Solving an Elliptic PDE Eigenvalue Problem via Automated Multi-Level Substructuring and Hierarchical Matrices

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Diplom-Wirtschaftsmathematiker
Peter Gerds

aus Leipzig

Berichter: Universtitätsprofessor Dr. Lars Grasedyck
Universtitätsprofessor Dr. Steffen Börm

Tag der mündlichen Prüfung: 10.10.2017

Abstract

To solve an elliptic PDE eigenvalue problem in practice, typically the finite element discretisation is used. From approximation theory it is known that only the smaller eigenvalues and their corresponding eigenfunctions can be well approximated by the finite element discretisation because the approximation error increases with increasing size of the eigenvalue. The number of well approximable eigenvalues or eigenfunctions, however, is unknown. In this work asymptotic estimates of these quantities are derived. For example, it is shown that for three-dimensional problems under certain smoothness assumptions on the data only the smallest $\Theta(N^{2/5})$ eigenvalues and only the eigenfunctions associated to the smallest $\Theta(N^{1/4})$ eigenvalues can be well approximated by the finite element discretisation, when the N-dimensional finite element spaces of piecewise affine functions with uniform mesh refinement are used.

To solve the discretised elliptic PDE eigenvalue problem and to compute all well approximable eigenvalues and eigenfunctions, a new method is introduced which combines a recursive version of the *automated multi-level substructuring* (short AMLS) method with the concept of *hierarchical matrices* (short $\mathcal{H}$-matrices). AMLS is a domain decomposition technique for the solution of elliptic PDE eigenvalue problems where, after some transformation, a reduced eigenvalue problem is derived whose eigensolutions deliver approximations of the sought eigensolutions of the original problem.

Whereas the classical AMLS method is very efficient for elliptic PDE eigenvalue problems posed in two dimensions, it is getting very expensive for three-dimensional problems, due to the fact that it computes the reduced eigenvalue problem via dense matrix operations.

This problem is resolved by the use of hierarchical matrices. $\mathcal{H}$-matrices are a data-sparse approximation of dense matrices which, e.g., result from the inversion of the stiffness matrix that is associated to the finite element discretisation of an elliptic PDE operator. The big advantage of $\mathcal{H}$-matrices is that they provide matrix arithmetic with almost linear complexity. This fast $\mathcal{H}$-matrix arithmetic is used for the computation of the reduced eigenvalue problem. Beside this, the size of the reduced eigenvalue problem is bounded by a new recursive version of AMLS which further reduces the costs for the computation and the solution of this problem. Altogether this leads to a new method which is well-suited for three-dimensional problems and which allows us to compute a large amount of eigenpair approximations in optimal complexity.

Zusammenfassung

Elliptische PDE Eigenwertprobleme werden in der Praxis typischerweise mithilfe der Finite-Element-Diskretisierung gelöst. Aus der Approximationstheorie ist bekannt, dass nur die kleinsten Eigenwerte und die zugehörigen Eigenfunktionen sich gut durch die Finite-Element-Diskretisierung approximieren lassen, da der entsprechende Approximationsfehler mit der Größe des Eigenwertes wächst. Resultate bezüglich der Anzahl der gut approximierbaren Eigenwerte und Eigenfunktionen sind bisher aber noch unbekannt. In dieser Arbeit werden Abschätzungen hergeleitet, die es erlauben diese Größen asymptotisch zu beschreiben. So wird zum Beispiel gezeigt, dass für drei-dimensionale Probleme (unter bestimmten Glattheitsbedingungen der Daten) nur die kleinsten $\Theta(N^{2/5})$ Eigenwerte und die Eigenfunktionen zu den kleinsten $\Theta(N^{1/4})$ Eigenwerten gut durch die Finite-Element-Diskretisierung approximierbar sind, wenn N-dimensionale Finite-Element-Räume mit stückweise affinen Funktionen bei gleichmäßiger Gitterverfeinerung verwendet werden.

Um das diskretisierte elliptische PDE Eigenwertproblem zu lösen und um alle gut approximierbaren Eigenwerte und Eigenfunktionen zu berechnen, wird in dieser Arbeit eine neue Methode vorgestellt, welche eine rekursive Version der *Automated Multi-Level Substructuring* (kurz AMLS) Methode mit dem Konzept der *hierarchischen Matrizen* (kurz $\mathcal{H}$-Matrix) kombiniert. AMLS ist eine Gebietszerlegungsmethode zum Lösen elliptischer PDE Eigenwertprobleme, bei der nach einer bestimmten Problemtransformation ein reduziertes Eigenwertproblem aufgestellt wird. Die Eigenlösungen des reduzierten Problems liefern schließlich Approximationen der gesuchten Eigenlösungen des Ausgangsproblems.

Die klassische AMLS Methode ist sehr effizient für PDE Eigenwertprobleme definiert auf einem zwei-dimensionalen Gebiet, jedoch wird die Methode sehr teuer für drei-dimensionale Probleme, da in AMLS das reduzierte Problem mittels vollbesetzter Matrixoperationen berechnet wird.

In dieser Arbeit wird dieses Effizienzproblem von AMLS durch den Einsatz von hierarchischen Matrizen gelöst. $\mathcal{H}$-Matrizen sind kosteneffiziente Approximation von vollbesetzten Matrizen, welche beispielsweise auftreten bei der Invertierung der Steifigkeitsmatrix der Finite-Element-Diskretisierung elliptischer PDE Operatoren. Der große Vorteil von $\mathcal{H}$-Matrizen ist, dass diese eine Matrixarithmetik mit fast linearer Komplexität ermöglichen. Diese schnelle $\mathcal{H}$-Matrixarithmetik wird verwendet um das reduzierte Problem zu berechnen. Darüber hinaus wird die Größe des reduzierten Problems durch eine neue rekursive Version von AMLS beschränkt, was die Kosten für das Aufstellen und das Lösen des reduzierten Problems weiter verringert. Insgesamt führt dies zu einer neuen Methode, welche sehr gut geeignet ist um drei-dimensionale elliptische PDE Eigenwertprobleme zu lösen und welche eine Vielzahl von Eigenpaar Approximationen in optimaler Komplexität berechnet.

Acknowledgements

First of all, I would like to thank Prof. Dr. Lars Grasedyck for giving me the opportunity to perform my research on elliptic PDE eigenvalue problems and the AMLS method which has been the basis for this thesis. I am very grateful for his guidance and support, he had always time for discussions when assistance was required. It was a pleasure to work with him.

Furthermore, I want to thank Dr. Ronald Kriemann for providing me the software library HLIBpro [50] which made the implementation of my program code much easier, and which helped to improve my programming skills.

Writing my thesis and working at the *Institut für Geometrie und Praktische Mathematik* was a great time and interesting experience, in particular also because of the great friendliness and helpfulness of my colleagues. Working together in teaching, private conversations and activities with my colleagues have been a welcome change from my everyday research.

Last but not least, I want to thank my family and all my friends for their support and their interest in my work. At this point I want to thank in particular Veronique Overhoff and Paul Gerds for their everlasting encouragement.

Contents

List of Symbols and Abbreviations

Abbreviations

EVP	eigenvalue problem
FEM	finite element method
PDE	partial differential equation
SIL	shift-invert Lanczos

Greek Letters

Δu	Laplace operator $\Delta u := \sum_{i=11}^{d} \frac{\partial^2}{\partial x_i^2} u$ for $u : \mathbb{R}^d \to \mathbb{R}$
$\rho(T)$	resolvent set of an operator $T \in L(X)$, page 147
$\sigma(T)$, $\sigma_p(T)$	spectrum and point spectrum of an operator $T \in L(X)$, page 147

Miscellaneous Symbols

$(\cdot,\cdot)_0$, $\|\cdot\|_0$	inner product and norm on $L^2(\Omega)$, page 156
$(\cdot,\cdot)_k$, $\|\cdot\|_k$, $\|\cdot\|_k$	inner product, norm and seminorm on $H^k(\Omega)$, page 159
$(\cdot,\cdot)_X$, $\|\cdot\|_X$	inner product and norm on linear space X, page 149
$\lceil x \rceil$, $\lfloor x \rfloor$	$\lceil x \rceil := \min\{n \in \mathbb{Z} : n \geq x\}$ and $\lfloor x \rfloor := \max\{n \in \mathbb{Z} : n \leq x\}$
∇u	gradient $\nabla u := (\frac{\partial}{\partial x_1} u, \ldots, \frac{\partial}{\partial x_d} u)^T$ for $u : \mathbb{R}^d \to \mathbb{R}$
$\neg a$	logical negation of statement a
$\| f \|_\infty$	supremum norm of function $f : \Omega \to \mathbb{R}$ is $\| f \|_\infty := \sup_{x \in \Omega} \| f(x) \|$
$\| T \|_{Y \leftarrow X}$	operator norm of $T : X \to Y$, page 146
$\partial / \partial n$	normal derivative, page 158
$\partial\Omega$	boundary of a subset $\Omega \subset \mathbb{R}^d$, page 156
$\mathbb{R}_{>0}$	$\mathbb{R}_{>0} := \{x \in \mathbb{R} : x > 0\}$
$a \equiv b$	logical equivalence of statement a and b
$f \in \mathcal{O}(g)$	$\limsup_{x \to x_0} \| f(x)/g(x) \| < \infty$ with $x_0 \in \mathbb{R} \cup \{\infty, -\infty\}$
$f \in \Omega(g)$	$\liminf_{x \to x_0} \| f(x)/g(x) \| > 0$ with $x_0 \in \mathbb{R} \cup \{\infty, -\infty\}$

$f \in \Theta(g)$	$0 < \liminf_{x\to x_0} \mid f(x)/g(x) \mid \leq \limsup_{x\to x_0} \mid f(x)/g(x) \mid < \infty$
$f \in o(g)$	$\lim_{x\to x_0} \mid f(x)/g(x) \mid = 0$ with $x_0 \in \mathbb{R} \cup \{\infty, -\infty\}$

Roman Letters

$L^\infty(\Omega)$	set of functions that are bounded almost everywhere, page 156
div$F$	divergence div$F := \sum_{i=1}^d \frac{\partial}{\partial x_i} F_i$ for $F : \mathbb{R}^d \to \mathbb{R}^d$
$\mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$	set of $\mathcal{H}$-matrices induced by block cluster tree $\mathcal{T}_{I\times I}$, page 69
Id	quadratic identity matrix
Im(T)	image (range) of $T \in L(X,Y)$ is given by $\{Tx \in Y : x \in X\}$, page 147
Ker(T)	kernel (null space) of $T \in L(X,Y)$ is given by $\{x \in X : Tx = 0\}$, page 147
$H^k(\Omega)$, $H_0^k(\Omega)$	Sobolev spaces, page 159
$K(X,Y)$, $K(X)$	set of compact operators from X to Y, and X to X, page 147
$L^2(\Omega)$	space of square integrable functions $f : \Omega \to \mathbb{R}$, page 156
$L(X,Y)$, $L(X)$	set of bounded operators from X to Y, and X to X, page 146
$\mathcal{L}(\mathcal{T}_I)$, $\mathcal{L}(\mathcal{T}_{I\times I})$	leaves of cluster tree and leaves of block cluster tree, page 68
$\mathcal{T}$, $\mathcal{T}_h$	triangulation of a domain, page 30
$\mathcal{P}$	prolongation operator, page 24
supp(f)	support of the function f, page 155
$C_0^\infty(\Omega)$	set of $C^\infty(\Omega)$-functions with compact supports, page 155
$\mathcal{T}_I$, $\mathcal{T}_{I\times I}$	cluster tree and block cluster tree, page 68
$H_{00}^{1/2}(\Gamma)$	trace space of $H_0^1(\Omega)$ on the interface Γ, page 47
$C^{0,\lambda}(\overline{\Omega})$,$C^{0,1}(\overline{\Omega})$	set of Hölder/Lipschitz continuous functions , page 156
$C^0(\Omega)$	set of continuous functions $f : \Omega \to \mathbb{R}$, page 155
$C^\infty(\Omega)$	set of functions which have continuous derivatives of all orders, page 155
$C^{k,\lambda}$, $C^{0,1}$	different smoothness classes for a boundary $\partial\Omega$, page 157
$C^{k,\lambda}(\overline{\Omega})$, $C^{k,1}(\overline{\Omega})$	set of k-fold Hölder/Lipschitz continuously differentiable functions , page 157
$C^k(\Omega)$, $C^\infty(\Omega)$	set of k-fold and infinitely continuously differentiable functions, page 155
$E(\lambda)$	eigenspace of eigenvalue λ, page 18
$H^{1/2}(\partial\Omega)$	trace space on the boundary $\partial\Omega$ (Sobolev-Slobodeckij space), page 160
I	identity mapping $I : X \to Y$, $x \mapsto x$ with $X \subset Y$, page 147

1. Introduction

This work focuses on the efficient solution of the continuous eigenvalue problem

$$\begin{cases} Lu = \lambda u & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega \end{cases} \tag{1.1}$$

where Ω is a bounded d-dimensional domain (d=2,3) with a Lipschitz boundary $\partial\Omega$ and L is a uniformly elliptic second order partial differential operator in divergence form

$$Lu = -\mathrm{div}\big(A\nabla u\big) + cu = -\sum_{i,j=1}^{d} \frac{\partial}{\partial x_i}\left(a_{ij}\frac{\partial}{\partial x_j}u\right) + cu$$

with $L^\infty(\Omega)$-functions a_{ij}, c where $A := (a_{ij})_{i,j=1}^{d}$ and $c \geq 0$, and where eigenvalues $\lambda \in \mathbb{C}$ and associated eigenfunctions $u \neq 0$ are sought. Partial differential equation (short PDE) eigenvalue problems of the form (1.1) arise in many fields of physical and engineering application, and their efficient solution is of high importance, especially for costly three-dimensional real-world problems.

