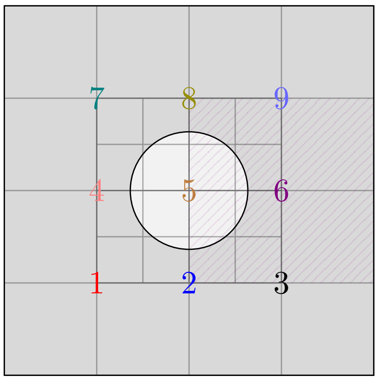
$$\mathbf{M}^{-1} = \sum_{i=1}^{n_{\text{blocks}}} \mathbf{P}_i \underbrace{\left(\mathbf{P}_i^T \mathbf{A} \mathbf{P}_i\right)^{-1}}_{\mathbf{A}_i^{-1}} \mathbf{P}_i^T \,, \tag{12}$$

where n_{blocks} denotes the number of blocks, i the index corresponding to the i^{th} block containing m basis functions such that $\mathbf{A}_i \in \mathbb{R}^{m \times m}$. $\mathbf{P}$ and $\mathbf{P}^T$ are restriction and prolongation operators and $\mathbf{P}_i \in \mathbb{R}^{n_{\text{DOFs}} \times m}$.

The proper selection of the basis function blocks is essential for guaranteeing a robust smoother that can deal with conditioning problems due to cut elements. Following [12, 13] we select the additive Schwarz blocks in such a way that basis functions that can become potentially linear dependent are grouped together. An *elementwise* approach is used for block selection in uniform grids, i.e. all basis functions that are supported on an element are considered to be one group, see Figure 6a. A *patchwise* approach is employed for multi-level *hp*-grids, which forms additive Schwarz blocks comprising all basis functions supported on a group of base elements around a mesh node as indicated in Figure 6b. It should be noted that these two approaches result in overlapping additive Schwarz blocks.



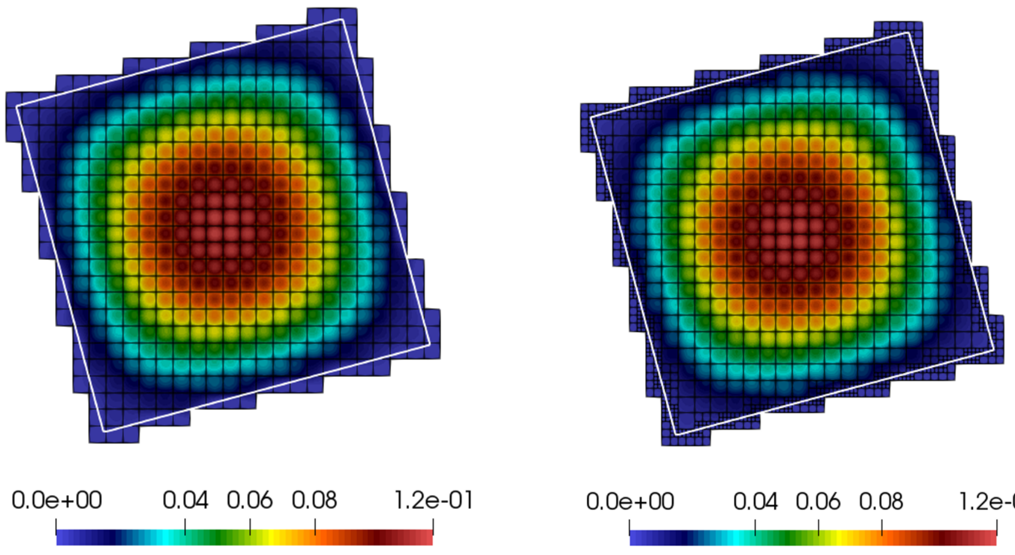
(a) Elementwise blocks



(b) Patchwise blocks

Figure 6: Selection of the additive Schwarz groups for finite cell problems on uniform grids and multi-level *hp*-refined grids. This selection is performed on the granularity of the base elements for uniform grids and node-based element patches for multi-level *hp*-grids. The numbers in the figures indicate the number of additive Schwarz blocks and the support of the 6th block is shaded

We again illustrate the suitability of the proposed smoothers by considering the smallest eigenmodes of a Poisson problem on a square domain that is rotated with respect to a fixed background grid. Figure 7 visualizes the smallest eigenmodes of the systems that are preconditioned using the elementwise and patchwise additive Schwarz techniques on uniform grids and multi-level *hp*-refined grids, respectively. Both approaches are able to adequately deal with small modes due to cut cells



(a) Smallest eigenmode of an immersed system on a uniform grid with elementwise additive Schwarz preconditioning.

(b) Smallest eigenmode of an immersed system on a multi-level *hp*-grid with patchwise additive Schwarz preconditioning.

Figure 7: Illustration of the smallest eigenmodes of a system arising from the Poisson equation on a uniform grid, Figure 7a, and a multi-level *hp*-grid, Figure 7b. The system matrices are preconditioned using the additive Schwarz techniques.

3.3.2 The relaxation parameter ω

Although the additive Schwarz smoother can be easily applied in a massively parallel setting, it requires sufficient stabilization in order to converge. It is well known that the convergence of fixed-point iterations requires the eigenvalues of the iteration matrix to be bounded, i.e.

$$\rho(\mathbf{I} - \omega \mathbf{M}^{-1}\mathbf{A}) < 1. \tag{13}$$

In [36], it is shown that the largest eigenvalue of the matrix $\mathbf{M}^{-1}\mathbf{A}$ is bounded from above by the maximum overlap n_{max} of the additive Schwarz blocks i.e. $\lambda_{\text{max}}(\mathbf{M}^{-1}\mathbf{A}) \leq n_{\text{max}}$. Since the value of n_{max} is 2^d for Cartesian grids, the relaxation parameter ω can be chosen as $\omega = 2/n_{\text{max}}$, thus guaranteeing stability of the fixed-point iteration. Note, however, that the relation between the largest eigenvalue and n_{max} is an inequality and that it is possible to choose values for ω that are higher than $2/n_{\text{max}}$. For the integrated Legendre basis functions considered in this work, the best performance was achieved by choosing $\omega = 1/3$ for two-dimensional problems and $\omega = 2/15$ for three-dimensional problems, when using an additive Schwarz smoother based on elementwise blocks. When the additive Schwarz blocks are chosen in a patchwise manner, n_{max} is larger, and the best convergence rates are achieved using $\omega = 1/6$ and $\omega = 1/18$ in two and three dimensions respectively. The values of ω suggested in this work were determined in a heuristic approach and achieve the mesh-independent convergence rates for different geometries as shown in Section 4.

It should be noted that different smoothing strategies can be applied on the different multigrid levels, e.g. additive Schwarz on the finest level and Gauss-Seidel or Jacobi smoothing on lower levels. This is, however, not investigated in this work and additive Schwarz smoothing is applied on all levels with $\ell \neq 0$.

3.3.3 Implementational aspects

The multigrid scheme proposed in this paper is implemented in an in-house finite element code written in `C++`. The code's modular design enables it to be used on a variety of computing platforms ranging from desktop computers to massively parallel systems. It supports hybrid parallel simulations through the use of MPI [58] and OpenMP[59]. Generation of the computational mesh in large-scale computations is carried out in a distributed manner using a parallel adaptive Cartesian grid. This strategy ensures that the memory resources are utilized in a scalable manner, since no single MPI task needs to know the complete extent of the computational domain. The code framework utilizes the functionality of the `Epetra` package in `Trilinos` [60] for the parallel construction and storage of the system matrices. `Epetra` is also used to perform parallel linear algebra operations. The system matrices of the different multigrid levels are cached during the solver setup phase to increase the efficiency of the overall solution process. The storage cost of these matrices is minimal and scales inversely in proportion to the number of MPI tasks.

Computational costs

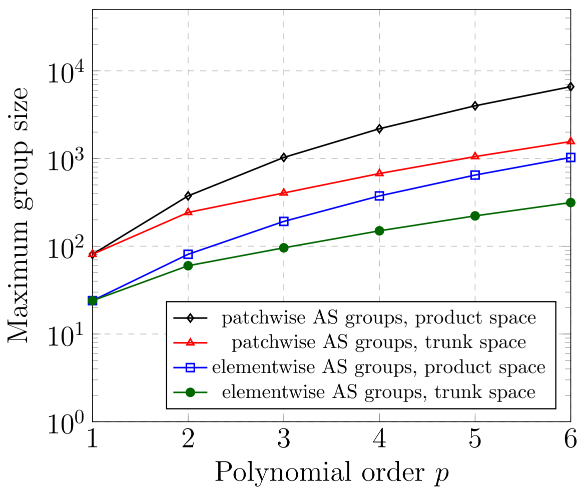
The two main procedures that influence the overall computational cost of the proposed multigrid solution scheme are: *i)* constructing and applying the additive Schwarz smoothers and *ii)* solving the coarse problem.
Construction of the AS smoothers following (12) entails the inversion of sub-matrices derived from basis function groups. The cost of this operation is dictated by the number and size of these sub-matrices. The number of sub-matrices increases linearly with the number of elements. The maximum size of a sub-matrix formed from the additive Schwarz groups, denoted by $m_{\max}$, is proportional to the polynomial order p, the spatial dimension d, the manner in which the AS groups are selected (elementwise or patchwise selection) and the number of unknown field variables denoted by n_f. For a Poisson problem $n_f = 1$ while $n_f = 3$ for a three-dimensional linear elastic problem. Since the number of DOFs associated with the topological components for the tensor product and trunk space elements is known [53], it is possible to compute an upper bound for $m_{\max}$ for uniform grids, see Table 1 and Figure 8a.