In contrast to the common boundary value problem

$$\begin{cases} Lu = f & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega \end{cases} \tag{1.2}$$

where for a fixed right-hand side $f \in L^2(\Omega)$ a suitable solution u is sought, the eigenvalue problem (1.1) is non-linear since both the eigenvalue λ and the eigenfunction u are unknown. For this reason the analysis and the solution of eigenvalue problem (1.1) is much more challenging than it is for boundary value problem (1.2). The Fredholm-Riesz-Schauder theory has to be applied in order to show that eigenvalue problem (1.1) possesses, after some reformulation, a countable family of *weak* eigensolutions

$$\big(\lambda_j, u_j\big)_{j=1}^{\infty} \in \mathbb{R}_{>0} \times H_0^1(\Omega) \setminus \{0\} \qquad \text{with } \lambda_j \leq \lambda_{j+1}. \tag{1.3}$$

In the most cases, the eigensolutions (1.3) cannot be computed analytically, they have to be computed numerically. In practice typically the finite element discretisation is applied for this purpose: Using an N-dimensional finite element space denoted by $V_N \subset H_0^1(\Omega)$ and spanned by its basis functions $(\varphi_i^{(N)})_{i=1}^{N}$, the continuous eigenvalue problem is discretised and an algebraic eigenvalue problem of the form

$$\begin{cases} \text{find } (\lambda^{(N)}, x^{(N)}) \in \mathbb{R} \times \mathbb{R}^N \setminus \{0\} \text{ with} \\ K^{(N)}\, x^{(N)} \;=\; \lambda^{(N)}\, M^{(N)}\, x^{(N)} \end{cases} \tag{1.4}$$

is derived with symmetric sparse matrices $K^{(N)}, M^{(N)} \in \mathbb{R}^{N\times N}$, and with eigenpairs

$$\left(\lambda_j^{(N)}, x_j^{(N)}\right)_{j=1}^{N} \in \mathbb{R}_{>0} \times V_N \setminus \{0\} \qquad \text{with } \lambda_j^{(N)} \leq \lambda_{j+1}^{(N)}. \tag{1.5}$$

The discrete eigensolutions $(\lambda_j^{(N)}, u_j^{(N)})$ are then used for the approximation of the sought continuous eigensolutions (λ_j, u_j) when $N \to \infty$, and where the functions $u_j^{(N)} \in V_N$ are defined by $u_j^{(N)} := \sum_{i=1}^{N} (x_j^{(N)})_i \; \varphi_i^{(N)}$ for $j = 1, \ldots, N$.

Using the finite element discretisation for the solution of eigenvalue problem (1.1), two elementary questions arise which are covered in this work: The first question is which of the discrete eigensolutions $(\lambda_j^{(N)}, u_j^{(N)})$ provide good approximations for the sought continuous eigensolutions (1.3). From approximation theory (see, e.g., [9, 64]) it is known that only the smaller eigenvalues λ_j and their corresponding eigenfunctions u_j can be well approximated by the finite element space V_N because the approximation error increases with increasing size of the eigenvalue. To the best of the author's knowledge, results on the number of well approximable eigensolutions are not available in literature. However, based on the error estimates presented in [64], the author could derive in this work asymptotic bounds for these quantities. To derive these bounds, it is essential to show that the eigenvalues λ_j are asymptotically described by $\lambda_j \in \Theta(j^{2/d})$. This result is proved as well in this work and is new to the best of the author's knowledge. Altogether, it is shown that, for example, for three-dimensional problems under certain smoothness assumptions on the data only the first $\Theta(N^{2/5})$ eigenvalues and only the first $\Theta(N^{1/4})$ eigenfunctions can be well approximated by the finite element discretisation using the finite element spaces $(V_N)_{N\in\mathbb{N}}$ of piecewise affine functions with uniform mesh refinement.

Hence, in this work we are only interested in computing a portion of the smallest eigenpairs of the discrete problem (1.4), e.g., the first

$$n_{\text{es}} = CN^{2/5} \in \mathbb{N} \quad \text{or} \quad n_{\text{es}} = CN^{1/4} \in \mathbb{N}$$

eigenpairs with some constant $C > 0$. The computation of the remaining eigenpairs of (4.4), that are associated associated to larger eigenvalues, is not reasonable because typically they do not provide useful approximations for the continuous eigensolutions (1.3).

The second question, which is the focus of this work, is how for a given finite element space V_N all well approximable eigenvalues and eigenfunctions can be efficiently computed. In practice the eigenpairs $(\lambda_j^{(N)}, x_j^{(N)})_{j=1}^{n_{\text{es}}}$ of the discrete problem (1.4) are typically computed by a classical approach, i.e., by some iterative algebraic eigensolver (such as the Lanczos method [8] or the subspace iteration [10]) which is coupled with a preconditioner or a linear solver. Such classical approaches are well suited if the number of sought eigensolutions n_{es} is rather small, e.g., if $n_{\text{es}} = 5$. Another approach for the solution of eigenvalue problem (1.4), which is very efficient when a large amount of eigenpairs is sought, is the so-called *automated multi-level substructuring* (short AMLS) method.

The AMLS method is an efficient substructuring method for the solution of elliptic PDE eigenvalue problems, which was mainly developed by Bennighof and co-authors [14, 16, 47]. The idea behind AMLS is to substructure the domain Ω of eigenvalue problem (1.1) recursively into several subdomains that are separated by interfaces. On each of these subdomains and interfaces certain eigenvalue problems are defined which are induced by the global problem, and

which are typically small and easy to solve. In the next step, from each of these subproblems a few eigensolutions are computed which are meant to represent the global problem on the corresponding subdomain or interface. The computed eigensolutions of the subproblems are then used to form a subspace onto which the global eigenvalue problem is projected. The projection results in a *reduced eigenvalue problem* of smaller size which is typically easy to solve, and whose eigenpairs finally provide approximations of the sought n_{es} eigensolutions of the global problems (1.1) and (1.4).

The AMLS method has proven to be very efficient for solving large-scale eigenvalue problems arising in structural engineering analysis (see, e.g., [15, 47, 55]). Especially when a large number of eigenpair approximations is required, AMLS has shown to be more efficient than classical approaches (cf. [42]). Since the computational costs of AMLS increase only slightly with the number of sought eigenpairs, AMLS can compute a large number of eigenpairs at once. A very popular classical approach, which is commonly used in structural engineering analysis, is the *shift-invert block Lanczos* (short block-SIL) algorithm [36]. Breakthrough calculations could be presented in [55] when AMLS has been benchmarked against block-SIL within a vibro-acoustic analysis of an automobile body, and when AMLS running on a commodity workstation has been several times faster than block-SIL running on a supercomputer.

However, when AMLS is applied to a discrete eigenvalue problem it computes only approximations of the discrete eigenpairs whereas classical approaches, like block-SIL, compute these eigenpairs almost numerically exact. This seems to be disadvantageous, but since in our setting discrete eigenvalue problems result always from a finite element discretisation of a continuous problem, all eigenpairs of the discrete problem are related to a discretisation error. Hence, as long as the projection error caused by AMLS is of the same order as the discretisation error, the computed eigenpair approximations of AMLS are of comparable quality as the eigenpairs computed by some classical approach.

Although AMLS has proven to be very efficient, one problem is the computation of the above-mentioned interface eigenvalue problems via dense matrix operations. When AMLS is applied to three-dimensional problems the complexity is dominated by this part.

In this work a new method is presented which combines a recursive version of the AMLS method with the concept of *hierarchical matrices* (short $\mathcal{H}$-matrices). The new method is called $\mathcal{H}$-AMLS has been already introduced in [31] by the author. The $\mathcal{H}$-matrices [38, 39], which are used, are a data-sparse approximation of dense matrices which e.g. result from the inversion [13, 29] or the LU-factorisation [12, 29, 34, 59] of the stiffness matrix from the finite element discretisation of an elliptic PDE operator. The big advantage of $\mathcal{H}$-matrices is that they provide matrix arithmetic with almost linear complexity [33, 35]. In the new method this fast $\mathcal{H}$-matrix arithmetic is used for the computation of the interface eigenvalue problems. This allows us to treat also three-dimensional problems efficiently. Furthermore, it is essential in the AMLS method that the size of the reduced eigenvalue problem is kept small. This is achieved by a new recursive formulation of AMLS. This approach leads to a new method where all previously expensive steps of AMLS are performed in almost linear complexity $\mathcal{O}(N \log^{\alpha} N)$ where N denotes the size of eigenvalue problem (1.4). The remaining bottleneck is more of a theoretical nature: In order to set up the reduced eigenvalue problem and to extract the eigenvectors from the reduced problem, the $\mathcal{H}$-AMLS method involves $\mathcal{H}$-matrix times vector multiplications which accumulate to costs of the order $\mathcal{O}(n_{\text{es}} N \log^{\alpha} N)$, and it involves the usual scalar product which is accumulating to at most $\mathcal{O}(n_{\text{es}}^2 N)$ multiplications or additions. However, these operations

have very small constants involved so that their effect on practical computations is hardly visible. In numerical experiments it is observed that the computational costs of $\mathcal{H}$-AMLS stay in $\mathcal{O}(n_{\text{es}}N)$ for very large-scale problems, i.e., optimal complexity is reached. Furthermore, the different steps of the $\mathcal{H}$-AMLS method are very well parallelisable. To benefit from the multiple cores of today's workstations and compute servers, the $\mathcal{H}$-AMLS method has been parallelised for shared memory systems. Last but not least, this work introduces an additional improvement step for $\mathcal{H}$-AMLS which further improves the accuracy of the computed eigenpair approximations. Altogether, this leads to a very efficient eigensolver for elliptic PDE eigenvalue problems which is, in particular, well suited for problems posed in three dimensions.

The remainder of this work is organised as follows: In Chapter 2 the elliptic PDE eigenvalue problem (1.1) is analysed and results concerning the existence and the regularity of eigensolutions are presented. After this, in Chapter 3 the solution of eigenvalue problem (1.1) is discussed. This chapter includes a detailed analysis of the finite element discretisation where a priori error estimates for the eigenvalue and eigenfunction approximation are presented, and where asymptotic results concerning the number of well approximable eigenvalues and eigenfunctions are derived. The results of Chapter 2 and 3 are summarised in Chapter 4 where the underlying problem setting is specified. In Chapter 5 a description of the classical AMLS method is given, where the method is first explained and motivated in a continuous setting and then described in an algebraic setting to show how AMLS is applied in practice. Furthermore, it is outlined in Chapter 5 why the classical AMLS method is getting expensive for three-dimensional problems. In Chapter 6 a short introduction to $\mathcal{H}$-matrices is given and in Chapter 7 the new $\mathcal{H}$-AMLS method is presented. For numerical experiments an efficient implementation of $\mathcal{H}$-AMLS is important, especially for the parallelisation on shared memory systems. The issue of implementation and parallelisation is discussed in detail in Chapter 8. Finally, in Chapter 9 numerical results are presented where $\mathcal{H}$-AMLS is applied to three-dimensional problems and the approximation error, the computational time and the parallel performance of the method are analysed. The results of this work are summarised in Chapter 10 and an outlook is given.

To improve the readability, certain topics of this work are discussed in the appendix: In Appendix A abstract variational eigenvalue problems are analysed using the Fredholm-Riesz-Schauder theory, and in Appendix B results from the theory of Sobolev spaces are briefly recalled. The results of Appendix A and B are used in Chapter 2 when the elliptic PDE eigenvalue problem is analysed. Finally, in Appendix C results on the asymptotic distribution of the eigenvalues λ_j are derived, and in Appendix D error estimates for the finite element discretisation are provided. Appendix C and D are the foundation of Chapter 3 when the number of well approximable eigensolutions is discussed.

2. Analysis of Elliptic PDE Eigenvalue Problems

This chapter is focused on the analysis of the elliptic PDE eigenvalue problem

$$\begin{cases} L[u](x) = \lambda u(x) & \text{for all } x \in \Omega, \\ u(x) = 0 & \text{for all } x \in \partial\Omega \end{cases} \tag{2.1}$$

where $\Omega \subset \mathbb{R}^d$ is a bounded Lipschitz domain and L is a uniformly elliptic second order partial differential operator in divergency form

$$\begin{aligned} L[u](x) &= -\operatorname{div}\big(A\nabla u\big)(x) + c(x)u(x) \\ &= -\sum_{i,j=1}^{d} \frac{\partial}{\partial x_i}\left(a_{ij}(x)\frac{\partial}{\partial x_j}u(x)\right) + c(x)u(x) \qquad \text{for all } x \in \Omega \end{aligned} \tag{2.2}$$

with sufficiently smooth $L^\infty(\Omega)$-functions a_{ij}, c where $A := (a_{ij})_{i,j=1}^d$ and $c \geq 0$.