tensor product space	**trunk space**
$n_f(np+1)^3$	$n_f(n+1)^2(3np-2n+1)$ for $p<4$
	$0.5n_f(n+1)(3n^2p^2-9n^2p+6np+14n^2-2n+2)$ for $4 \leq p \leq 5$
	$n_f(n^3p^3-3n^3p^2+9n^2p^2+20n^3p-9n^2p+18(np-n^3+2n^2)+6)$ for $p \geq 6$

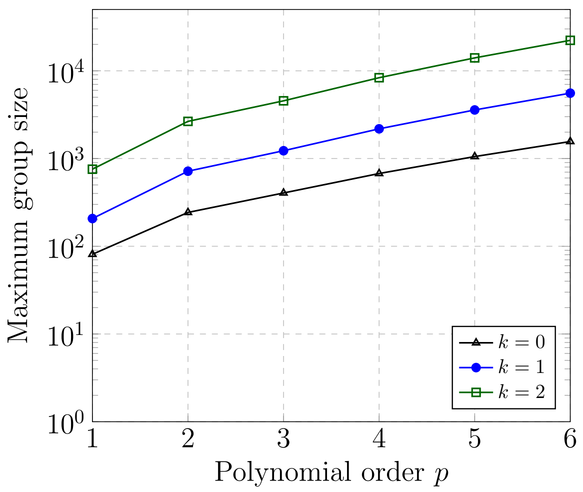
Table 1: Maximum size of the additive Schwarz groups, $m_{\max}$, in a uniform three-dimensional grid. p represents the element polynomial order, n_f the number of field variables in the problem and n is a factor that is equal to one for elementwise blocks and equal to two when patchwise block selection is applied.

In the case of multi-level hp-grids, $m_{\max}$ is not bounded and its size depends on the refinement level k and the refinement pattern applied to the mesh elements. Figure 8b shows the value of $m_{\max}$ for the benchmark problem considered in Section 4.3. From Table 1 and Figure 8 it is clear that the inversion of the patchwise additive Schwarz sub-matrices can become increasingly expensive for high polynomial orders and refinement levels. It is therefore important that optimized algorithms are used to perform these inversions. Our code framework makes use of distributed and shared memory parallelism to accelerate the construction of the additive Schwarz smoothers. For "small" sub-matrices, where $\mathbf{A}_i \in \mathbb{R}^{m \times m}$ and $m < 1000$, the built-in invert function of the `BOOST` library is used for the inversion. When m exceeds 1000, the direct solver `Pardiso` [61] is used to invert

$\mathbf{A}_i$ by solving an equation system with m righthand side vectors. This operation can be written as $\mathbf{A}_i\mathbf{X} = \mathbf{B}$, where $\mathbf{B}$ is a $m \times m$ matrix whose columns are made up of the unit vectors $\mathbf{e}_j$ with $j \in [0, m]$.



(a) Maximum AS group size for uniform grids.



(b) Maximum group size for the benchmark example involving multi-level hp-refinement in Section 4.3.

Figure 8: Illustration of the maximum size of the basis function groups used to construct the additive Schwarz smoothers for three-dimensional linear elastic problems.

The use of the `Epetra` package for the storage of the additive Schwarz smoothers $\mathbf{M}_\ell^{-1}$ allows our numerical code to make use of its optimized parallel multiplication kernels. This ensures that the smoothers can be applied in an efficient and scalable manner. We again leverage the `AztecOO` package in `Trilinos` for the solution of the coarse systems with $p = 1$ and $k = 0$. These systems are solved using the package's parallel CG solver and the additive Schwarz preconditioner published in [13].

4 Numerical examples

The following section investigates the performance of the proposed hierarchical multigrid framework in a series of numerical examples. We use the multigrid V-cycle algorithm primarily as a preconditioner within the CG method but also show results in which the algorithm is utilized as a stand-alone solver. The quality of the solution obtained in the different examples is measured by monitoring the norm of the relative residual defined as

$$\frac{\|\mathbf{r}_i\|_2}{\|\mathbf{b}\|_2} = \frac{\|\mathbf{b} - \mathbf{A}\mathbf{x}_i\|_2}{\|\mathbf{b}\|_2} \tag{14}$$

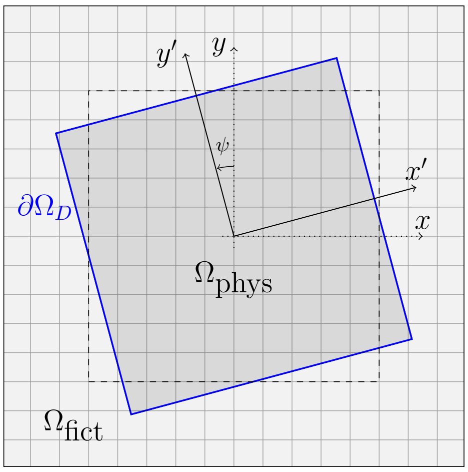
with the righthand side vector $\mathbf{b}$ and the terms $\mathbf{r}_i$ and $\mathbf{x_i}$ that denote the residual and approximation of the finest grid in the i^{th} iteration, respectively. When the multigrid algorithm is used as a solver, the contraction number ρ_i, defined as the quotient between two consecutive residual norms i.e.,

$$\rho_i = \frac{\|\mathbf{r}_i\|_2}{\|\mathbf{r}_{i-1}\|_2}, \tag{15}$$

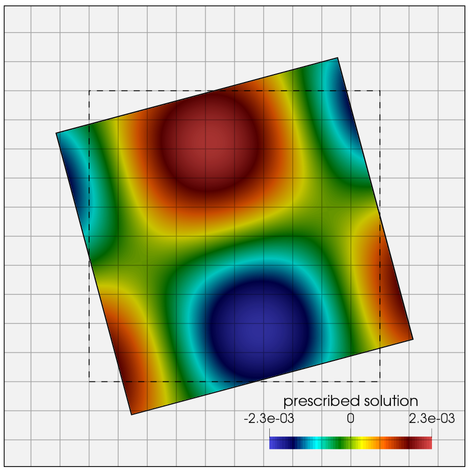
is also used as a measure to judge the effectiveness of the solution scheme. ρ_{max} is a measure that denotes the maximum contraction number (largest reduction factor) in a series of iterations. Only symmetric positive definite systems arising from problems in linear elasticity or the Poisson equation are considered in this section. A total of five pre- and post-smoothing steps are applied on every multigrid level $\ell \neq 0$ in each simulation. All computations are performed on the SuperMUC-NG system hosted at the Leibniz Supercomputing Center in Garching, Germany. The compute nodes used have an architecture comprising dual-socket Intel Xeon Scalable Platinum 8168 processors (Skylake). Each node has a total of 48 cores and 96GB of main memory. Furthermore, the following compiler and library versions are used: IntelMPI compiler version 19.0, Trilinos 12.12.1 and the GNU compiler 7.0. The most aggressive level of compiler optimization flags are chosen for the nodal architecture and include -O3 and -funroll-loops among others.

4.1 Poisson problem on a square domain

The first numerical example investigates the performance of the proposed *hp*-multigrid algorithm as a stand-alone solver and as a preconditioner within a CG scheme. It aims to demonstrate the effectiveness of the additive Schwarz technique when used as a smoother for finite cell problems on uniform meshes. We consider a Poisson problem posed on a square domain of unit length in which the physical domain is rotated about the origin with respect to a fixed background grid as shown in Figure 9. This setup results in different cut scenarios when the angle of rotation ψ is varied. The method of manufactured solutions is applied in this numerical test and a temperature field $\bar{u}$ is chosen such that



(a) Geometry of the rotating square domain.



(b) Prescribed solution.

Figure 9: Rotating Poisson problem with a manufactured solution.

$$\bar{u} = \frac{1}{2\kappa a^2} \cos(a\, x') \sin(a\, y'), \tag{16}$$

with $a = \frac{3}{2}\pi$ and x', y' denoting the coordinates of the rotated coordinate system defined at the center of the domain. A source term s is derived from the manufactured solution following the Poisson equation $-\kappa \Delta u = s$ with $s = \cos(a\, x') \sin(a\, y')$ and applied in Ω_{phys}. Dirichlet boundary conditions are prescribed on all edges of the square domain such that $u = \bar{u}$ on Γ_D. These constraints are enforced using the penalty method with $\beta = 10^4$. A value of $\alpha = 10^{-8}$ is applied in Ω_{fict} and a value of $\kappa = 10$ is used in Ω_{phys}.

4.1.1 Convergence behavior in the mesh-fitting case

We first analyze the performance of the hierarchical multigrid scheme in the mesh-fitting case where $\psi = 0$. We consider uniform grids using high-order elements with different polynomial orders with $p \in [2, \ldots, 5]$ and mesh sizes $h = \{\frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}\}$. Moreover, the effectiveness of four different smoothers is investigated: *i)* Jacobi smoothing, *ii)* Gauss-Seidel smoothing, *iii)* additive Schwarz smoothing with elementwise blocks and *iv)* additive Schwarz smoothing with patchwise blocks. In the study at hand, the convergence behavior of the hierarchical multigrid approach as a stand-alone solver and a preconditioner is analyzed for the different combinations of smoother, polynomial orders and mesh sizes. Each solver is terminated when the value of the relative residual is below 10^{-9} or when the number of iterations exceeds 500. The $*$ symbol is used to represent scenarios in which the solver did not converge within 500 iterations. The results of this study are summarized in Table 2.

Table 2 shows the convergence behavior of the multigrid algorithm as a solver and preconditioner for $\psi = 0°$. In this boundary-conforming example, all smoothers achieve convergence rates that are independent of the mesh size h. These results indicate the correct implementation of the multigrid scheme. These convergence rates are, however, dependent on the polynomial order for the Jacobi, Gauss-Seidel and elementwise additive Schwarz smoothers as shown in Table 2 and indicated by the maximum contraction of the residual for a mesh with $h = \frac{1}{32}$ recorded in Table 3. These smoothers appear to perform better for odd polynomial orders than for even orders. A similar odd-even pattern is reported in [24] and may be due to the asymmetry of the manufactured solution. Convergence rates independent of the polynomial order p are obtained using the patchwise additive Schwarz smoother.

4.1.2 Convergence behavior in immersed configurations

The multigrid algorithm is now used in an immersed setting, in which the domain Ω_{phys} is rotated about the origin as shown in Figure 9a. In analogy to Section 4.1, we compare the performance of different smoothers for $\psi = 30°$ and summarize the results in Table 4. The effect of the ill-conditioning due to the cut cells is clearly seen in this example as the standard smoothers, i.e. the Jacobi and Gauss-Seidel methods, fail to improve the conditioning of the linear systems and are characterized by poor convergence of the multigrid solution techniques applied in this study. The additive Schwarz smoothers do a better job of detecting almost linear dependent functions as shown in Table 4. Note that the elementwise smoother may contain a few small modes that cause slow convergence when multigrid is utilized as a stand-alone solver. These modes, however, only result in a few additional CG iterations, when the multigrid algorithm is used as a preconditioner. The patchwise smoother robustly deals with all small modes due to cut cells and results in a convergence behavior that is independent of h and p. Table 5 presents the maximum contraction number for different polynomial orders of a mesh with $h = \frac{1}{32}$.

Rotating Poisson problem: Convergence study for $\psi = 0^\circ$

	Multigrid as a solver			
	Jacobi smoother (5,5)			
h \ p	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	23	25	25	24
3	17	16	15	15
4	31	33	35	37
5	24	24	26	27

	CG with multigrid preconditioner			
	Jacobi smoother (5,5)			
h \ p	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	10	10	10	10
3	8	8	8	8
4	13	13	13	13
5	10	11	11	11

	Multigrid as a solver			
	Gauss-Seidel smoother (5,5)			
h \ p	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	7	7	7	7
3	6	6	6	6
4	13	13	14	14
5	10	10	10	10

	CG with multigrid preconditioner			
	symm. Gauss-Seidel smoother (5,5)			
h \ p	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	6	6	6	6
3	5	5	5	5
4	8	8	8	8
5	8	7	7	7

	Multigrid as a solver			
	elementwise AS smoother (5,5)			
h \ p	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	12	13	13	13
3	10	10	10	9
4	17	17	17	17
5	14	12	13	13

	CG with multigrid preconditioner			
	elementwise AS smoother (5,5)			
h \ p	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	7	8	8	8
3	7	7	7	7
4	9	9	9	9
5	8	8	8	8

	Multigrid as a solver			
	patchwise AS smoother (5,5)			
h \ p	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	8	8	8	7
3	7	6	6	6
4	7	7	7	6
5	6	6	5	5

	CG with multigrid preconditioner			
	patchwise AS smoother (5,5)			
h \ p	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	6	6	6	6
3	6	5	5	5
4	6	6	6	6
5	5	5	5	5

Table 2: Convergence study for $\psi = 0$ comparing the performance of different smoothers for varying element sizes and polynomial orders. The figures in the tables represent the number of iterations required to reach a tolerance in the relative residual of 10^{-9}. Five pre- and post-smoothing steps are performed per V-cycle on each multigrid level with $\ell \neq 0$.

p \ Smoother	2	3	4	5
Jacobi	.478	.391	.683	.617
Gauss-Seidel	.125	.095	.362	.255
Elementwise AS	.309	.252	.430	.414
Patchwise AS	.162	.164	.140	.162

Table 3: Maximum contraction number $\rho_{\max}$ of the p-multigrid solver for $h = \frac{1}{32}$ and $\psi = 0^\circ$.

Rotating Poisson problem: Convergence study for $\psi = 30^\circ$

	Multigrid as a solver			
	Jacobi smoother (5,5)			
p \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	*	*	*	*
3	*	*	*	*
4	*	*	*	*
5	*	*	*	*

	CG with multigrid preconditioner			
	Jacobi smoother (5,5)			
p \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	133	162	201	180
3	*	*	*	*
4	*	*	*	*
5	*	*	*	*

	Multigrid as a solver			
	Gauss-Seidel smoother (5,5)			
p \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	*	*	*	*
3	*	*	*	*
4	*	*	*	*
5	*	*	*	*

	CG with multigrid preconditioner			
	symm. Gauss-Seidel smoother (5,5)			
p \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	8	139	240	279
3	*	*	*	*
4	*	*	*	*
5	*	*	*	*

	Multigrid as a solver			
	elementwise AS smoother (5,5)			
p \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	19	15	13	12
3	19	14	12	11
4	20	18	17	17
5	20	16	15	15

	CG with multigrid preconditioner			
	elementwise AS smoother (5,5)			
p \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	9	9	9	8
3	11	10	9	8
4	11	11	10	10
5	12	10	9	9

	Multigrid as a solver			
	patchwise AS smoother (5,5)			
p \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	8	8	7	6
3	6	7	6	5
4	5	6	5	5
5	5	5	5	4

	CG with multigrid preconditioner			
	patchwise AS smoother (5,5)			
p \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	7	6	6	6
3	6	6	6	5
4	5	6	5	5
5	5	5	5	5

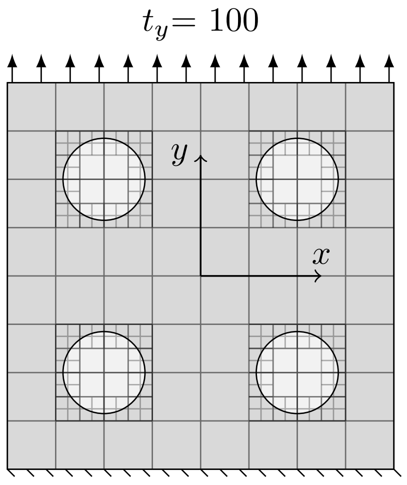
Table 4: Convergence study for $\psi = 30^\circ$ comparing the performance of different smoothers for varying element sizes and polynomial orders. The figures in the tables represent the number of iterations required by the solver while the symbol * indicates that the solver did not converge to a tolerance of 10^{-9} within 500 iterations. Five pre- and post-smoothing steps are performed per V-cycle on each multigrid level with $\ell \neq 0$

Smoother \ p	2	3	4	5
Jacobi	.998	.998	.998	.998
Gauss-Seidel	.996	div.	div.	div.
Elementwise AS	.356	.355	.521	.484
Patchwise AS	.142	.117	.100	.108

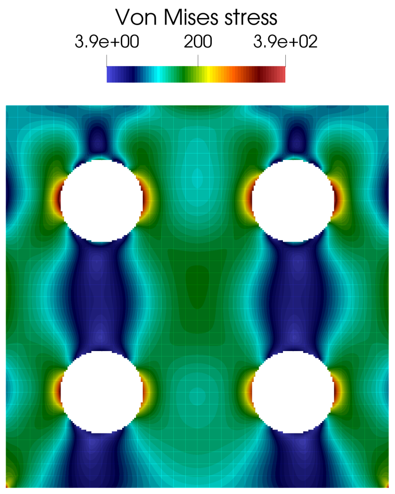
Table 5: Maximum contraction number $\rho_{\max}$ of the p-multigrid solver for $h = \frac{1}{32}$ and $\psi = 30^\circ$. The symbol div. indicates that the solver diverged.

4.2 Perforated linear elastic plate

The previous numerical example established the suitability of using a multigrid preconditioner in conjunction with additive Schwarz smoothers for FCM problems arising from the Poisson equation. The performance of this preconditioner is now investigated in the context of linear elasticity. To this end, a square domain with a length of $l = 4$ is subjected to a tensional force on one end and fully clamped on the other end as depicted in Figure 10a. The square domain has four circular cavities with a radius of $0.3\sqrt{2}$ and is characterized by an elastic modulus $E = 2.069 \cdot 10^5$ MPa and a Poisson's ratio $\nu = 0.29$. The finite cell method is applied in this example with $\alpha = 10^{-8}$ in Ω_{fict} and a penalty parameter $\beta = 10^8$. The number of elements per direction is chosen such that $h \in \{\frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}\}$.



(a) Perforated linear elastic plate.



(b) Von Mises stress.

Figure 10: Setup and von Mises stress of the linear elastic perforated plate example.

Using the setup shown in Figure 10a, numerical studies are carried out that investigate the convergence of a CG solver when the presented additive Schwarz techniques are employed as preconditioners or as smoothers within an *hp*-multigrid preconditioner. The study is conducted on multi-level *hp*-refined grids with a fixed polynomial order and varying levels of refinement. Elements that are intersected by the immersed boundary are refined recursively to a predefined depth as shown in Figure 10a.

Table 6 shows the number of iterations required by the different preconditioner and smoother configurations for multilevel *hp*-grids with quadratic elements and varying levels of refinement. When the elementwise additive Schwarz blocks are utilized as both smoothers and preconditioners, increasing the refinement depth leads to an increase in the number of iterations. The multigrid algorithm that uses these elementwise blocks for smoothing does not achieve convergence rates that are independent of the mesh parameter h. This behavior is attributed to the fact that certain small modes can remain untreated by the elementwise blocks leading to slow convergence. In contrast, the patchwise blocks yield convergence rates that are independent of the refinement depth employed. Furthermore, the multigrid algorithm that employs a patchwise smoothing approach achieves mesh-independent convergence rates.