Definition 2.1 (Ellipticity) *The partial differential equation* (2.1) *and the associated operator* L *in* (2.2) *are called elliptic if for all* $x \in \Omega$ *the matrix* $A(x) := \big(a_{ij}(x)\big)_{i,j=1}^d$ *is symmetric positive definite, i.e., that the eigenvalues of* $A(x)$ *are positive. An elliptic operator* L *and the associated partial differential equation are called uniformly* [1] *elliptic if the eigenvalues of* $A(x)$ *are uniformly bounded from below by a positive constant, i.e., if it holds*

$$0 < a_{\min} := \inf_{x\in\Omega} \min_{\xi\in\mathbb{R}^d\setminus\{0\}} \frac{\xi^T A(x)\xi}{\xi^T\xi}.$$

Even though eigenvalue problem (2.1) is non-linear, its analysis is based in principle on the analysis of the linear PDE problem

$$\begin{cases} L[u](x) = f(x) & \text{for all } x \in \Omega, \\ u(x) = 0 & \text{for all } x \in \partial\Omega \end{cases} \tag{2.3}$$

where for a fixed right-hand side $f \in L^2(\Omega)$ a suitable solution u is sought. Since boundary value problem (2.3) possesses for each $f \in H_0^1(\Omega)$ a unique *weak* solution $u_f \in H_0^1(\Omega)$ a solution operator $T : H_0^1(\Omega) \to H_0^1(\Omega)$ is induced which allows to reformulate the weak formulation of eigenvalue problem (2.1) as an eigenvalue problem of an operator. The key in the analysis is to show that the solution operator T is compact, which allows to apply the Fredholm-Riesz-Schauder theory in order to characterise the spectrum of the operator T. This approach finally

[1] It is noted that in literature the definition of *uniform ellipticity* slightly differs: For example, in [37] the same definition for uniform ellipticity is used as here, whereas in [32] it is required that the ratio of maximum to minimum eigenvalue of the matrix $A(x)$ is bounded for $x \in \Omega$.

makes the elliptic PDE eigenvalue problem (2.1) accessible and it can be shown that it possess a countable family of weak eigensolutions. To improve the readability of this chapter, this approach is discussed in detail in Appendix A where variational eigenvalue problems are discussed in a general setting and where the Fredholm-Riesz-Schauder theory for compact operators is briefly recalled. The results derived in Appendix A are finally used in this chapter for the analysis of problem (2.1).

The remainder of this chapter is organised as follows: It is started in Section 2.1 with the analysis of boundary value problem (2.3), where the weak formulation of (2.3) is derived which guarantees the existence of a unique weak solution. In Section 2.2 the weak formulation of eigenvalue problem (2.3) is derived whose special structure allows to prove, based on the results derived in Appendix A, the existence of a countable family of weak eigensolutions. Beside this, in Section 2.2 the regularity of the weak eigenfunctions is discussed, i.e., it is discussed how the smoothness of the eigenfunctions depends on the smoothness of the domain and the coefficients of the PDE operator. Finally, in Section 2.3 an example is discussed in order to get a better understanding of the presented existence and regularity results for elliptic PDE eigenvalue problems.

Remark 2.2 *The restriction to homogeneous Dirichlet boundary conditions $u = 0$ on $\Gamma := \partial\Omega$ in* (2.1) *is not essential in this work, it only simplifies the analysis. Beside homogeneous Dirichlet boundary conditions the elliptic PDE eigenvalue problem $Lu = \lambda u$ in Ω can be equipped with various other boundary conditions (cf. [37, Chapter 11]), for example with:*

- *Homogeneous Neumann boundary conditions $(A\vec{n})^T \nabla u = 0$ on Γ where $\vec{n}$ is the exterior normal field on the boundary Γ (cf. Remark B.12).*
- *Mixed boundary conditions of the form $u = 0$ on Γ_D and $(A\vec{n})^T \nabla u = 0$ on Γ_N, where Γ_D and Γ_N are parts of the boundary Γ with positive $(d-1)$-dimensional measure such that it holds $\Gamma = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$.*

Throughout the whole chapter, if not differently noted, it is assumed that $\Omega \subset \mathbb{R}^d$ is a bounded domain with a Lipschitz boundary $\partial\Omega$.

2.1. Analysis of the Boundary Value Problem

The equations (2.3) are also referred to as the *classical formulation* of the boundary value problem, and the aim is to find a sufficiently smooth function $u : \Omega \to \mathbb{R}$ which is fulfilling (2.3).

Definition 2.3 (Classical Solution) *A function u is called classical solution of the boundary value problem* (2.3) *if $u \in C^2(\Omega) \cap C^0(\overline{\Omega})$ and if u fulfils both equations in* (2.3) *point-wise.*

However, finding such a classical solution can be difficult. The classical formulation of the boundary value problem, respectively the underlying function space

$$V := \left\{ u \in C^2(\Omega) \cap C^0(\overline{\Omega}) \,:\, u = 0 \text{ on } \partial\Omega \right\},$$

are rather unsuitable to derive satisfying results on existence or uniqueness of a solution. Instead of searching for solutions in the function space V it is often advantageous to search for solutions in a larger and less smooth function space, and to weaken the classical formulation slightly. The

space which will be used instead is the Sobolev space $H_0^1(\Omega) = \{\varphi \in H^1(\Omega) : \varphi|_{\partial\Omega} = 0\}$, i.e., the space of $L^2(\Omega)$-functions which posses the first weak derivative and which vanish at the boundary $\partial\Omega$ (cf. Appendix B.3). The resulting *weak formulation* of boundary value problem (2.3) provides additional structure which allows us to derive results on existence, uniqueness and regularity of a corresponding *weak solution*.

To derive the weak formulation of problem (2.3) the following identity is used: Let the functions $w \in C^1(\Omega)$ and $v \in C_0^\infty(\Omega)$ be given then we obtain by partial integration in the variable x_i (note that $\Omega \subset \mathbb{R}^d$ is assumed to be a bounded Lipschitz domain) the identity

$$\int_\Omega v(x)\frac{\partial w(x)}{\partial x_i}\,\mathrm{d}x = -\int_\Omega \frac{\partial v(x)}{\partial x_i} w(x)\,\mathrm{d}x \tag{2.4}$$

where it is noted that in (2.4) no boundary integral occurs since $v \in C_0^\infty(\Omega)$ and correspondingly the integrand vanishes close to the boundary $\partial\Omega$. Hence, assuming $a_{ij} \in C^1(\Omega)$ it holds for a function $u \in C^2(\Omega)$ that

$$\int_\Omega v(x)\frac{\partial}{\partial x_i}\left(a_{ij}(x)\frac{\partial u(x)}{\partial x_j}\right)\,\mathrm{d}x = -\int_\Omega \frac{\partial v(x)}{\partial x_i} a_{ij}(x)\frac{\partial u(x)}{\partial x_j}\,\mathrm{d}x. \tag{2.5}$$

If we assume that boundary value problem (2.3) has a classical solution $u \in V$, and if we multiply the identity $Lu = f$ from (2.3) by some test function $v \in C_0^\infty(\Omega)$ and integrate over Ω we obtain from (2.5) that the classical solution fulfils

$$\int_\Omega fv\,\mathrm{d}x = -\int_\Omega \operatorname{div}\big(A\nabla u\big)v\,\mathrm{d}x + \int_\Omega cuv\,\mathrm{d}x = \int_\Omega \nabla v^T A\nabla u\,\mathrm{d}x + \int_\Omega cuv\,\mathrm{d}x. \tag{2.6}$$

It follows that a function $u \in V$ is a classical solution of problem (2.3) if and only if u is a solution of the problem

$$\begin{cases} \text{find } \ u \in V \ \text{ such that} \\ a(u,v) \ = \ l(v) \quad \forall\, v \in C_0^\infty(\Omega) \end{cases} \tag{2.7}$$

with the bilinear form and the linear functional

$$a(u,v) := \int_\Omega \nabla u^T A\nabla v + cuv\,\mathrm{d}x \qquad \text{and} \qquad l(v) := \int_\Omega fv\,\mathrm{d}x. \tag{2.8}$$

Note that the assumption $a_{ij} \in C^1(\Omega)$, which is needed for the classical formulation of the boundary problem, can be weakened for problem (2.7) since $a(\cdot,\cdot)$ is also well defined for $a_{ij} \in L^\infty(\Omega)$ and $c \in L^\infty(\Omega)$. Furthermore, the bilinear form $a(\cdot,\cdot)$ and the linear functional $l(\cdot)$ are also well defined on the Sobolev space $H_0^1(\Omega)$. Correspondingly, problem (2.7) can be generalised in the following way: Instead of searching for solutions in the space $V \subset H_0^1(\Omega)$ we search for solutions in the larger and less smooth Hilbert space $H_0^1(\Omega)$, and solve the variational problem

$$\begin{cases} \text{find } \ u \in H_0^1(\Omega) \ \text{ such that} \\ a(u,v) \ = \ l(v) \quad \forall\, v \in H_0^1(\Omega) \end{cases} \tag{2.9}$$

which is called the *weak formulation* or the *variational formulation* of boundary value problem (2.3). Since the space $C_0^\infty(\Omega)$ is dense in $H_0^1(\Omega)$, and because $a(\cdot,\cdot)$ and $l(\cdot)$ are continuous (see proof of Theorem 2.5) it is equivalent to use in (2.9) test functions $v \in H_0^1(\Omega)$ instead of $v \in C_0^\infty(\Omega)$.

Definition 2.4 (Weak Solution) *A solution $u \in H_0^1(\Omega)$ of problem (2.9) is called a* weak solution *of the boundary value problem (2.3).*

Note that a weak solution is by definition an element of $H_0^1(\Omega)$, but not necessarily of $C^2(\Omega)$; and that a classical solution is by definition an element of V, but not necessarily of $H_0^1(\Omega)$. However, if there is a weak solution u fulfilling (2.9) with $u \in H_0^1(\Omega) \cap C^2(\Omega)$ and if $a_{ij} \in C^1(\Omega)$ then from (2.6) follows that $\int_\Omega (Lu - f)v \, \mathrm{d}x = 0$ for all $v \in C_0^\infty(\Omega)$. Since $C_0^\infty(\Omega)$ is dense in $L^2(\Omega)$ we have $Lu = f$, i.e., u is also a classical solution of (2.3) with $u \in V$. Vice versa, if $u \in V$ is a classical solution of (2.3) with $u \in H_0^1(\Omega) \cap V$ then it follows that u is a solution of problem (2.7) and hence a solution of (2.9), i.e., u is also a weak solution. In this sense the weak formulation (2.9) and the classical formulation (2.3) of the elliptic eigenvalue problem are equivalent.

The big advantage of the weak formulation is that it inherits additional structure which allows to derive results on existence and uniqueness of a weak solution:

Theorem 2.5 *Let the partial differential operator L in (2.2) be uniformly elliptic and*

$$a_{ij} \in L^\infty(\Omega) \text{ for } i,j = 1,\dots,d \quad \text{and} \quad c \in L^\infty(\Omega) \text{ with } c(x) \geq 0 \text{ almost everywhere in } \Omega.$$

Then the variational problem (2.9) has for every $f \in L^2(\Omega)$ a unique solution $u \in H_0^1(\Omega)$ with

$$\|u\|_1 \leq C \|f\|_0$$

where $C > 0$ is a constant independent of f.

Proof: The existence of a unique solution follows directly from the Lax-Milgram theorem (cf. Appendix A.3). First of all, it is checked if the corresponding assumptions on the linear functional $l(\cdot)$ and the bilinear form $a(\cdot,\cdot)$ are fulfilled (cf. Theorem A.18).

$l(\cdot)$ is continuous on $H_0^1(\Omega)$:

The linear functional $l(\cdot)$ in (2.8) is bounded on $H_0^1(\Omega)$ for each $f \in L^2(\Omega)$ since

$$|l(v)| = \Big| \int_\Omega f v \, \mathrm{d}x \Big| \leq \|f\|_0 \|v\|_0 \leq \|f\|_0 \|v\|_1 \quad \forall\, v \in H_0^1(\Omega).$$

$a(\cdot,\cdot)$ is continuous on $H_0^1(\Omega) \times H_0^1(\Omega)$:

For the bilinear form $a(\cdot,\cdot)$ defined in (2.8) we have for $u, v \in H_0^1(\Omega)$ that

$$\begin{aligned} |a(u,v)| &\leq \sum_{i,j=1}^n \|a_{ij}\|_{L^\infty(\Omega)} \Big\| \frac{\partial u}{\partial x_i} \Big\|_0 \Big\| \frac{\partial v}{\partial x_j} \Big\|_0 + \|c\|_{L^\infty(\Omega)} \|u\|_0 \|v\|_0 \\ &\leq \sum_{i,j=1}^n \|a_{ij}\|_{L^\infty(\Omega)} \|u\|_1 \|v\|_1 + \|c\|_{L^\infty(\Omega)} \|u\|_1 \|v\|_1 \leq C_B \|u\|_1 \|v\|_1 \end{aligned}$$

with some $C_B > 0$ independent of u and v. Hence, the bilinear form $a(\cdot,\cdot)$ is continuous.