	CG with elementwise AS preconditioner			
k \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
0	60	76	135	262
1	81	220	234	1192
2	183	367	807	2114
3	294	483	1803	2566

	CG with multigrid preconditioner elementwise AS smoother (5,5)			
k \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
0	12	9	9	9
1	15	30	29	79
2	31	46	63	115
3	50	67	102	114

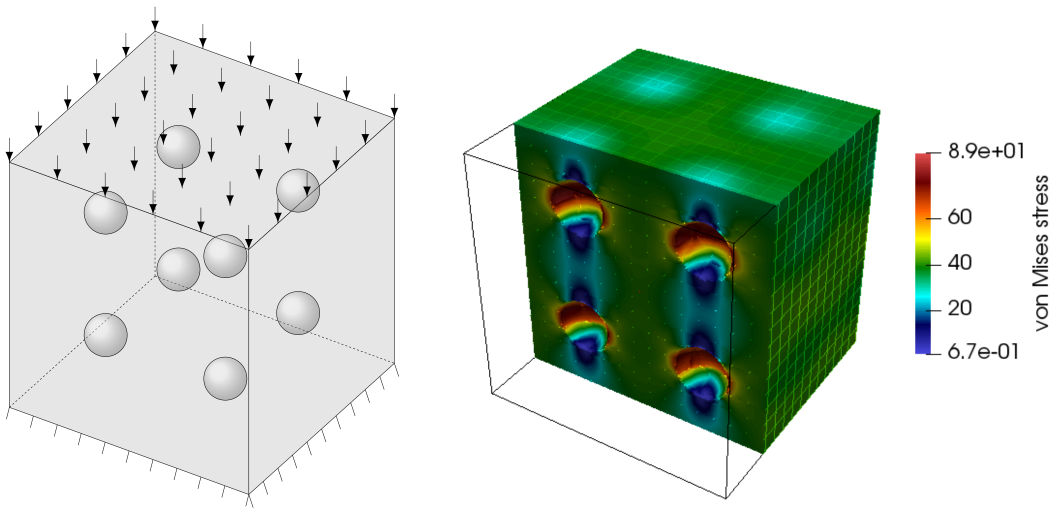
	CG with patchwise AS preconditioner			
k \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
0	43	64	109	210
1	43	64	109	210
2	43	65	109	210
3	43	64	109	210

	CG with multigrid preconditioner patchwise AS smoother (5,5)			
k \ h	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$
2	7	7	7	7
3	7	7	7	7
4	7	7	6	6
5	6	6	6	6

Table 6: Perforated plate example: Convergence behavior of a CG solver for four levels of refinement and four different preconditioners. The study considers multi-level *hp*-refined finite cell meshes with varying resolutions and a fixed polynomial order of $p = 2$.

4.3 Cube with spherical cavities

To assess the performance of the proposed *hp*-multigrid approach in a three-dimensional setting, we now consider a simple example consisting of a linear elastic cube of unit length with spherical cavities subject to compressional loading. The cube has the same material properties as the perforated plate in the previous example and the values of $\alpha = 10^{-8}$ and $\beta = 10^{10}$ are chosen. The radii of the cavities are again $r = 0.3\sqrt{2}$. A homogeneous pressure load $P = 100$ N/mm^2 is applied on the upper surface of the cube as shown in Figure 11a.

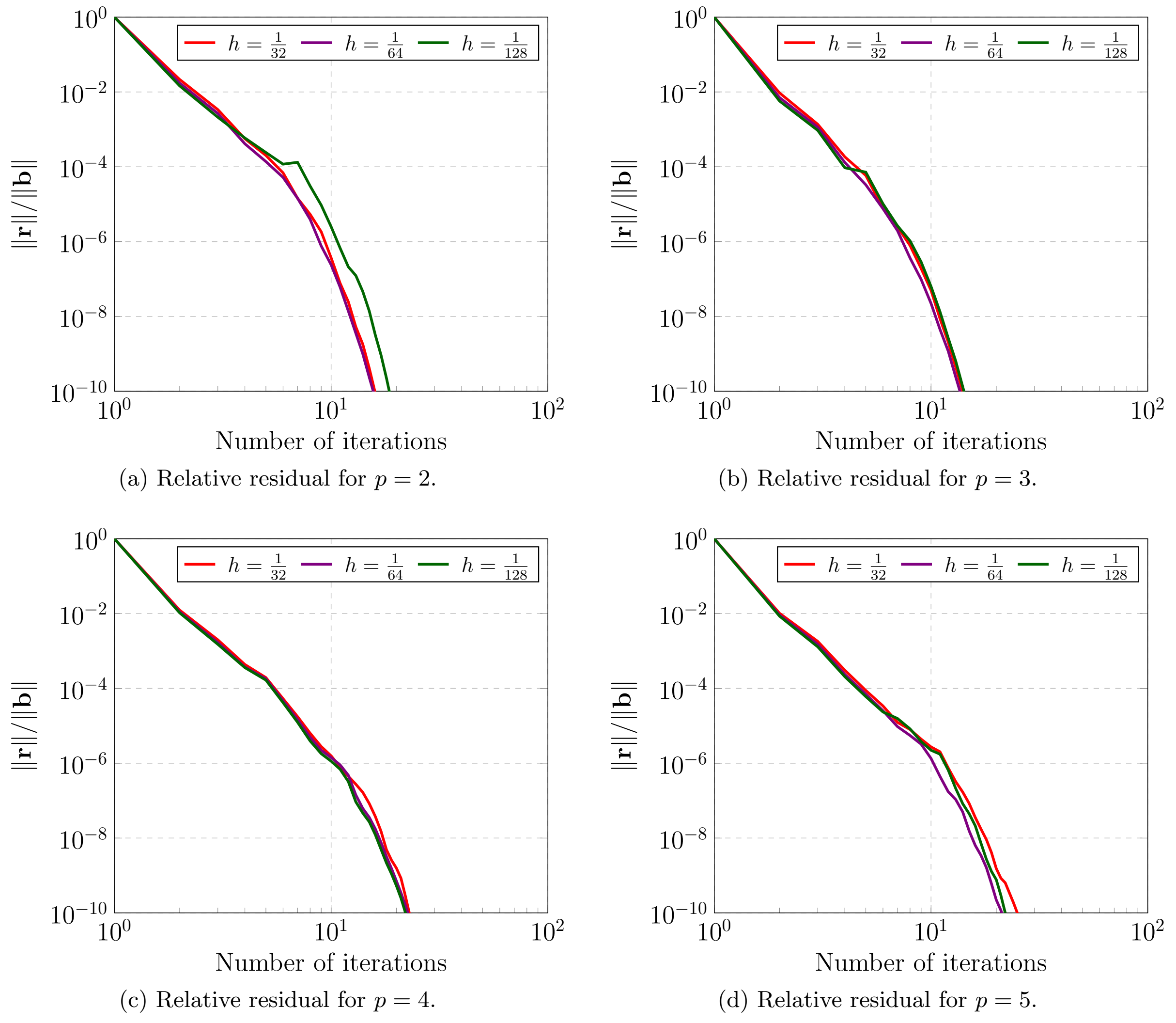


(a) Cube geometry.   (b) Von Mises stress.

Figure 11: Cube with spherical cavities example.

Influence of the polynomial order

The effect of the element polynomial order on the convergence of a CG solver with a p-multigrid preconditioner utilizing elementwise additive Schwarz smoothing is the subject of the following study. To this end, a sequence of meshes with varying element sizes is analyzed. The mesh size is chosen such that $h = \{\frac{1}{32}, \frac{1}{64}, \frac{1}{128}\}$. The number of unknowns ranges from 417 339 for the coarsest $p = 2$ mesh to 135 864 459 for the finest mesh with $p = 5$. Similar to the two-dimensional benchmark cases, the proposed p-multigrid approach leads to convergence rates that are independent of the mesh size h as shown in Figure 12. The number of iterations ranges between 16 and 19 for $p = 2$, between 14 and 15 for $p = 3$, between 22 and 23 for $p = 4$ and between 22 and 26 for $p = 5$.



(a) Relative residual for $p = 2$.

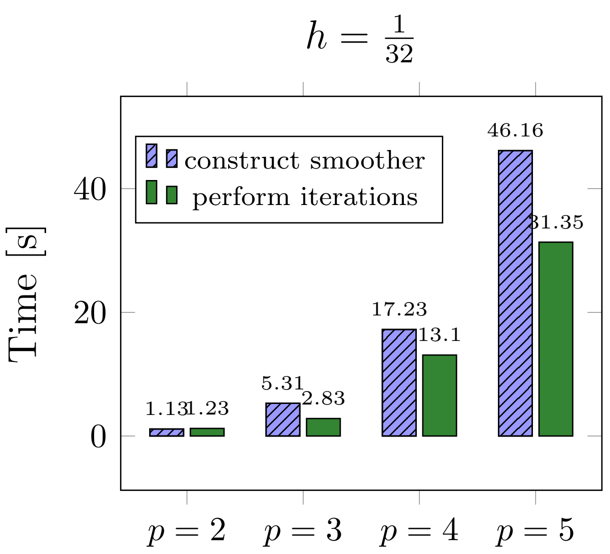
(b) Relative residual for $p = 3$.

(c) Relative residual for $p = 4$.

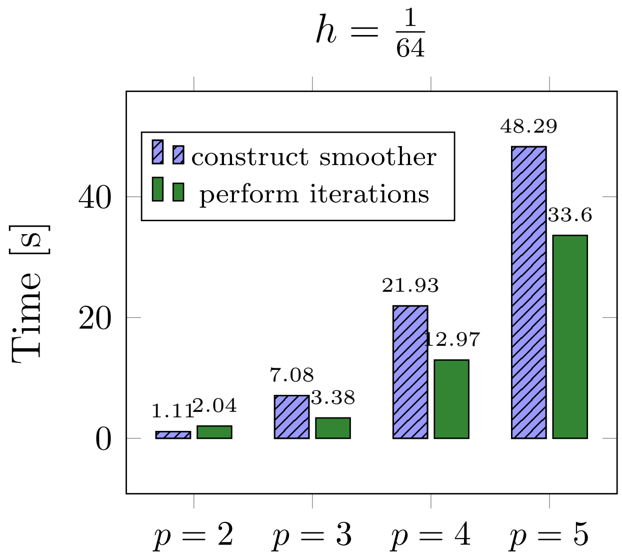
(d) Relative residual for $p = 5$.

Figure 12: Convergence of the CG solver with a p-multigrid preconditioner and elementwise additive Schwarz smoothing for hexahedral trunk space elements.

Figure 13 shows the computational cost of the CG solver with a p-multigrid focusing on the time spent to construct the smoother and to perform the CG iterations. The latter procedure includes the application of the smoother and the solution of the coarse problem. From Figures 13a and 13b one can see that the computational cost of the solver increases for higher polynomial orders.



(a) CPU timings for $h = \frac{1}{32}$. All simulations are run on 96 cores.

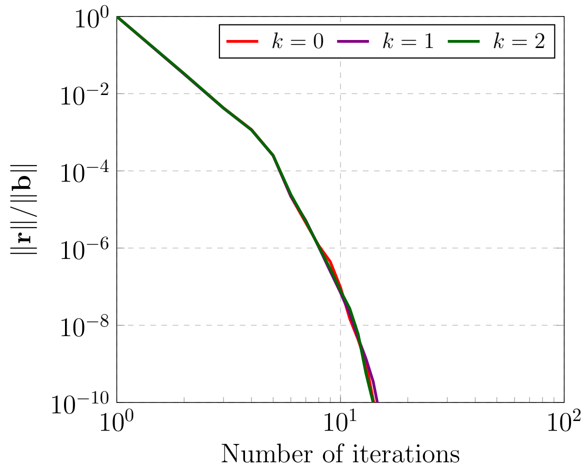


(b) CPU timings for $h = \frac{1}{64}$. All simulations are run on 768 cores.

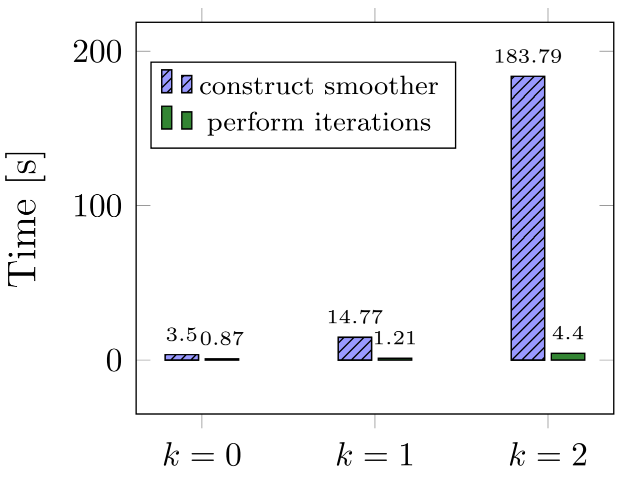
Figure 13: Execution time for the CG solver with a p-multigrid preconditioner.

Influence of the refinement level

The effect of the refinement level on the performance of the hp-multigrid preconditioner is now studied. Starting from an initial grid of 32^3 elements with $p = 2$, the mesh is refined in two steps towards the spherical cavities yielding a total of $4.1 \cdot 10^5$, $5.2 \cdot 10^5$ and $8.9 \cdot 10^5$ DOFs for $k = 0$, $k = 1$ and $k = 2$, respectively. The patchwise additive Schwarz groups are used to construct the smoothers that are applied on every multigrid level $\ell \neq 0$ in analogy to the two-dimensional examples considered in the previous studies. This approach yields convergence rates that are independent of the refinement level as shown in Figure 14a.



(a) Convergence behavior.



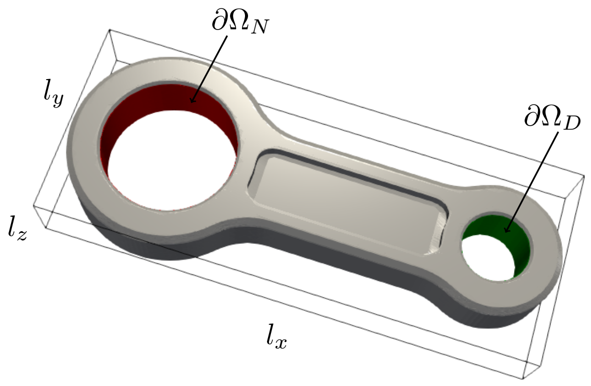
(b) Computation cost of the hierarchical multigrid solver for a set of simulations running on 384 cores.

Figure 14: Performance of the CG solver with an hp-multigrid preconditioner and patchwise additive Schwarz smoothing for a three-dimensional benchmark involving multi-level hp-refinement.

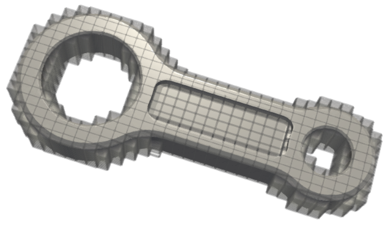
The computational cost of the solver for the different refinement levels considered is shown in Figure 14b. The results shown are obtained in hybrid simulations on 394 cores that are partitioned into 32 MPI tasks each with 6 OMP threads. As expected, the inversion of the sub-matrices for the construction of the patchwise AS smoothers is the most time-consuming operation. This procedure is performed using the sparse direct solver `Pardiso`, since it outperforms the built-in invert function of the `BOOST` library [62] for large matrices. The time spent setting up the patchwise smoothers can be further reduced by the use of a more optimized inversion algorithm. Moreover, it is possible to conceive different strategies for the grouping of basis functions that yield smaller group sizes and therefore lower computation times.

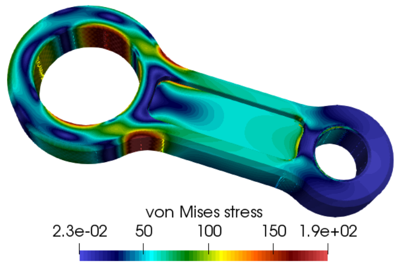
4.3.1 Loading of an aluminum rod

The next example considers the loading of an aluminum rod with an elastic modulus E = 70 GPa and a Poisson's ratio $\nu = 0.3$. Figure 15a shows the rod's geometry with $l_x = 150$ mm, $l_y = 60$ mm and $l_z = 20$ mm. The cylindrical surfaces labeled Ω_D are fixed using the penalty method with a penalty parameter $\beta = 10^6$. A surface load $t_x = 10\,\mathrm{N/mm}^2$ is applied on the surfaces labeled Ω_N in the direction of the rod's shaft. The resultant force acting on the rod is $F_x = 2\pi rh \cdot t_x = 2\pi \cdot 20 \cdot 10 \approx 12.566\,\mathrm{kN}$. Three finite cell discretizations with $h \in \{2, 1, 0.5\}$ are considered and the polynomial degree of the grids is chosen as $p \in \{2, 4\}$. Table 7 summarizes the number of DOFs in each grid. An octree scheme with a depth of 3 is used for the integration of the element matrices and the righthand side.



(a) Rod geometry and boundary conditions


(b) Finite cell mesh.



(c) Von Mises stress

Figure 15: Loading of an aluminum connecting rod.

		DOFs		
h	mesh resolution	$p = 2$	$p = 3$	$p = 4$
2	$75 \times 30 \times 10$	156 594	271 596	492 285
1	$150 \times 60 \times 20$	1 066 059	1 861 425	3 410 673
0.5	$300 \times 120 \times 40$	7 880 868	13 755 963	25 365 804

Table 7: Summary of the number of DOFs for the different discretizations considered in the aluminum rod example.

	CG with elementwise AS preconditioner		
p \ h	2	1	0.5
2	766	1502	2966
3	770	1508	2973
4	835	1593	3254

	CG with multigrid preconditioner elementwise AS smoother (5,5)		
p \ h	2	1	0.5
2	16	15	15
3	14	13	13
4	18	17	17

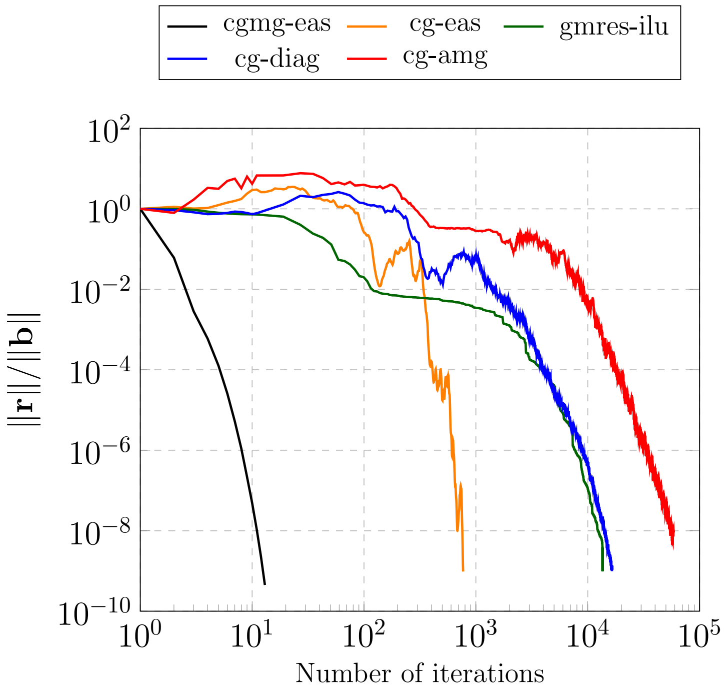
Table 8: Aluminium rod example: Convergence of a CG solver with two different preconditioners; *i)* elementwise additive Schwarz preconditioner and *ii)* p-multigrid preconditioner with elementwise additive Schwarz smoothing. The considered grids consist of trunk space hexahedral elements with different polynomial orders.