$a(\cdot,\cdot)$ is $H_0^1(\Omega)$-elliptic:

Since the partial differential operator L in (2.2) is uniformly elliptic it follows from Definition 2.1 that there exists a constant $C_E > 0$ such that for all $\xi \in \mathbb{R}^d$ it holds

$$\xi^T A(x)\, \xi \geq C_E\, \xi^T \xi \qquad \forall\, x \in \Omega.$$

Using $\xi = \nabla u$ for $u \in H_0^1(\Omega)$, and that $c(x) \geq 0$ almost everywhere in Ω, we obtain

$$a(u,u) = \int_\Omega \nabla u^T A \nabla u \, \mathrm{d}x + \int_\Omega c u^2 \, \mathrm{d}x \geq C_E \int_\Omega \nabla u^T \nabla u \, \mathrm{d}x = C_E |u|_1^2. \tag{2.10}$$

Note that $\|\cdot\|_1$ and $|\cdot|_1$ are equivalent norms in $H_0^1(\Omega)$ since Ω is bounded (cf. Theorem B.17). Correspondingly, we obtain from (2.10) that there exists a constant $\widetilde{C}_E > 0$ such that $a(u,u) \geq \widetilde{C}_E \|u\|_1^2$ for all $u \in H_0^1(\Omega)$ which proofs the ellipticity of $a(\cdot,\cdot)$.

Since $H_0^1(\Omega)$ is a Hilbert space and the bilinear form $a(\cdot,\cdot)$ is symmetric, all assumptions of Theorem A.18 are fulfilled. Correspondingly for each $f \in L^2(\Omega)$ in (2.9) exists a unique solution $u_f \in H_0^1(\Omega)$ which allows us to define a solution operator

$$T : L^2(\Omega) \to H_0^1(\Omega) \qquad \text{with} \qquad f \mapsto u_f. \tag{2.11}$$

As in the proof of Lemma A.20 it can be shown that the operator T is continuous, in particular it holds $\|u_f\|_1 \leq \widetilde{C}_E^{-1} \|f\|_0$. ∎

Based on the analysis of the boundary value problem (2.3), in the next section the elliptic PDE eigenvalue problem (2.1) is analysed.

2.2. Analysis of the Eigenvalue Problem

In the first part of this section the weak formulation of the elliptic PDE eigenvalue problem (2.1) is derived which is nearly equivalent to the classical formulation. Similar to the previous section, the weak formulation of the eigenvalue problem inherits a special structure which allows us to derive results on the existence of weak eigensolutions. In the second part of this section the smoothness of the weak eigenfunctions is discussed.

2.2.1. Existence of Eigensolutions

The weak formulation of eigenvalue problem (2.1) is derived in the same way as the weak formulation of the boundary value problem (2.3): Assume that eigenvalue problem (2.1) possesses a classical eigensolution $u \in V \setminus \{0\}$ with eigenvalue $\lambda \in \mathbb{C}$. Multiplying $Lu = \lambda u$ by a test function $v \in C_0^\infty(\Omega)$, integration over Ω and applying partial integration (2.5) we obtain that

$$\int_\Omega \lambda u v \, \mathrm{d}x = -\int_\Omega \operatorname{div}\big(A \nabla u\big) v \, \mathrm{d}x + \int_\Omega c u v \, \mathrm{d}x = \int_\Omega \nabla v^T A \nabla u \, \mathrm{d}x + \int_\Omega c u v \, \mathrm{d}x,$$

which is leading (as in the previous section) to the variational problem

$$\begin{cases} \text{find } (\lambda, u) \in \mathbb{C} \times H_0^1(\Omega) \setminus \{0\} \text{ such that} \\ a(u,v) \;=\; \lambda\,(u,v)_0 \qquad \forall\, v \in H_0^1(\Omega) \end{cases} \tag{2.12}$$

with bilinear form $a(\cdot,\cdot)$ from (2.8) and the $L^2(\Omega)$ inner product $(u,v)_0 := \int_\Omega uv \, \mathrm{d}x$. Problem (2.12) is the weak formulation of the elliptic PDE eigenvalue problem (2.1).

The existence result presented in the following theorem is based on Corollary A.23 and Appendix A.3 where variational eigenvalue problems are discussed in a more general setting. The

key in the corresponding existence proof is to show that the solution operator $T : H_0^1(\Omega) \to H_0^1(\Omega)$ [i.e., the operator from (2.11) restricted to $H_0^1(\Omega)$] of the variational problem (2.9) is compact, which allows us to characterise the spectrum of T by the Fredholm-Riesz-Schauder theory (see Appendix A.3), which finally proves the existence of eigensolutions of the variational eigenvalue problem (2.12).

Theorem 2.6 (Existence of Eigensolutions) *Let the partial differential operator L in* (2.2) *be uniformly elliptic and*

$$a_{ij} \in L^\infty(\Omega) \text{ for } i,j = 1,\dots,d \quad \text{and} \quad c \in L^\infty(\Omega) \text{ with } c(x) \geq 0 \text{ almost everywhere in } \Omega.$$

Then the variational eigenvalue problem (2.12) *possesses a countable family of eigensolutions*

$$\big(\lambda_j, u_j\big)_{j=1}^\infty \in \mathbb{R}_{>0} \times H_0^1(\Omega) \setminus \{0\} \tag{2.13}$$

with eigenvalues λ_j ordered[2] *such that $\lambda_j \leq \lambda_{j+1}$. In particular, it holds:*

i) All eigenvalues λ_j are positive real and we have $\lambda_j \xrightarrow{j\to\infty} \infty$.

ii) The eigenspace $E(\lambda_j) \subset H_0^1(\Omega)$ of the eigenvalue λ_j, which is defined by

$$E(\lambda_j) := \operatorname{span}\Big\{ u \in H_0^1(\Omega) \;:\; a(u,v) = \lambda_j (u,v)_0 \quad \forall\, v \in H_0^1(\Omega) \Big\}, \tag{2.14}$$

is finite-dimensional.

iii) If it holds $\lambda_j \neq \lambda_k$ then the corresponding eigenfunctions u_j and u_k are orthogonal with respect to $a(\cdot,\cdot)$ and $(\cdot,\cdot)_0$, i.e., we have $a(u_j,u_k) = 0$ and $(u_j,u_k)_0 = 0$.

iv) The eigenfunctions $\big(u_j\big)_{j=1}^\infty$ form a basis of $H_0^1(\Omega)$ and without loss of generality it can be assumed that all eigenfunctions are orthonormal with respect to $a(\cdot,\cdot)$ or $(\cdot,\cdot)_0$.

Proof: In the proof of Theorem 2.5 it is shown that the bilinear form $a(\cdot,\cdot)$ is symmetric, continuous and elliptic. Furthermore, bilinear form $(\cdot,\cdot)_0$ fulfils assumption i) and ii) of Precondition A.19 since $(\cdot,\cdot)_0$ is the inner product of $H_0^1(\Omega)$. Furthermore, for all in $H_0^1(\Omega)$ bounded sequences $(u_j)_{j\in\mathbb{N}}$ there exists a subsequence $(u_{j_k})_{k\in\mathbb{N}}$ which is Cauchy w.r.t. $\|\cdot\|_0 = (\cdot,\cdot)_0^{1/2}$ since the embedding of $H_0^1(\Omega)$ in $L^2(\Omega)$ is compact, and hence assumption iii) of Precondition A.19 is fulfilled as well. Moreover, $H_0^1(\Omega)$ is a infinite-dimensional Banach space. Correspondingly, all assumptions of Corollary A.23 are satisfied and thus the statement of this theorem is proven. ■

Theorem 2.6 answers the question of the existence of eigensolutions of the elliptic PDE eigenvalue problem (2.1): If the assumptions of Theorem 2.6 are fulfilled, the weak formulation (2.12) of the eigenvalue problem possesses a countable family of weak eigensolutions $(\lambda_j, u_j)_{j=1}^\infty$. In the case that the eigenfunction $u_j \in H_0^1(\Omega)$ belongs as well to the space $C^2(\Omega)$, then the weak eigensolution (λ_j, u_j) is also a classical solution of the original PDE eigenvalue problem (2.1).

[2] Eigenvalues are repeated in (2.13) according to their geometric multiplicity (the geometric multiplicity is the dimension of the corresponding eigenspace).

Remark 2.7 (Subspace Eigenvalue Problem) *Let S be a closed subspace of $H_0^1(\Omega)$ and let the assumptions of Theorem 2.6 be fulfilled. Because of $S \subset H_0^1(\Omega)$ the bilinear forms $a(\cdot,\cdot)$ and $(\cdot,\cdot)_0$ from* (2.12) *can be restricted to $S \times S$, and hence the variational problem*

$$\begin{cases} \text{find } (\lambda^S, u^S) \in \mathbb{R} \times S \setminus \{0\} \text{ such that} \\ a(u^S, v) \;=\; \lambda^S \, (u^S, v)_0 \qquad \forall\, v \in S \end{cases} \tag{2.15}$$

is well defined. Note that S forms together with the norm of $H_0^1(\Omega)$ a Banach space since S is a closed subspace in $H_0^1(\Omega)$. Furthermore, all the properties of the bilinear forms $a(\cdot,\cdot)$ and $(\cdot,\cdot)_0$ which have been checked in the proof of Theorem 2.6 hold as well for the bilinear forms restricted to $S \times S$. As in the proof of Theorem 2.6 we conclude from Corollary A.23 that the variational eigenvalue problem (2.15) *possesses a countable family of eigensolutions*

$$\left(\lambda_j^S, u_j^S \right)_{j=1}^N \in \mathbb{R}_{>0} \times S \setminus \{0\} \qquad \text{with } \lambda_j^S \le \lambda_{j+1}^S \tag{2.16}$$

with positive eigenvalues and where $N = \dim S$. Furthermore, the eigenfunctions $(u_j^S)_{j=1}^N$ form a basis of S which can be assumed, without loss of generality, to be orthonormal with respect to $a(\cdot,\cdot)$ or $(\cdot,\cdot)_0$.

2.2.2. Regularity Results

Another important question of elliptic PDE eigenvalue problems concerns the smoothness of the eigenfunctions. In particular, one is interested in how the smoothness (in the sense of Sobolev spaces) of the eigenfunctions (2.13) is linked to the smoothness of the domain and of the coefficients of the PDE operator. These so-called *regularity properties* of the eigenfunctions are important, especially, when the variational problem (2.12) gets discretised, and a priori error estimates between the exact and the approximated eigensolutions have to be derived (cf. Section 3.2). In the following two relevant regularity results are presented.

Theorem 2.8 (Regularity I) *Consider a fixed $t \in \mathbb{N}$. Let Ω be a bounded domain with boundary of class $C^{t,1}$ and let the partial differential operator L in* (2.2) *be uniformly elliptic with $c(x) \ge 0$ almost everywhere in Ω. Moreover, it is assumed that the coefficients of the PDE operator L fulfil*

$$\begin{aligned} D^\alpha a_{ij} &\in L^\infty(\Omega) \quad \text{for all } |\alpha| \le t \text{ and all } i,j = 1,\dots,d, \\ D^\alpha c &\in L^\infty(\Omega) \quad \text{for all } |\alpha| \le t-1. \end{aligned}$$

Then the results of Theorem 2.6 become valid, and for each weak eigensolution (λ, u) of the variational problem (2.12) *it holds that $u \in H^{1+t}(\Omega)$. In particular, we have $E(\lambda) \subset H^{1+t}(\Omega) \cap H_0^1(\Omega)$.*

Proof: The results of Theorem 2.6 are valid since the assumptions made here are stronger. The regularity result $u \in H^{1+t}(\Omega)$ follows from [37, Theorem 9.1.16 and Theorem 11.1.5]. ∎

Theorem 2.8 shows how smoothness properties of the eigenfunctions are linked to the smoothness of the domain and the PDE operator, and how raising the smoothness assumptions on the data results in the regularity of the eigenfunctions in higher order Sobolev spaces. It can be even shown that the eigenfunctions of problem (2.12) are analytic if corresponding smoothness assumptions are made on the data:

Precondition 2.9 (Analytic Data) *Let L be a partial differential operator of the form* (2.2) *and $\Omega \subset \mathbb{R}^d$ the underlying domain where the following assumptions are fulfilled:*

Assumptions on the Domain

It is assumed that $\Omega \subset \mathbb{R}^d$ is a bounded Lipschitz domain with analytic boundary (cf. Definition B.10 and Definition B.11).