Comparison to different iterative solvers

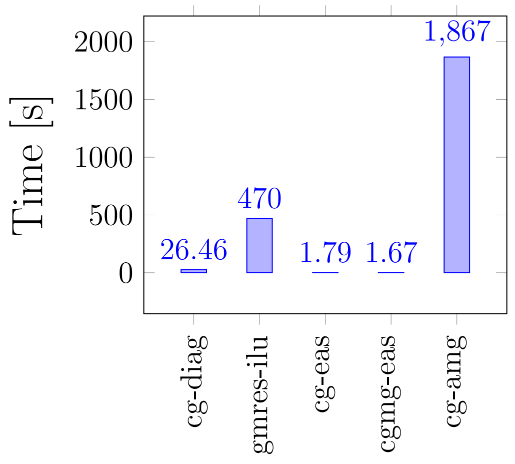
The study at hand compares the performance of the proposed multigrid solution scheme to alternative iterative approaches. In the first part of this study, the convergence behavior of the multigrid scheme with elementwise AS smoothing is compared to a CG solver that uses the additive Schwarz technique as a preconditioner. The results for the different finite cell discretizations considered in this example are presented in Table 8. Both preconditioners yield the expected convergence rates i.e. the elementwise additive Schwarz preconditioner yields rates proportional to h^{-1}, while the use of the p-multigrid preconditioner leads to convergence rates independent of the mesh size. Moreover, there is only a minimal difference in the convergence behavior of different polynomial orders for all preconditioners.

Next, we compare the convergence behavior and execution time of the presented multigrid approach to different solvers and preconditioners available in the `AztecOO` package of `Trilinos`. The solvers applied include: *i)* a CG solver with diagonal scaling, denoted by cg-diag, *ii)* a CG solver with elementwise additive Schwarz preconditioning, denoted by cg-eas, *iii)* the solution approach presented in this contribution that consists of a CG solver with a p-multigrid preconditioner and elementwise additive Schwarz smoothing, denoted by cgmg-eas *iv)* a CG solver with an algebraic multigrid solver based on smoothed aggregation, denoted by cg-amg and *v)* a GMRES solver with an incomplete LU preconditioner, gmres-ilu. Note that the solvers cg-diag, cg-amg and grmes-ilu are included in the `AztecOO` package. These different solvers are used to solve the linear systems arising from the aluminium rod with $h = 2$ and $p = 3$. All simulations are carried out on 192 cores that are divided into 32 MPI tasks each with 6 OpenMP threads.

Figure 16 presents the comparison of the convergence behavior and execution time for the solution of a FCM problem on a complex geometry using five different solver and preconditioner configurations. The solver time shown in Figure 16b includes both the construction and application of the solvers. The results affirm that the additive Schwarz-based solution techniques are well suited for large immersed systems as they not only have convergence rates superior to those of conventional solvers but also exhibit lower computational times. It should be noted that the poor performance of the cg-amg and gmres-ilu solvers is attributed to the ill-conditioning due to the cut elements, which both solvers fail to adequately address.



(a) Relative residual.



(b) Solver time.

Figure 16: Comparison of the convergence behavior and execution time of different iterative solvers in the aluminum rod example.

4.3.2 Weak scalability: Popcorn benchmark

The final numerical example considers a classical immersed finite element benchmark on popcorn geometry. This geometry is commonly used in interface problems e.g. [63, 64] and immersed analyses, see e.g. [4]. A level-set function $\phi(x, y, z)$ is used to describe the surface of the popcorn geometry where

$$\phi(x, y, z) = \sqrt{x^2 + y^2 + z^2} - r_0 - \sum_{k=0}^{11} A e^{-((x-x_k)^2+(y-y_k)^2+(z-z_k)^2)/\sigma^2}, \tag{17}$$

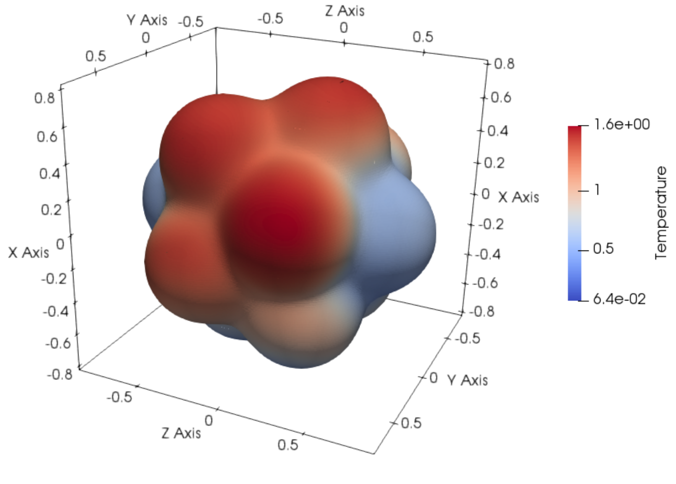
and

$$\begin{aligned}
(x_k, y_k, z_k) &= \frac{r_0}{\sqrt{5}}\Big(2\cos\Big(\frac{2k\pi}{5}\Big), \sin\Big(\frac{2k\pi}{5}\Big), 1\Big), \quad \text{for} \quad k \in [0, 4], \\
(x_k, y_k, z_k) &= \frac{r_0}{\sqrt{5}}\Big(2\cos\Big(\frac{(2(k-5)-1)\pi}{5}\Big), \sin\Big(\frac{(2(k-5)-1)\pi}{5}\Big), 1\Big), \quad \text{for} \quad k \in [5, 9], \\
(x_{10}, y_{10}, z_{10}) &= (0, 0, r_0), \\
(x_{11}, y_{11}, z_{11}) &= (0, 0, -r_0),
\end{aligned}$$

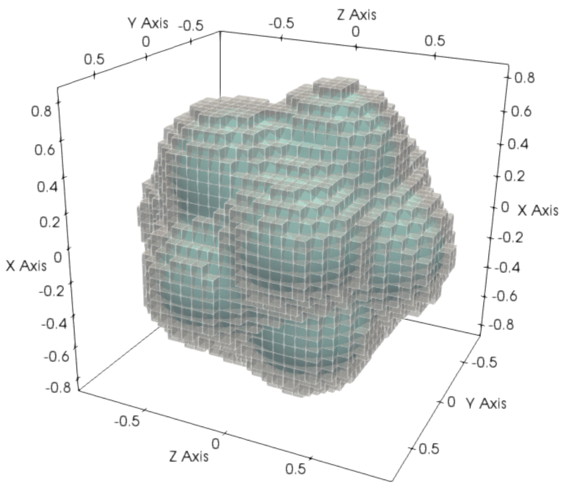
with the parameters r_0=0.6, A=3 and σ=0.2. The physical domain Ω_{phys} constitutes all points lying on the popcorn surface and its interior, i.e. all points with $\phi \leq 0$. Following [63], a Poisson problem with a prescribed solution field

$$u(x, y, z) = x^3 + xy^2 + y^3 + z^4 + \sin(3(x^2 + y^2)), \quad \boldsymbol{x} \in \Omega_{phys}. \tag{18}$$

is considered in the current example. This solution is defined through a source term $s = \Delta u$ and the enforcement of Dirichlet boundary conditions on $\phi = 0$. A marching cubes algorithm is run during the simulation to obtain the surfaces required for boundary condition application. The penalty method with $\beta = 10^4$ is used to apply the Dirichlet boundary conditions and the value of α is set to 10^{-6}. An embedding domain consisting of a cube $(-1, 1)^3$ is used in all numerical investigations.



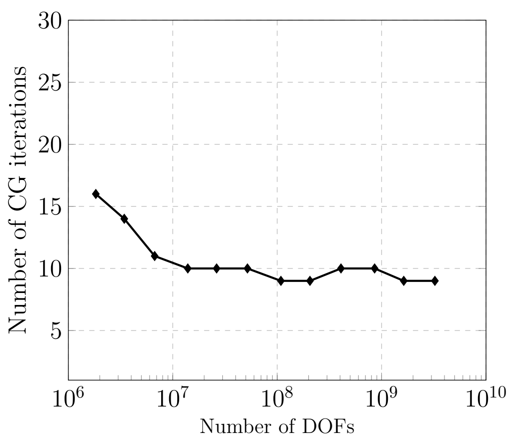
(a) Popcorn geometry.



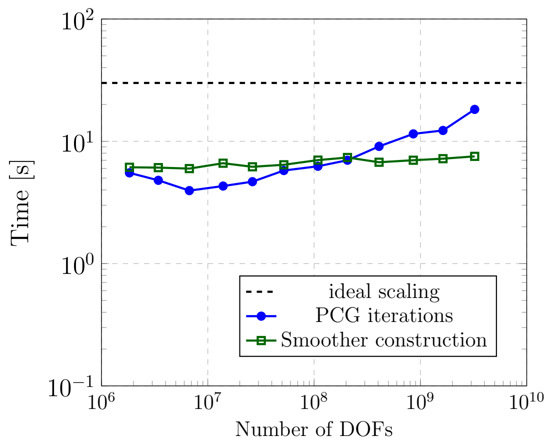
(b) Finite cell mesh.

Figure 17: Poisson problem posed on a popcorn domain.

A weak scaling analysis is performed to investigate the efficiency of the proposed hierarchical multigrid approach. The simulations are performed on the SuperMUC-NG supercomputer at the Leibniz Supercomputing Center in Garching, Germany. The number of elements is increased in 12 steps while at the same time increasing the number of compute nodes n_{nodes}, from 1 to 2048. The 48 cores within a node are partitioned such that each node has a total of 8 MPI tasks and 6 OpenMP threads per task. The computational domain is generated in a fully parallel manner and discretized using hexahedral trunk space elements with a polynomial order $p = 3$. The number of elements per MPI task is approximately 41 000 in all simulations. The simulation with the coarsest mesh comprising a total of 250 336 elements and $1.83 \cdot 10^6$ DOFs is run on 48 cores while the one with the finest mesh is run on 98 304 cores, and has a total of 460 980 224 elements and approximately $3.2 \cdot 10^9$ DOFs. The different linear systems are solved using a parallel CG solver with a p-multigrid preconditioner and elementwise additive Schwarz smoothing.



(a) Convergence behavior.



(b) Solver time.

Figure 18: Weak scaling analysis of the popcorn benchmark.

Figure 18 shows the results of the weak scaling analysis. It reports the number of iterations needed by the solver and the time spent to construct the additive Schwarz smoother and solve linear systems. The multigrid approach proposed in this manuscript shows favorable weak scaling, as the computational time does not significantly increase with an increase of the problem size.

5 Conclusions

The paper at hand presents a hierarchical multigrid method for immersed high-order discretizations based on the finite cell method and multi-level *hp*-refinement. This scheme robustly deals with ill-conditioning due to cut elements resulting in convergence rates independent of the cut configuration, element size and in certain scenarios, the polynomial order. The applicability of the scheme for immersed finite cell analyses on large distributed memory systems is portrayed in a collection of second-order problems arising from the Poisson equation and linear elasticity.

The cornerstone of this contribution is the development of a simple and efficient multigrid scheme that exploits the hierarchical nature of the basis functions in the finite cell method and the multi-level *hp*-scheme. For uniform grids, the multigrid levels are constructed using an arithmetic *p*-sequence that progressively reduces to polynomial order until $p = 1$. The nested subspace of a certain polynomial order is spanned by the basis functions up to and including this order. The *p*-multigrid approach is combined with an *h*-coarsening of the refined elements in the case of multi-level *hp*-refined grids. The main benefit of the proposed scheme is its simplicity, as the restriction and prolongation operations reduce to binary operations that can be easily performed without the assembly of restriction and prolongation operators.

Additive Schwarz-based smoothers are applied in this contribution to deal with the ill-conditioning due to cut cells. Two different procedures are suggested for finite cell discretizations: an elementwise approach is applied to uniform grids and a patchwise approach is used for multi-level *hp*-refined grids. Both smoothers are shown to sufficiently deal with small modes that occur on cut cells and yield convergence rates independent of the cut configuration and the mesh size h. The patchwise additive Schwarz smoother is shown to yield convergence rates independent of the polynomial order in the two dimensional numerical tests.

Furthermore, we demonstrate that the proposed multigrid scheme can be implemented within an efficient finite element framework in a massively parallel environment. A series of numerical examples demonstrate the suitability of the presented solution technique for large-scale immersed analysis. Faster convergence rates are achieved when the multigrid scheme is used as a preconditioner within a

CG method rather than a stand-alone solver. The execution time for selected benchmark examples is also provided. These results indicated that the presented multigrid scheme exhibits low computation times as well as favorable weak scalability. The proposed scheme is shown to outperform standard iterative solvers both in terms of iteration counts and execution time for immersed systems.

Although the solution scheme presented in this contribution significantly improves the solution of high-order immersed systems involving *hp*-refinement, it can be further developed to increase its applicability. The construction of the patchwise preconditioner for high polynomials orders and several levels of refinement can be improved by applying more efficient matrix inversion techniques. This can be combined with the use of alternative strategies for the selection of additive Schwarz groups. These strategies can be designed to yield smaller group sizes and lower execution times. In this work, additive Schwarz smoothers are applied on every multigrid level with $\ell \neq 0$. The use of variable smoothing techniques on different multigrid levels is an important subject that can be addressed in future research. Apart from improving the efficiency of the solver, it would be also beneficial to invest time in a thorough mathematical analysis of the proposed multigrid method. This entails the derivation of bounds for the condition numbers as well as mathematical proofs for the multigrid method in an immersed setting.

Acknowledgements

The authors gratefully acknowledge the Competence Network for Scientific High Performance Computing in Bavaria (KONWIHR) and the Gauss Centre for Supercomputing e.V. (www.gauss-centre.eu) for the financial support and computing time provided on the SUPERMUG-NG at Leibniz Supercomputing Centre (www.lrz.de). We also extend our gratitude to the International Graduate School of Science and Engineering (IGSSE) of the Technical University of Munich for its financial support as well as the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Projektnummer 414265976 – TRR 277.

References

[1] J. Parvizian, A. Düster, and E. Rank, “Finite cell method,” *Computational Mechanics*, vol. 41, no. 1, pp. 121–133, 2007.

[2] A. Düster, J. Parvizian, Z. Yang, and E. Rank, “The finite cell method for three-dimensional problems of solid mechanics,” *Computer Methods in Applied Mechanics and Engineering*, vol. 197, no. 45–48, pp. 3768–3782, 2008.

[3] E. Burman and P. Hansbo, “Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method,” *Applied Numerical Mathematics*, vol. 62, no. 4, pp. 328 – 341, 2012.

[4] E. Burman, S. Claus, P. Hansbo, M. G. Larson, and A. Massing, “CutFEM: Discretizing geometry and partial differential equations,” *International Journal for Numerical Methods in Engineering*, vol. 104, no. 7, pp. 472–501, 2014.

[5] S. Badia, F. Verdugo, and A. F. Martín, “The aggregated unfitted finite element method for elliptic problems,” *Computer Methods in Applied Mechanics and Engineering*, vol. 336, pp. 533–553, 2018.

[6] E. Nadal Soriano, J. Ródenas, J. Albelda, M. Tur, J. Tarancón, and F. Fuenmayor, “Efficient finite element methodology based on cartesian grids: Application to structural shape optimization,” *Abstract and Applied Analysis*, 2013.

[7] J. M. Navarro-Jiménez, M. Tur, F. J. Fuenmayor, and J. J. Ródenas, “On the effect of the contact surface definition in the cartesian grid finite element method,” *Advanced Modeling and Simulation in Engineering Sciences*, vol. 5, p. 12, May 2018.

[8] A. Main and G. Scovazzi, “The shifted boundary method for embedded domain computations. Part I: Poisson and Stokes problems,” *Journal of Computational Physics*, vol. 372, pp. 972–995, 2018.

[9] A. Main and G. Scovazzi, “The shifted boundary method for embedded domain computations. Part II: Linear advection-diffusion and incompressible Navier-Stokes equations,” *Journal of Computational Physics*, vol. 372, pp. 996–1026, 2018.

[10] F. de Prenter, C.V. Verhoosel, G.J. van Zwieten, and E.H. van Brummelen, “Condition number analysis and preconditioning of the finite cell method,” *Computer Methods in Applied Mechanics and Engineering*, vol. 316, pp. 297–327, 2017.

[11] S. Badia and F. Verdugo, “Robust and scalable domain decomposition solvers for unfitted finite element methods,” *Journal of Computational and Applied Mathematics*, vol. 344, pp. 740–759, 2017.

[12] F. de Prenter, C. Verhoosel, and E. van Brummelen, “Preconditioning immersed isogeometric finite element methods with application to flow problems,” *Computer Methods in Applied Mechanics and Engineering*, vol. 348, pp. 604–631, 2019.

[13] J. Jomo, F. de Prenter, M. Elhaddad, D. D’Angella, C. Verhoosel, S. Kollmannsberger, J. Kirschke, V. Nübel, E. van Brummelen, and E. Rank, “Robust and parallel scalable iterative solutions for large-scale finite cell analyses,” *Finite Elements in Analysis and Design*, vol. 163, pp. 14–30, 2019.

[14] E. Burman, “Ghost penalty,” *Comptes Rendus Mathematique*, vol. 348, no. 21, pp. 1217 – 1220, 2010.

[15] D. Elfverson, M. G. Larson, and K. Larsson, “CutIGA with basis function removal,” *Advanced Modeling and Simulation in Engineering Sciences*, vol. 5, no. 1, p. 6, 2018.

[16] B. Marussig and T. Hughes, “A Review of Trimming in Isogeometric Analysis: Challenges, Data Exchange and Simulation Aspects,” *Archives of Computational Methods in Engineering*, vol. 25, no. 4, pp. 1059–1127, 2017.

[17] W. Hackbusch and U. Trottenberg, *Multigrid Methods. Proceedings of the Conference Held at Köln-Porz, November 23-27, 1981.* 01 1982.

[18] W. Hackbusch, *Multi-Grid Methods and Applications.* Springer Series in Computational Mathematics, Springer Berlin Heidelberg, 2013.

[19] U. Trottenberg, C. Ulrich Trottenberg, C. Oosterlee, A. Schuller, A. Brandt, P. Oswald, and K. Stüben, *Multigrid.* Elsevier Science, 2001.

[20] W. Briggs, V. Henson, and S. McCormick, *A Multigrid Tutorial, 2nd Edition.* 01 2000.

[21] P. Wesseling, *An Introduction to Multigrid Methods.* An Introduction to Multigrid Methods, R.T. Edwards, 2004.

[22] Y. Shapira, *Matrix-Based Multigrid: Theory and Applications.* Numerical methods and algorithms, Kluwer Academic Publishers, 2003.