Assumptions on the PDE operator

L is a PDE operator of the form (2.2) *and it is assumed that:*

i) The matrix $A(x) := \big(a_{ij}(x)\big)_{i,j=1}^{d}$ is symmetric for all $x \in \Omega$ and it holds

$$0 < a_{\min} := \inf_{x\in\Omega} \max_{\xi\in\mathbb{R}^d\setminus\{0\}} \frac{\xi^T A(x)\xi}{\xi^T\xi} \;\le\; \sup_{x\in\Omega} \max_{\xi\in\mathbb{R}^d\setminus\{0\}} \frac{\xi^T A(x)\xi}{\xi^T\xi} =: a_{\max} < \infty.$$

ii) It holds $c(x) \ge 0$ for all $x \in \Omega$.

iii) The coefficients of L are infinitely differentiable, i.e., it holds

$$a_{ij} \in C^\infty(\Omega) \text{ for } i,j = 1,\dots,d \qquad \text{and} \qquad c \in C^\infty(\Omega).$$

iv) There exist constants $C_A, C_c, \gamma_A, \gamma_c \in \mathbb{R}_{>0}$ such that for all $n \in \mathbb{N}_0$ it holds

$$|A|_{n,\infty} := \left\| \left\{ \sum_{|\alpha|=n} \frac{n!}{\alpha!} |D^\alpha A|^2 \right\}^{1/2} \right\|_{L^\infty(\Omega)} \le C_A\, n!\, (\gamma_A)^n$$

$$|c|_{n,\infty} := \left\| \left\{ \sum_{|\alpha|=n} \frac{n!}{\alpha!} |D^\alpha c|^2 \right\}^{1/2} \right\|_{L^\infty(\Omega)} \le C_c\, n!\, (\gamma_c)^n$$

where $|D^\alpha A(x)|$ is defined as the spectral norm of the matrix $D^\alpha A(x) := \big(D^\alpha a_{ij}(x)\big)_{i,j=1}^{d}$.

In particular, this means that the PDE operator L is uniformly elliptic and that its coefficient functions are analytic.

The requirements of Precondition 2.9 are fulfilled, for example, when $\Omega \subset \mathbb{R}^d$ is a bounded domain with a boundary which is a graph of a polynomial function, and when L is a PDE operator of the form (2.2) with polynomial coefficients functions, where $A(x) := (a_{ij}(x))_{i,j=1}^{d}$ is symmetric and $c(x) \ge 0$ for all $x \in \Omega$, and where the eigenvalues of $A(x)$ are uniformly bounded from above and below by positive constants.

Theorem 2.10 (Regularity II) *Let the domain Ω and the partial differential operator L fulfil Precondition 2.9. Then the results of Theorem 2.6 become valid, and any eigenfunction of the variational eigenvalue problem* (2.12) *is analytic. In particular, we have $E(\lambda) \subset \cap H_0^1(\Omega)$ and each weak eigensolution of* (2.12) *is also a classical solution of the PDE eigenvalue problem* (2.1).

Proof: Note that from assumption iv) of Precondition 2.9 it follows that $c \in L^\infty(\Omega)$ and $a_{ij} \in L^\infty(\Omega)$, and hence, together with the other assumptions of this theorem, it follows that

the results of Theorem 2.6 become valid. Furthermore, in [41] and [64, Theorem 3.1] it is proven that the eigenfunctions of (2.12) are analytic. More precisely, in [64, Theorem 3.1] it is shown that $E(\lambda) \subset H^k(\Omega)$ for all $k \in \mathbb{N}$. From Sobolev's embedding theorem (cf. Theorem B.22) it follows that $E(\lambda) \subset C^k(\overline{\Omega})$ for all $k \in \mathbb{N}_0$. Since it additionally holds $E(\lambda) \subset H_0^1(\Omega)$ we conclude that $E(\lambda) \subset C^\infty(\Omega) \cap H_0^1(\Omega)$. Furthermore, because of $E(\lambda) \subset C^k(\overline{\Omega})$ for all $k \in \mathbb{N}_0$ it follows that $E(\lambda) \subset C^2(\Omega) \cap C^0(\overline{\Omega})$, i.e., each weak eigensolution (λ, u) of problem (2.12) is also a classical solution of (2.1). ■

To get a better understanding of the presented results for elliptic PDE eigenvalue problems, in the next section an example is discussed.

2.3. Laplace Eigenvalue Problem on the Unit Cube

We consider the Laplace eigenvalue problem defined on the d-dimensional unit cube

$$\begin{cases} -\Delta u = \lambda u & \text{in } \Omega = (0,1)^d, \\ u = 0 & \text{on } \partial\Omega. \end{cases} \tag{2.17}$$

The weak formulation of this eigenvalue problem is given by

$$\begin{cases} \text{find } (\lambda, u) \in \mathbb{R} \times H_0^1(\Omega) \setminus \{0\} \text{ such that} \\ \int_\Omega \nabla u^T \nabla v \, \mathrm{d}x = \lambda \int_\Omega uv \, \mathrm{d}x \quad \forall\, v \in H_0^1(\Omega). \end{cases} \tag{2.18}$$

According to Theorem 2.6 problem (2.18) has a countable family of eigensolutions (2.13) with positive eigenvalues, where the corresponding eigenfunctions form a basis of $H_0^1(\Omega)$.

Due to the rectangular structure of the domain Ω and the simple form of the PDE operator L, eigenvalue problem (2.17) is one of the rare examples where the eigensolutions can be computed analytically. This is done as follows: First we determine the eigensolutions in the one-dimensional case. For this purpose, we consider the homogeneous linear differential equation $-u''(x_1) - cu(x_1) = 0$ in $x_1 \in (0,1)$ with boundary conditions $u(0) = 0 = u(1)$ where $c > 0$ is some constant. All solutions of this problem are given by

$$u(x_1) = \begin{cases} 0 & \text{if } c \notin \{\alpha_1^2\pi^2 \,:\, \alpha_1 \in \mathbb{N}\}, \\ a\sin(\sqrt{c}x_1) & \text{if } c \in \{\alpha_1^2\pi^2 \,:\, \alpha_1 \in \mathbb{N}\} \end{cases} \qquad \text{for } x_1 \in [0,1]$$

where $a \in \mathbb{R}$ is some arbitrary constant. It follows that for $d = 1$ all eigenfunctions of the PDE problem (2.17) are given by

$$\{a\sin(\alpha_1\pi x_1) \,:\, \alpha_1 \in \mathbb{N}, a \in \mathbb{R} \setminus \{0\}\}. \tag{2.19}$$

Since the domain $\Omega = (0,1)$ and the PDE operator $L = -\Delta$ fulfil the regularity assumptions[3] of Theorem 2.10 we conclude that each weak eigensolution (λ, u) of problem (2.18) is also a classical eigensolution of (2.17) with $u \in C^\infty(\Omega) \cap H_0^1(\Omega)$, and hence all eigenfunctions of the variational problem (2.18) are given as well by (2.19). From Theorem 2.6 iii) we obtain that

[3] Note that the regularity assumptions of Theorem 2.10 on the boundary $\partial\Omega$ are only fulfilled for problem (2.18) when $d = 1$, since we only have $\partial\Omega \in C^{0,1}$.

the eigenfunctions $\{\sin(\alpha_1 \pi x_1) : \alpha_1 \in \mathbb{N}\}$ are orthogonal on $H_0^1(\Omega)$ with $\Omega = (0,1)$, and from Theorem 2.6 iv) we conclude that these orthogonal eigenfunctions form a basis of $H_0^1(\Omega)$. From [25, Chapter II, Section 1] it follows that the d-fold tensor product of the one-dimensional orthogonal basis functions forms an orthogonal basis of $H_0^1(\Omega)$ with $\Omega = (0,1)^d$, i.e., the function system $\{u_\alpha(x) : \alpha \in \mathbb{N}^d\}$, with multi-index $\alpha = (\alpha_1, \ldots, \alpha_d)$ and

$$u_\alpha(x) := \prod_{i=1}^{d} \sin(\alpha_i \pi x_i) \qquad \text{for } x = (x_1, \ldots, x_d)^T \in [0,1]^d, \tag{2.20}$$

is orthogonal and complete in $H_0^1(\Omega)$. An easy calculation shows that for all $\alpha \in \mathbb{N}^d$ the functions $u_\alpha(x)$ are eigenfunctions of the PDE problem (2.17) with the associated eigenvalues

$$\lambda_\alpha := \pi^2 \sum_{i=1}^{d} \alpha_i^2, \tag{2.21}$$

and because of $u_\alpha \in H_0^1(\Omega)$ it follows that $(\lambda_\alpha, u_\alpha)$ is also an eigensolution of (2.18). In particular, all eigensolutions of (2.18) are described by $(\lambda_\alpha, u_\alpha)_{\alpha \in \mathbb{N}^d}$. The reason is as follows: Assume that there exists an additional eigenfunction $u \in H_0^1(\Omega)$ of problem (2.18) which is linear independent to the eigenfunctions u_α with $\alpha \in \mathbb{N}^d$. Then we can assume, without loss of generality, that u is orthogonal in $H_0^1(\Omega)$ to u_α for all $\alpha \in \mathbb{N}^d$. However, this is in contradiction to the completeness of the function system $(u_\alpha)_{\alpha \in \mathbb{N}^d}$.

Note that it holds $u_\alpha \in C^\infty(\Omega) \cap H_0^1(\Omega)$, and that the eigenvalues λ_α tend to infinity (as it is claimed by Theorem 2.6) if $\alpha_i \to \infty$ for some $i \in \{1, \ldots, d\}$. Furthermore, we observe that for $d > 1$ the dimension of the eigenspaces $E(\lambda_\alpha)$ can be larger than one. For example, if in the three-dimensional case the indices α_1, α_2, $\alpha_3 \in \mathbb{N}$ are pairwise different, then there exist at least 6 linear independent eigenfunctions (e.g., $u_{(\alpha_1,\alpha_2,\alpha_3)}$, $u_{(\alpha_1,\alpha_3,\alpha_2)}$, etc.) which share the same eigenvalue $\lambda = \pi^2(\alpha_1^2 + \alpha_2^2 + \alpha_3^2)$.

3. Solving Elliptic PDE Eigenvalue Problems

In contrast to the example given in (2.18), in general the variational eigenvalue problem (2.12) cannot be solved analytically, it has to be solved numerically. For this purpose the Ritz-Galerkin discretisation is introduced: In the Ritz-Galerkin discretisation the variational eigenvalue problem (2.12) gets discretised by the use of some finite dimensional subspace, and an algebraic eigenvalue problem is obtained. Depending on the properties of the used subspace, the discrete eigenpairs of the algebraic problem are approximating the sought eigensolutions of the variational problem (2.12).

The remainder of this chapter is organised as follows: In Section 3.1 the Ritz-Galerkin discretisation is described, and in Section 3.2 corresponding approximation properties are discussed and general error estimates are presented. In Section 3.3 the well-known finite element spaces are recalled, which are based on a triangulation of the domain Ω with some underlying mesh width $h > 0$ and which consist of piecewise polynomial functions of degree $p \in \mathbb{N}$. These finite element spaces are very well suited, and widely used in practice, as a concrete choice for the underlying subspace of the Ritz-Galerkin discretisation. The Ritz-Galerkin discretisation combined with the finite element space is called finite element discretisation, and in Section 3.4 corresponding a priori error estimates for the eigenvalue and eigenfunction approximation are presented which are explicit in the underlying discretisation parameters h and p. Furthermore, the important issue is discussed how many eigenvalues and eigenfunctions can be well approximated by a given finite element space.

3.1. Ritz-Galerkin Discretisation of the Eigenvalue Problem

The basic idea of the Ritz-Galerkin discretisation is to replace the infinite-dimensional space $H_0^1(\Omega)$ of the variational problem (2.12) by some suitable finite-dimensional subspace V_h with

$$V_h \subset H_0^1(\Omega) \quad \text{and} \quad N_h := \dim V_h < \infty.$$

The index $h > 0$ used for the subspace V_h will get relevant in the next section where h is used to indicate a family of finite-dimensional subspaces $(V_h)_{h>0}$ which are approximating the underlying Sobolev space $H_0^1(\Omega)$ for $h \to 0$, and where h will refer to some mesh width parameter.

Since it holds $V_h \subset H_0^1(\Omega)$ the bilinear forms $a(\cdot,\cdot)$ and $(\cdot,\cdot)_0$ from (2.12) can be restricted to $V_h \times V_h$, and hence the finite-dimensional variational problem

$$\begin{cases} \text{find } (\lambda^{(h)}, u^{(h)}) \in \mathbb{R} \times V_h \setminus \{0\} \text{ such that} \\ a(u^{(h)}, v) \;=\; \lambda^{(h)}\,(u^{(h)}, v)_0 \qquad \forall\, v \in V_h \end{cases} \tag{3.1}$$

is well defined. The transition of the variational eigenvalue problem (2.12) to the eigenvalue problem (3.1) is called *Ritz-Galerkin discretisation* or short *Galerkin discretisation*. In literature sometimes problem (3.1) itself is also referred to as the Ritz-Galerkin discretisation of problem

(2.12). Furthermore, the Galerkin discretisation is called *conforming* since the space V_h is a subset of $H_0^1(\Omega)$.

Since the subspace $V_h \subset H_0^1(\Omega)$ is finite-dimensional, it follows that V_h is closed, and we conclude from Remark 2.7 that under the assumptions of Theorem 2.6 the discrete variational eigenvalue problem (3.1) has N_h eigensolutions of the form

$$\big(\lambda_j^{(h)}, u_j^{(h)}\big)_{j=1}^{N_h} \in \mathbb{R}_{>0} \times V_h \setminus \{0\} \qquad \text{with } \lambda_j^{(h)} \le \lambda_{j+1}^{(h)} \tag{3.2}$$

where all eigenvalues are positive. The eigensolutions (3.2) are called *Ritz-Galerkin eigensolutions* (associated to subspace V_h) of eigenvalue problem (2.12). In this context, we refer to (3.1) and $(\lambda_j^{(h)}, u_j^{(h)})$ as the discrete problem and the discrete eigensolution, and refer to (2.12) and (λ_j, u_j) as the continuous problem and the continuous eigensolution. Furthermore, the eigenspaces for the discrete eigenvalues $\lambda_j^{(h)}$ are defined by

$$E_h\big(\lambda_j^{(h)}\big) := \operatorname{span}\Big\{u \in V_h \;:\; a(u,v) = \lambda_j^{(h)}(u,v)_0 \quad \forall\, v \in V_h\Big\}.$$

For the actual computation of the discrete eigensolutions a basis of the subspace V_h is needed which shall be given in the following by $\big(\varphi_i^{(h)}\big)_{i=1}^{N_h}$. Using this basis each $v^{(h)} \in V_h$ can be uniquely represented by an suitable coefficient vector $x^{(h)} \in \mathbb{R}^{N_h}$ via

$$v^{(h)} = \sum_{i=1}^{N_h} x_i^{(h)}\, \varphi_i^{(h)} \qquad \text{and} \qquad x^{(h)} := (x_1^{(h)}, \dots, x_{N_h}^{(h)})^T \in \mathbb{R}^{N_h}.$$

The above representation leads to the so-called *prolongation operator*

$$\mathcal{P} : \mathbb{R}^{N_h} \to V_h \subset H_0^1(\Omega) \quad \text{with} \quad x^{(h)} \mapsto \sum_{i=1}^{N_h} x_i^{(h)}\, \varphi_i^{(h)} \tag{3.3}$$

which is bijective. Representing $u^{(h)}$ in (3.1) by its coefficient vector $x^{(h)}$ — i.e., it holds $u^{(h)} = \mathcal{P}x^{(h)}$ — we can reformulate the Ritz-Galerkin discretisation (3.1) as

$$\begin{cases} \text{find } \; (\lambda^{(h)}, x^{(h)}) \in \mathbb{R} \times \mathbb{R}^{N_h} \setminus \{0\} \; \text{ such that} \\ \sum_{j=1}^{N_h} a(\,\varphi_j^{(h)}, \varphi_i^{(h)}\,)\, x_j^{(h)} \;=\; \lambda^{(h)} \sum_{j=1}^{N_h} (\,\varphi_j^{(h)}, \varphi_i^{(h)}\,)_0\, x_j^{(h)} \qquad \forall\, i = 1, \dots, N_h. \end{cases}$$

This finally leads to the generalised algebraic eigenvalue problem

$$\begin{cases} \text{find } (\lambda^{(h)}, x^{(h)}) \in \mathbb{R} \times \mathbb{R}^{N_h} \setminus \{0\} \text{ with} \\ K^{(h)}\, x^{(h)} \;=\; \lambda^{(h)}\, M^{(h)}\, x^{(h)} \end{cases} \tag{3.4}$$

with the symmetric and positive definite matrices

$$K^{(h)} := \Big(a(\varphi_j^{(h)}, \varphi_i^{(h)})\Big)_{i,j=1}^{N_h} \in \mathbb{R}^{N_h \times N_h} \quad \text{and} \quad M^{(h)} := \Big((\varphi_j^{(h)}, \varphi_i^{(h)})_0\Big)_{i,j=1}^{N_h} \in \mathbb{R}^{N_h \times N_h}. \tag{3.5}$$

The matrix $K^{(h)}$ is called *system matrix* or *stiffness matrix*, and $M^{(h)}$ is called *mass matrix*. The eigenvalue problems (3.4) and (3.1) are equivalent in the sense that

$$\big(\lambda^{(h)}, x^{(h)}\big) \text{ is an eigenpair of (3.4)} \iff \big(\lambda^{(h)}, \mathcal{P}x^{(h)}\big) \text{ is an eigenpair of (3.1).} \tag{3.6}$$

3.2. Approximation Properties of the Ritz-Galerkin Discretisation

The aim of the Ritz-Galerkin discretisation is to approximate the sought eigensolutions (λ, u) of the continuous problem (2.12) by the eigensolutions $(\lambda^{(h)}, u^{(h)})$ of the discrete problem (3.1), i.e., that the corresponding discretisation errors $\lambda - \lambda^{(h)}$ and $u - u^{(h)}$ of the eigenvalues and eigenfunctions shall tend to zero. For this purpose, a family of finite-dimensional subspaces $(V_h)_{h>0} \subset H_0^1(\Omega)$ is needed which is approximating the underlying Sobolev space $H_0^1(\Omega)$ in the sense that

$$\lim_{h\to 0} \inf_{v^{(h)}\in V_h} \|u - v^{(h)}\|_1 = 0 \qquad \text{for all } u \in H_0^1(\Omega). \tag{3.7}$$

In the following basic approximation properties of the Ritz-Galerkin discretisation are presented, and it is discussed under which assumptions the discrete eigenvalues and eigenfunctions of problem (3.1) converge to its continuous counterparts of problem (2.12).

Remark 3.1 *The notation $(V_h)_{h>0}$ which is used to indicate a family of subspaces is an abbreviation and has to be interpreted as follows: We consider a family of subspaces $(V_h)_{h\in\mathcal{U}}$ where $\mathcal{U}$ is some arbitrary subset of $(0,\infty)$ where 0 is an accumulation point of $\mathcal{U}$.*

The first observation which can be made in the analysis of the Ritz-Galerkin discretisation is that the infinitely many eigensolutions of the continuous problem (2.12) are faced only with finitely many eigensolutions of the discrete problem (3.1). Correspondingly a uniform approximation of all eigensolutions (λ, u) of the continuous problem by eigensolutions $(\lambda^{(h)}, u^{(h)})$ of the discrete problem is not possible. It is only possible to characterise a fixed eigenvalue λ and a fixed eigenfunction u of (2.12) as an accumulation point of discrete eigenvalues $\{\lambda^{(h)} : h > 0\}$ and discrete eigenfunctions $\{u^{(h)} : h > 0\}$, and to estimate the corresponding approximation errors. However, before corresponding convergence results are discussed in detail, certain variational principles are presented which allow to characterise the eigenvalues and eigenfunctions of the continuous and discrete problem as various extrema of the *Rayleigh quotient*

$$R(u) := a(u,u) \,/\, (u,u)_0 \qquad \text{with } u \in H_0^1(\Omega) \setminus \{0\}.$$

Throughout the whole section we will assume that the eigensolutions of the continuous problem (2.12) and of the discrete problem (3.1) are ordered as in (2.13) and (3.2) by the size and the geometric multiplicity of the eigenvalues, i.e., it holds

$$\begin{aligned} \big(\lambda_j, u_j\big)_{j=1}^{\infty} &\in \mathbb{R}_{>0} \times H_0^1(\Omega) \setminus \{0\} && \text{with } \lambda_j \le \lambda_{j+1} \text{ for } j \in \mathbb{N}, \\ \big(\lambda_j^{(h)}, u_j^{(h)}\big)_{j=1}^{N_h} &\in \mathbb{R}_{>0} \times V_h \setminus \{0\} && \text{with } \lambda_j^{(h)} \le \lambda_{j+1}^{(h)} \text{ for } j = 1, \dots, N_h - 1. \end{aligned}$$

Arranging the eigensolutions in this order enables the following well-known variational principles which can be found for example in [5] or [7].

Minimum-Maximum Principle:

$$\lambda_j = \min_{\substack{H_j \subset H_0^1(\Omega) \\ \text{with } \dim H_j = j}} \max_{u \in H_j \setminus \{0\}} R(u) \qquad = \max_{u \in \operatorname{span}\{u_1,\dots,u_j\}\setminus\{0\}} R(u) \qquad \text{for } j = 1,2,3,\dots,$$

$$\lambda_j^{(h)} = \min_{\substack{H_j^{(h)} \subset V_h \\ \text{with } \dim H_j^{(h)} = j}} \max_{u \in H_j^{(h)} \setminus \{0\}} R(u) \quad = \max_{u \in \operatorname{span}\{u_1^{(h)},\dots,u_j^{(h)}\}\setminus\{0\}} R(u) \qquad \text{for } j = 1,\dots,N_h.$$

Minimum Principle:

If it is additionally assumed (without loss of generality) that the eigenfunctions of the continuous and discrete problem are chosen orthonormal with respect to $(\cdot,\cdot)_0$, i.e., we have

$$\begin{aligned} a(u_i,u_j) \;&=\; \lambda_j (u_i,u_j)_0 \;&=& \lambda_j \delta_{ij} && \text{for all } i,j \in \mathbb{N} \\ a\big(u_i^{(h)},u_j^{(h)}\big) \;&=\; \lambda_j^{(h)} \big(u_i^{(h)},u_j^{(h)}\big)_0 \;&=& \lambda_j^{(h)} \delta_{ij} && \text{for all } i,j = 1,\dots,N_h, \end{aligned}$$

then it holds

$$\lambda_1 = \min_{u \in H_0^1(\Omega)\setminus\{0\}} R(u) = R(u_1), \qquad \lambda_j = \min_{\substack{u \in H_0^1(\Omega)\setminus\{0\}, \\ a(u,u_i)=0 \\ \text{for } i=1,\dots,j-1}} R(u) = R(u_j) \qquad \text{for } j = 2,3,4,\dots,$$

$$\lambda_1^{(h)} = \min_{u \in V_h \setminus \{0\}} R(u) = R\big(u_1^{(h)}\big), \qquad \lambda_j^{(h)} = \min_{\substack{u \in V_h\setminus\{0\}, \\ a(u,u_i^{(h)})=0 \\ \text{for } i=1,\dots,j-1}} R(u) = R\big(u_j^{(h)}\big) \qquad \text{for } j = 2,\dots,N_h.$$

From the Minimum-Maximum principle directly follows that

$$\lambda_j \le \lambda_j^{(h)} \qquad \text{for } j = 1,\dots,N_h.$$

Furthermore, the Minimum-Maximum principle helps to prove the following convergence result of the Ritz-Galerkin discretisation.

Theorem 3.2 (Qualitative Convergence Results) *Let the assumptions of Theorem 2.6 be valid and let the family of finite-dimensional subspaces $(V_h)_{h>0} \subset H_0^1(\Omega)$ fulfil approximation property (3.7). Then it holds:*

i) The discrete eigenvalues are approximating the continuous ones, i.e., it holds

$$\lambda_j^{(h)} \xrightarrow{h\to 0} \lambda_j \qquad \text{for all } j \in \mathbb{N}.$$

ii) Consider for some fixed $j \in \mathbb{N}$ a sequence of discrete eigenfunctions $\big(u_j^{(h_i)}\big)_{i\in\mathbb{N}}$ with $u_j^{(h_i)} \in E_{h_i}\big(\lambda_j^{(h_i)}\big)$ and $\|u_j^{(h_i)}\|_1 = 1$ where $h_i \xrightarrow{i\to\infty} 0$. Then there exists a subsequence $\big(u_j^{(h_{i_k})}\big)_{k\in\mathbb{N}}$ which converges in $H_0^1(\Omega)$ to an eigenfunction $\widetilde{u}_j \in E(\lambda_j)$ and we have

$$\|\widetilde{u}_j - u_j^{(h_{i_k})}\|_1 \xrightarrow{k\to\infty} 0 \qquad \text{and} \qquad \|\widetilde{u}_j\|_1 = 1.$$

Proof: The result for the convergence of the eigenvalues follows directly from the Minimum-Maximum principle, the approximation property (3.7) of V_h, and the continuity of the Rayleigh quotient $R(\cdot)$. The result for the convergence of the eigenfunctions is a combination of [37, Theorem 11.2.11] and the convergence result for the eigenvalues described in i). ■

In the following the convergence rate of the eigenvalue and eigenfunction approximation is discussed. Note that $\dim E(\lambda_j) = \dim E_h(\lambda_j^{(h)})$ does not necessarily need to hold. In the case that $\dim E(\lambda_j) > 1$ it can happen that the multiple eigenvalue λ_j gets approximated by several discrete eigenvalues which differ from each other. However, if the eigenvalue λ_j is simple (i.e., it holds $\dim E(\lambda_j) = 1$) then the eigenvalues $\lambda_j^{(h)}$ are the only discrete eigenvalues that are converging to λ_j for $h \to 0$ and it holds $\dim E_h(\lambda_j^{(h)}) = 1$ when h is sufficiently small. This issue eases the error analysis of simple eigenvalues and typically leads to better error estimates than when λ_j is a multiple eigenvalue. For this reason, in literature the error analysis is often restricted to the case of simple eigenvalues λ_j which will be done here as well. For error estimates associated to multiple eigenvalues it is referred, for example, to [7].