[23] A. W. Craig and O. C. Zienkiewicz, "A multigrid algorithm using a hierarchical finite element basis," in *Multigrid Methods for Integral and Differential Equations* (D. J. Paddon and Holstein, eds.), pp. 310–312, Oxford,: Clarendon Press, 1985.

[24] I. Babuška, M. Griebel, and J. Pitkäranta, "The problem of selecting the shape functions for a p-type finite element," *International Journal for Numerical Methods in Engineering*, vol. 28, no. 8, pp. 1891–1908, 1989.

[25] C. Hofreither, B. Jüttler, G. Kiss, and W. Zulehner, "Multigrid methods for isogeometric analysis with THB-splines," *Computer Methods in Applied Mechanics and Engineering*, vol. 308, pp. 96–112, 2016.

[26] K. Gahalaut, J. Kraus, and S. Tomar, "Multigrid methods for isogeometric discretization," *Computer Methods in Applied Mechanics and Engineering*, vol. 253, pp. 413–425, 2013.

[27] R. Tielen, M. Möller, D. Göddeke, and C. Vuik, "p-multigrid methods and their comparison to h-multigrid methods within Isogeometric Analysis," *Computer Methods in Applied Mechanics and Engineering*, vol. 372, p. 113347, 2020.

[28] A. P. de la Riva, C. Rodrigo, and F. J. Gaspar, "A Robust Multigrid Solver for Isogeometric Analysis Based on Multiplicative Schwarz Smoothers," *SIAM Journal on Scientific Computing*, vol. 41, pp. 321–345, 2019.

[29] M. Donatelli, C. Garoni, C. Manni, S. Serra-Capizzano, and H. Speleers, "Robust and optimal multi-iterative techniques for IgA Galerkin linear systems," *Computer Methods in Applied Mechanics and Engineering*, vol. 284, pp. 230–264, 2015.

[30] M. Donatelli, C. Garoni, C. Manni, S. Serra-Capizzano, and H. Speleers, "Symbol-Based Multigrid Methods for Galerkin B-Spline Isogeometric Analysis," *SIAM Journal on Numerical Analysis*, vol. 55, no. 1, pp. 31–62, 2017.

[31] C. Bracco, D. Cho, C. Giannelli, and R. Vazquez, "BPX Preconditioners for Isogeometric Analysis Using (Truncated) Hierarchical B-splines," 2019.

[32] F. Verdugo, A. F. Martín, and S. Badia, "Distributed-memory parallelization of the aggregated unfitted finite element method," *Computer Methods in Applied Mechanics and Engineering*, vol. 357, p. 112583, 2019.

[33] S. Badia, A. F. Martín, E. Neiva, and F. Verdugo, "The aggregated unfitted finite element method on parallel tree-based adaptive meshes," *preprint available on arXiv*, 2020.

[34] S. Badia, A. F. Martín, E. Neiva, and F. Verdugo, "A generic finite element framework on parallel tree-based adaptive meshes," *preprint available on arXiv*, 2020.

[35] A. Nüßing, *Fitted and unitted finite element methods for solving the EEG forward problem.* PhD Thesis, University of Münster, Münster, 2018.

[36] F. de Prenter, C. V. Verhoosel, E. H. van Brummelen, J. A. Evans, C. Messe, J. Benzaken, and K. Maute, "Multigrid solvers for immersed finite element methods and immersed isogeometric analysis," *Computational Mechanics*, 2019.

[37] S. Saberi, A. Vogel, and G. Meschke, "Parallel finite cell method with adaptive geometric multigrid," in *Euro-Par 2020: Parallel Processing* (M. Malawski and K. Rzadca, eds.), (Cham), pp. 578–593, Springer International Publishing, 2020.

[38] E. M. Rønquist and A. Patera, "Spectral element multigrid. I. Formulation and numerical results," *Journal of Scientific Computing*, vol. 2, pp. 389–406, 1987.

[39] H. Yserentant, "Hierarchical bases of finite-element spaces in the discretization of nonsymmetric elliptic boundary value problems," *Computing*, vol. 35, pp. 39–49, 1985.

[40] H. Yserentant, "Hierarchical bases give conjugate gradient type methods a multigrid speed of convergence," *Applied Mathematics and Computation*, vol. 19, no. 1, pp. 347–358, 1986.

[41] S. Foresti, G. Brussino, S. Hassanzadeh, and V. Sonnad, "Multilevel solution of the p-version of finite elements," *Computer Physics Communications*, vol. 53, pp. 349–355, 05 1989.

[42] W. F. Mitchell, "The hp-multigrid method applied to hp-adaptive refinement of triangular grids," *Numerical Linear Algebra with Applications*, vol. 17, no. 2-3, pp. 211–228, 2010.

[43] C. R. Nastase and D. J. Mavriplis, "High-order discontinuous Galerkin methods using an hp-multigrid approach," *Journal of Computational Physics*, vol. 213, no. 1, pp. 330 – 357, 2006.

[44] H. Yserentant, "On the multi-level splitting of finite element spaces," *Numerische Mathematik*, vol. 49, pp. 379–412, 1986.

[45] N. Zander, T. Bog, S. Kollmannsberger, D. Schillinger, and E. Rank, "Multi-Level hp-Adaptivity: High-Order Mesh Adaptivity without the Difficulties of Constraining Hanging Nodes," *Computational Mechanics*, vol. 55, no. 3, pp. 499–517, 2015.

[46] N. Zander, T. Bog, M. Elhaddad, F. Frischmann, S. Kollmannsberger, and E. Rank, "The Multi-Level hp-Method for Three-Dimensional Problems: Dynamically Changing High-Order Mesh Refinement with Arbitrary Hanging Nodes," *Computer Methods in Applied Mechanics and Engineering*, vol. 310, pp. 252–277, 2016.

[47] I. Babuska, B. Szabo, and I. Katz, "The p-Version of the Finite Element Method," *SIAM Journal on Numerical Analysis*, vol. 18, no. 3, pp. 515–545, 1981.

[48] M. Elhaddad, N. Zander, T. Bog, L. Kudela, S. Kollmannsberger, J. Kirschke, T. Baum, M. Ruess, and E. Rank, "Multi-level hp-finite cell method for embedded interface problems with application in biomechanics," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 34, no. 4, p. e2951, 2018.

[49] A. Özcan, S. Kollmannsberger, J. Jomo, and E. Rank, "Residual stresses in metal deposition modeling: Discretizations of higher order," *Computers & Mathematics with Applications*, 2018.

[50] L. Hug, S. Kollmannsberger, Z. Yosibash, and E. Rank, "A 3D benchmark problem for crack propagation in brittle fracture," *Computer Methods in Applied Mechanics and Engineering*, vol. 364, p. 112905, 2020.

[51] I. Babuška, "The Finite Element Method with Penalty," *Mathematics of Computation*, vol. 27, no. 122, p. 221, 1973.

[52] D. Schillinger and M. Ruess, "The Finite Cell Method: A Review in the Context of Higher-Order Structural Analysis of CAD and Image-Based Geometric Models," *Archives of Computational Methods in Engineering*, vol. 22, no. 3, pp. 391–455, 2014.

[53] A. Düster, E. Rank, and B. A. Szabó, "The p-version of the finite element method and finite cell methods," in *Encyclopedia of Computational Mechanics*, vol. 2, pp. 1–35, Chichester, West Sussex: John Wiley & Sons, 2017.

[54] D. D'Angella, N. Zander, S. Kollmannsberger, F. Frischmann, E. Rank, A. Schröder, and A. Reali, "Multi-level hp-adaptivity and explicit error estimation," *Advanced Modeling and Simulation in Engineering Sciences*, vol. 3, no. 1, p. 33, 2016.

[55] V. Darrigrand, D. Pardo, T. Chaumont-Frelet, I. Gómez-Revuelto, and L. E. Garcia-Castillo, "A painless automatic hp-adaptive strategy for elliptic problems," *Finite Elements in Analysis and Design*, vol. 178, p. 103424, 2020.

[56] N. Zander, *Multi-Level hp-FEM: Dynamically Changing High-Order Mesh Refinement with Arbitrary Hanging Nodes.* PhD Thesis, Technische Universität München, Munich, 2016.

[57] Y. Saad, *Iterative Methods for Sparse Linear Systems.* Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2nd ed., 2003.

[58] M. P. Forum, "MPI: A Message-Passing Interface Standard," tech. rep., Knoxville, TN, USA, 1994.

[59] OpenMP Architecture Review Board, "OpenMP Application Program Interface Version 3.0," 2008.

[60] M. A. Heroux, R. A. Bartlett, V. E. Howle, R. J. Hoekstra, J. J. Hu, T. G. Kolda, R. B. Lehoucq, K. R. Long, R. P. Pawlowski, E. T. Phipps, A. G. Salinger, H. K. Thornquist, R. S. Tuminaro, J. M. Willenbring, A. Williams, and K. S. Stanley, "An overview of the Trilinos project," *ACM Transactions on Mathematical Software*, vol. 31, no. 3, pp. 397–423, 2005.

[61] O. Schenk and K. Gärtner, *PARDISO*, pp. 1458–1464. Boston, MA: Springer US, 2011.

[62] B. Schling, *The Boost C++ Libraries.* XML Press, 2011.

[63] I.-L. Chern and Y.-C. Shu, "A coupling interface method for elliptic interface problems," *Journal of Computational Physics*, vol. 225, no. 2, pp. 2138–2174, 2007.

[64] C. Annavarapu, M. Hautefeuille, and J. Dolbow, "A robust Nitsche's formulation for interface problems," *Computer Methods in Applied Mechanics and Engineering*, vol. s 225–228, pp. 44–54, 06 2012.