Theorem 3.3 (Quantitative Convergence Results) *Let the assumptions of Theorem 2.6 be valid and let the family of finite-dimensional subspaces $(V_h)_{h>0} \subset H_0^1(\Omega)$ fulfil approximation property (3.7). Furthermore, let (λ_j, u_j) be a continuous eigensolution with $\dim E(\lambda_j) = 1$ and $\|u_j\|_1 = 1$ with some fixed $j \in \mathbb{N}$. Then there exists a constant $C > 0$ and discrete eigenfunctions $\widetilde{u}_j^{(h)} \in E_h(\lambda_j^{(h)})$ such that for all $h > 0$ sufficiently small it holds:*

$$i) \qquad |\lambda_j - \lambda_j^{(h)}| \;\leq\; C \inf_{v^{(h)} \in V_h} \|u_j - v^{(h)}\|_1^2,$$

$$ii) \qquad \|u_j - \widetilde{u}_j^{(h)}\|_1 \;\leq\; C \inf_{v^{(h)} \in V_h} \|u_j - v^{(h)}\|_1.$$

Proof: The error estimates follow directly from [37, Theorem 11.2.19 and Theorem 11.2.20] where it is noted that e^* in [37, Theorem 11.2.19] has been chosen to $e^* := e/\|e\|_0^2$. Furthermore, it is noted that the eigenvalues $\lambda_j^{(h)}$ are the only discrete eigenvalues that converge to λ_j for $h \to \infty$ since $\dim E(\lambda_j) = 1$, and that according to Theorem A.16 the index of λ_j is equal to 1. Correspondingly all assumptions of [37, Theorem 11.2.19 and Theorem 11.2.20] are fulfilled. ■

As it can be seen in Theorem 3.3 the errors of the eigenvalue and the eigenfunction approximation depend both on $\inf_{v^{(h)} \in V_h} \|u_j - v^{(h)}\|_1$, i.e., on the approximability of the continuous eigenfunction u_j by the underlying subspace V_h of the Ritz-Galerkin discretisation. In practice typically the finite element spaces are used as a concrete choice for V_h, where the discretisation parameter h refers to the mesh width of the corresponding triangulation (cf. Section 3.3). A priori error estimates of the Ritz-Galerkin discretisation, which are explicit in the underlying mesh width h of the used finite element space V_h, can then easily be derived by combining appropriate regularity results of the continuous eigenfunctions (which describe their smoothness) with corresponding approximation properties of the finite element space. For example, assuming that V_h fulfils the approximation property

$$\inf_{v^{(h)} \in V_h} \|u - v^{(h)}\|_1 \;\leq\; C\, h\, \|u\|_{H^2(\Omega)} \qquad \text{for all } u \in H^2(\Omega) \tag{3.8}$$

with some constant $C > 0$ independent of h (which is valid, e.g., for the finite element space $V_h = \mathbb{X}^1_{h,0}$ from Theorem 3.11), and combining this with the regularity result of Theorem 2.8 one obtains from Theorem 3.3 the following error estimates:

Corollary 3.4 *Let the assumptions of Theorem 2.8 be valid for $t = 1$ and let the family of finite-dimensional subspaces $(V_h)_{h>0} \subset H^1_0(\Omega)$ fulfil approximation property (3.8). Furthermore, let (λ_j, u_j) be a continuous eigensolution with $\dim E(\lambda_j) = 1$ and $\|u_j\|_1 = 1$ with some fixed $j \in \mathbb{N}$. Then there exists a constant $C > 0$ and discrete eigenfunctions $\widetilde{u}_j^{(h)} \in E_h(\lambda_j^{(h)})$ such that for all $h > 0$ sufficiently small it holds*

$$|\lambda_j - \lambda_j^{(h)}| \;\leq\; C\,h^2 \qquad \text{and} \qquad \|u_j - \widetilde{u}_j^{(h)}\|_1 \;\leq\; C\,h.$$

For the error estimate of $u_j - u_j^{(h)}$ better results are obtained in the $L^2(\Omega)$-norm:

Theorem 3.5 *Let the same assumptions as in Corollary 3.4 be fulfilled. Then there exists a constant $C > 0$ and discrete eigenfunctions $\widetilde{u}_j^{(h)} \in E_h(\lambda_j^{(h)})$ such that for all $h > 0$ sufficiently small it holds*

$$\|u_j - \widetilde{u}_j^{(h)}\|_0 \;\leq\; C\,h^2.$$

Proof: The error estimate follows directly from [37, Theorem 11.2.22]. Note that the eigenvalues $\lambda_j^{(h)}$ are the only discrete eigenvalues that converge to λ_j for $h \to \infty$ since $\dim E(\lambda_j) = 1$, and that according to Theorem A.16 the index of λ_j is equal to 1. Furthermore, the needed H^2-regularity of the bilinear form $a(\cdot,\cdot)$ follows from [37, Theorem 9.1.16], and $E(\lambda_j) \subset H^2(\Omega)$ follows from Theorem 2.8. Correspondingly, all assumptions which are made in [37, Theorem 11.2.22] are fulfilled. ■

Note that the constant C which is involved in the error estimates of Theorem 3.3 – Theorem 3.5 in general depends on each individual eigenvalue λ_j. Furthermore, the discretisation parameter h has to be sufficiently small in order that the corresponding convergence rates for $\lambda_j^{(h)}$ and $u_j^{(h)}$ become valid. This means that in practice these convergence rates get only visible when $h \leq h_{\max}$ where $h_{\max} > 0$ is fixed maximal mesh width which depends on the size of the corresponding eigenvalue λ_j and its spectral separation. This issue is discussed in detail in Section 3.4 for the well-known finite element spaces which are typically used in practice as concrete choice for the subspace V_h. But first of all, the finite element spaces are briefly recalled in the following section.

3.3. Introduction of Finite Element Spaces

To apply the Ritz-Galerkin discretisation and to compute the discrete eigensolutions $(\lambda^{(h)}, x^{(h)})$ [respectively the eigensolutions $(\lambda^{(h)}, u^{(h)})$] of the algebraic eigenvalue problem $(K^{(h)}, M^{(h)})$ from (3.4), a basis $\big(\varphi_i^{(h)}\big)_{i=1}^{N_h}$ of the underlying subspace V_h is needed. Using an arbitrary basis, however, might be disadvantageous, since in that case the stiffness matrix $K^{(h)} \in \mathbb{R}^{N_h \times N_h}$ and the mass matrix $M^{(h)} \in \mathbb{R}^{N_h \times N_h}$ from (3.5) are possibly dense, i.e., that for the most of the

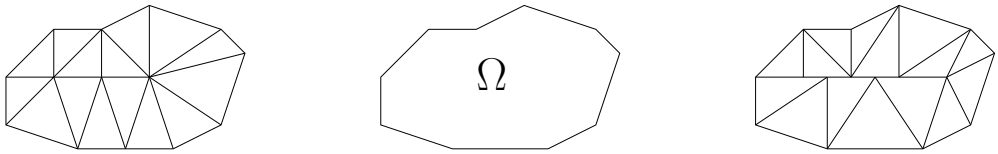


Figure 3.1.: Triangulation of a polygonal domain (middle): admissible (left), non-admissible (right) with so-called *hanging nodes* (i.e., vertices of triangles lay on edges of other triangles).

indices $i, j = 1, \ldots, N^{(h)}$ the matrix entries

$$(K^{(h)})_{ij} \;=\; a(\,\varphi_j^{(h)}, \varphi_i^{(h)}\,) \;=\; \int_\Omega (\nabla\varphi_j^{(h)})^T\, A\,(\nabla\varphi_i^{(h)}) \,+\, c\,\varphi_j^{(h)}\,\varphi_i^{(h)}\;\mathrm{d}x, \tag{3.9}$$

$$(M^{(h)})_{ij} \;=\; (\,\varphi_j^{(h)}, \varphi_i^{(h)}\,)_0 \;=\; \int_\Omega \varphi_j^{(h)}\,\varphi_i^{(h)}\;\mathrm{d}x \tag{3.10}$$

are different to zero. A dense matrix structure is costly: For example, the matrix-vector multiplication $K^{(h)}x^{(h)}$ results in costs of the order $\mathcal{O}(N^2)$ with $N := N_h$. Correspondingly, if $N^{(h)}$ is getting large the Ritz-Galerkin discretisation with a general basis is getting unfeasible, and hence a basis is needed such that $K^{(h)}$, $M^{(h)}$ become sparse. The following observation shows how the basis functions have to be chosen:

Lemma 3.6 *Denote $I_i := \mathrm{supp}(\varphi_i^{(h)}) \setminus \partial\mathrm{supp}(\varphi_i^{(h)})$ the interior of the basis function $\varphi_i^{(h)}$ then the matrix entries $K_{ij}^{(h)}$ and $M_{ij}^{(h)}$ are zero if it holds $I_i \cap I_j = \emptyset$.*

Proof: Since the integration in (3.9) and (3.10) can be reduced to $I_i \cap I_j$ the matrix entries $K_{ij}^{(h)}$ and $M_{ij}^{(h)}$ become zero in the case that $I_i \cap I_j = \emptyset$. ∎

The finite element spaces, which are discussed in the following, are based on the observation made in Lemma 3.6. The finite element spaces are spanned by basis functions with a small local support and are in general very well suited, and widely used in practice, for the Ritz-Galerkin discretisation of scalar elliptic partial differential equations, and hence for the discretisation of eigenvalue problem (2.12). The Ritz-Galerkin discretisation using finite element spaces for V_h is simply called *finite element method* (short FEM) or *finite element discretisation*. In the following main principles of finite element spaces are summarised and approximation properties are presented. For ease of representation the discussion is restricted to the very important class of *simplicial finite element spaces*. More information on finite elements can be found, e.g., in [23], [24] and [37].

3.3.1. Simplicial Finite Elements

The construction of finite element spaces and its basis functions starts with the subdivision of the domain Ω into a finite number of small subsets $T_i \subset \Omega$ for $i = 1, \ldots, t$ with some $t \in \mathbb{N}$. These small subsets T_i are called *finite elements* and are the foundation of the corresponding basis functions respectively their supports. The subdivision of Ω is called *triangulation*[1] and is denoted by $\mathcal{T} = \{T_1, \ldots, T_t\}$. In the following it is assumed that the triangulation consists of simplices with the following properties.

[1] The term triangulation is used independently of the underlying spatial dimension d and independently of the concrete shape of T_i, i.e., also when in the case of $d = 3$ cubes are used for $T_i \subset \Omega$.

Definition 3.7 (Admissible Triangulation) *A triangulation $\mathcal{T} = \{T_1, \dots T_t\}$ of the domain $\Omega \subset \mathbb{R}^d$ is called admissible if it holds:*

i) Each $T_i \subset \mathbb{R}^d$ is an open d-simplex (i.e., an interval, a triangle or a tetrahedron for $d = 1, 2$ or 3),

ii) $\overline{\Omega} = \bigcup_{i=1}^{t} \overline{T}_i$ with $T_i \cap T_j = \emptyset$ for $i \neq j$,

iii) Every vertex, edge, face of $\overline{T}_i$ is either a subset of $\partial\Omega$ or a vertex, edge, face of another $\overline{T}_j$ with $i \neq j$.

According to Definition 3.7 an admissible triangulation is only possible when $\Omega \subset \mathbb{R}^d$ is a polytope[2]. For the sake of simplicity, we assume in the following that Ω has such a shape and that $d \leq 3$.

3.3.2. Simplicial Finite Element Spaces

In order to construct a family of finite element spaces $(V_h)_{h>0}$ [cf. Remark 3.1] which is approximating the Sobolev space $H_0^1(\Omega)$ [as needed in (3.7) for the Ritz-Galerkin discretisation] a family of admissible triangulations $\{\mathcal{T}_h\}_{h>0}$ is needed. The discretisation parameter h of the triangulations $\mathcal{T}_h$ is chosen in the following such that h equals the value $h(\mathcal{T}_h)$ where

$$h(\mathcal{T}_h) := \max\big\{ h_T \,:\, T \in \mathcal{T}_h \big\} \qquad \text{and} \quad h_T := \operatorname{diam}(T),$$

i.e., the parameter h explicitly refers to the mesh width of $\mathcal{T}_h$. The finite element spaces are defined as follows:

Definition 3.8 (Finite Element Space) *Let $\mathcal{T}_h$ be an admissible triangulation of Ω consisting only of d-simplices (cf. Definition 3.7). Then the associated simplicial finite element spaces are defined by*

$$\begin{aligned} \mathbb{X}_h^0 \;&:=\; \mathbb{X}_{\mathcal{T}_h}^0 \;:= \Big\{ v \in L^2(\Omega) \,:\, v_{|T} \in \mathbb{P}_0 \text{ for all } T \in \mathcal{T}_h \Big\}, \\ \mathbb{X}_h^p \;&:=\; \mathbb{X}_{\mathcal{T}_h}^p \;:= \Big\{ v \in C^0(\overline{\Omega}) \,:\, v_{|T} \in \mathbb{P}_p \text{ for all } T \in \mathcal{T}_h \Big\} \quad \text{for } p \geq 1, \end{aligned}$$

where $\mathbb{P}_p$ is the space of polynomials in $\mathbb{R}^d$ of total degree $p \in \mathbb{N}_0$ which is consisting of all functions $v : \mathbb{R}^d \to \mathbb{R}$ of the form

$$v(x) = \sum_{|\alpha| \leq p} c_\alpha \, x_1^{\alpha_1} \dots x_d^{\alpha_d} \qquad \text{for } x \in \mathbb{R}^d \text{ with } \alpha \in \mathbb{N}_0^d \text{ and } c_\alpha \in \mathbb{R}.$$

Hence, the spaces $\mathbb{X}_h^p$ consist of piecewise polynomials of total degree $p \in \mathbb{N}_0$ which are continuous for $p \geq 1$. The relation between $\mathbb{X}_h^p$ and Sobolev space $H^1(\Omega)$ is as follows:

Theorem 3.9 *For $p \geq 1$ it holds $\mathbb{X}_h^p \subset H^1(\Omega)$.*

[2] A polytope is a bounded set of points in $\mathbb{R}^d$ that has flat sides, i.e., for $d = 2$ it is a polygon and for $d = 3$ it is a polyhedron.

Proof: This result follows directly from [24, Theorem 5.1]. ■

For the Ritz-Galerkin discretisation of eigenvalue problem (2.12) a finite-dimensional subspace $V_h \subset H_0^1(\Omega)$ is required. Theorem 3.9 guarantees the inclusion $\mathbb{X}_h^p \subset H^1(\Omega)$ for $p \geq 1$, however, a subspace V_h is needed consisting of functions that are zero on $\partial\Omega$. For this purpose the following finite element spaces are introduced

$$\mathbb{X}_{h,0}^p := \left\{ v \in \mathbb{X}_h^p \,:\, v_{|\partial\Omega} = 0 \right\} = \mathbb{X}_h^p \cap H_0^1(\Omega) \qquad \text{for } p \geq 1. \tag{3.11}$$

3.3.3. Approximation Properties of Finite Element Spaces

Using $V_h = \mathbb{X}_{h,0}^p$ for the Ritz-Galerkin discretisation of eigenvalue problem (2.12), one is interested if approximation property (3.7) is fulfilled, and motivated by Theorem 3.3, how well the eigenfunctions $u \in H_0^1(\Omega)$ of the continuous problem (2.12) can be approximated by the finite element space $\mathbb{X}_{h,0}^p$. The approximability of u depends on its smoothness which can be guaranteed, e.g., by the regularity result of Theorem 2.8: Depending on certain smoothness assumptions on the data it holds $u \in H^k(\Omega)$ for some suitable $k \geq 2$. Thus, one is interested in error estimates between the spaces $\mathbb{X}_{h,0}^p$ and $H^k(\Omega)$, which are typically derived by constructing an interpolation operator $I_h^p : C^0(\overline{\Omega}) \to \mathbb{X}_h^p$ and by estimating the approximation error of $v - I_h^p v$ for $v \in C^0(\overline{\Omega})$ (see, e.g., [24]). Because of the Sobolev's embedding theorem (Theorem B.22) it holds $H^k(\Omega) \subset C^0(\overline{\Omega})$ for $k \geq 2$ and $d \leq 3$, and hence the error estimates of $v - I_h^p v$ are also valid for $v \in H^k(\Omega)$. The described approach results in error estimates which are summarised in Theorem 3.11, however, an additional assumption has to be made on the triangulations associated to the finite element spaces:

Definition 3.10 (Regular Family of Triangulations) *A family of admissible triangulations $\{\mathcal{T}_h\}_{h>0}$ is called regular ([24, Section 17]) if it holds:*

i) There exists a constant $\sigma > 0$ such that

$$\frac{h_T}{\rho_T} \leq \sigma \qquad \text{for all } T \in \mathcal{T}_h \text{ and all } \mathcal{T}_h \in \{\mathcal{T}_h\}_{h>0}$$

where $\rho_T := \sup\{\operatorname{diam}(B) : B \text{ is a ball with } B \subset T\}$.

ii) It holds $\inf\{h(\mathcal{T}) : \mathcal{T} \in \{\mathcal{T}_h\}_{h>0}\} = 0$, i.e., the mesh width of the triangulations approaches zero.

Theorem 3.11 (Approximation Properties) *Let $\Omega \subset \mathbb{R}^d$ be a bounded polytopal domain and $d \leq 3$. Furthermore, let $\{\mathcal{T}_h\}_{h>0}$ be a regular family of triangulations of Ω consisting only of d-simplices with $h = h(\mathcal{T}_h)$, and let $\mathbb{X}_h^p$ and $\mathbb{X}_{h,0}^p$ be the associated finite element spaces with $p \geq 1$. Then there exists a constant $C > 0$ independent of h such that for $t \in \{0, 1\}$ and $k \in \mathbb{N}$ with $2 \leq k \leq p+1$ it holds*

$$\inf_{v^{(h)} \in \mathbb{X}_h^p} \|u - v^{(h)}\|_t \leq C h^{k-t} |u|_k \qquad \text{for all } u \in H^k(\Omega)$$

$$\inf_{v^{(h)} \in \mathbb{X}_{h,0}^p} \|u - v^{(h)}\|_t \leq C h^{k-t} |u|_k \qquad \text{for all } u \in H^k(\Omega) \cap H_0^1(\Omega).$$

Proof: The result for $k = p+1$ follows directly from [24, Theorem 17.1] and by using Sobolev's embedding theorem. The results for $k < p+1$ follow from the error estimate associated to the polynomial degree $\widetilde{p} := k - 1$ and using the relation $\mathbb{X}_h^{\widetilde{p}} \subset \mathbb{X}_h^p$. ■

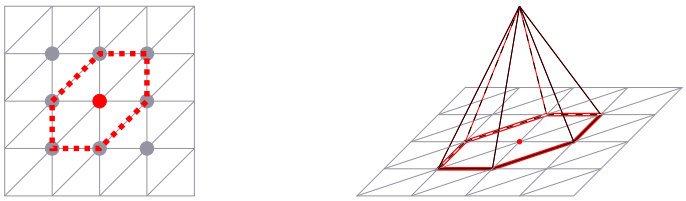


Figure 3.2.: Triangulation of the unit square with 9 inner nodal points that are associated to the finite element space $\mathbb{X}^1_{h,0}$. Furthermore, the nodal basis function $\varphi_i^{(h)} \in \mathbb{X}^p_{h,0}$, associated to the nodal point in the centre, has been illustrated (right) and its support (left).

3.3.4. The Nodal Basis of the Finite Element Space

Last but not least a basis $\left(\varphi_i^{(h)}\right)_{i=1}^{N_h}$ of $\mathbb{X}^p_{h,0}$ is needed such that the stiffness matrix $K^{(h)}$ and the mass matrix $M^{(h)}$ become sparse. Such a basis is derived as follows: Each triangulation $\mathcal{T}_h$ of Ω is associated with certain nodal points $\tilde{b}_i^{(h)} \in \overline{\Omega}$ with $i = 1, \ldots, \tilde{N}_h$ such that each function $v^{(h)} \in \mathbb{X}^p_h$ is uniquely described by the point values $v^{(h)}(\tilde{b}_i^{(h)})$ for $i = 1, \ldots, \tilde{N}_h$. These nodal points are allocated to the simplices $T \in \mathcal{T}_h$ and their number $\tilde{N}_h$ depends on the polynomial degree $p \geq 1$ and the spatial dimension. Due to the continuity condition $\mathbb{X}^p_h \subset C^0(\overline{\Omega})$ for $p \geq 1$ the nodal points have to coincide with the vertices of the simplices T (for $p \geq 1$), and with certain edge points of T (for $p \geq 2$), and possibly with certain inner points of T (see, e.g., [24, Section 6] for details). Using these nodal points the so-called *nodal basis* $(\tilde{\varphi}_i^{(h)})_{i=1}^{\tilde{N}_h}$ of $\mathbb{X}^p_h$ can be defined which is uniquely described by these functions $\tilde{\varphi}_i^{(h)} \in \mathbb{X}^p_h$ satisfying $\tilde{\varphi}_i^{(h)}(\tilde{b}_j^{(h)}) = \delta_{ij}$ for $i, j = 1, \ldots, \tilde{N}_h$. Considering only the inner points

$$\left\{ b_i^{(h)} \,:\, i = 1, \ldots, N_h \right\} := \bigcup_{i=1,\ldots,\tilde{N}_h} \left\{ \tilde{b}_i^{(h)} : \tilde{b}_i^{(h)} \notin \partial\Omega \right\}$$

that are not element of the boundary $\partial\Omega$, one defines the nodal basis $(\varphi_i^{(h)})_{i=1}^{N_h}$ of $\mathbb{X}^p_{h,0}$ which is uniquely described by these functions $\varphi_i^{(h)} \in \mathbb{X}^p_{h,0}$ satisfying $\varphi_i^{(h)}(b_j^{(h)}) = \delta_{ij}$ for $i, j = 1, \ldots, N_h$. By construction the basis functions $(\varphi_i^{(h)})_{i=1}^{N_h}$ have a small local support (cf. Figure 3.2) where

$$\operatorname{supp}(\varphi_i^{(h)}) \;\subset \bigcup_{\substack{T \in \mathcal{T}_h \\ \text{with } b_i^{(h)} \in \overline{T}}} \overline{T},$$

which results in sparse matrices $K^{(h)}$ and $M^{(h)}$ (cf. Lemma 3.6).

3.3.5. Isoparametric Finite Elements

So far it has been assumed that the domain $\Omega \subset \mathbb{R}^d$ is a polytope. However, finite element spaces are not restricted to such domains and can be extended to domains with a curved boundary. A well suited approach for this purpose are the so-called *isoparametric finite elements* (see, e.g., [23, 24, 37]) which are briefly described in the following.

Isoparametric finite elements generalise simplicial finite elements and the associated definition of an admissible triangulation $\mathcal{T}$ as follows: Instead of assuming in Definition 3.7 that each finite

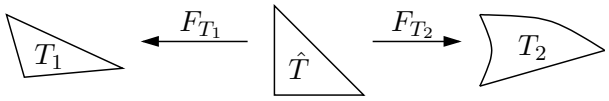


Figure 3.3.: Mapping of the reference element $\hat{T}$ (unit simplex) to the simplex finite element T_1 and to the isoparametric finite element T_2 where F_{T_1} is affine and F_{T_2} not.

element $T \in \mathcal{T}$ is a d-simplex, it is only required that T is an image of the d-dimensional unit simplex $\hat{T}$ (so-called *reference element*). In particular it has to hold that

i) each finite element $T \in \mathcal{T}$ is described by an diffeomorphism $F_T : \hat{T} \to T$,

ii) element maps of elements that share an edge or a face (or a higher-dimensional simplex at their surface) possess the same parametrization on that edge or face (or a higher-dimensional simplex).

If the element maps F_T are affine, then the simplicial finite elements from Definition 3.7 are obtained, otherwise one obtains elements with a curved boundary (cf. Figure 3.3). Using the isoparametric finite elements the definition of the finite element space $\mathbb{X}^p_{h,0}$ is generalised: Let $\mathcal{T}_h$ be an admissible triangulation of Ω consisting of isoparametric finite elements then the associated finite element space $\mathbb{X}^p_{h,0}$ is defined by

$$\mathbb{X}^p_{h,0} := \left\{ v \in C^0(\overline{\Omega}) \ : \ v_{|T} \circ F_T \in \mathbb{P}_p \text{ for all } T \in \mathcal{T}_h \right\} \cap H^1_0(\Omega) \qquad \text{for } p \geq 1. \tag{3.12}$$

When certain assumptions are made on the element maps F_T of a family of admissible isoparametric triangulations $\{\mathcal{T}_h\}_{h>0}$, then the approximation properties presented in Theorem 3.11 can be retained for the finite element space $\mathbb{X}^p_{h,0}$ from (3.12). These assumptions basically generalise the definition of a regular triangulation family for polytopal domains (cf. Definition 3.10) to domains with curved boundaries with isoparametric finite elements, and in literature exist different approaches for doing this (see, e.g., [37, Section 8.6] and [24, Chapter VI]). More details are presented in detail in the following section.

3.4. Approximation Properties of the Finite Element Discretisation

In Section 3.2 general approximation properties of the Ritz-Galerkin discretisation have been discussed. Depending on the properties of the finite-dimensional subspace V_h the discrete eigenvalues $\lambda_j^{(h)}$ and eigenfunctions $u_j^{(h)}$ of problem (3.1) are approximating the sought continuous eigenvalues λ_j and eigenfunctions u_j of problem (2.12). In Theorem 3.3 corresponding error estimates of $\lambda_j - \lambda_j^{(h)}$ and $u_j - u_j^{(h)}$ can be found. In this section approximation properties of the finite element discretisation are discussed, i.e., of the Ritz-Galerkin discretisation using the finite element spaces $\mathbb{X}^p_{h,0}$(cf. Section 3.3). More precisely, in this section error estimates of $\lambda_j - \lambda_j^{(h)}$ and $u_j - u_j^{(h)}$ are presented which will depend on the discretisation parameters h and p, the spatial dimension d, the size of the eigenvalue λ_j and its spectral separation. Furthermore, the following important issue is discussed: In Section 3.2 it was already highlighted that the discretisation parameter h has to be sufficiently small (cf. Theorem 3.3 – Theorem 3.5) in order that the predicted convergence rates of $\lambda_j^{(h)}$ and $u_j^{(h)}$ become valid. In practice this means

that the corresponding convergence rates get only visible when $h \leq h_{\max}$ (cf. [9]) with a fixed maximal mesh width $h_{\max} > 0$. In this section this maximal mesh width will be (asymptotically) quantified. Directly connected to the question on the maximal mesh width $h_{\max}$ — or respectively, the minimal dimension $N_{\min}$ of the finite element space $\mathbb{X}^p_{h,0}$ — is the question how many eigenvalues and how many eigenfunctions can be well approximated by the finite element discretisation using a given finite element space $\mathbb{X}^p_{h,0}$. Under certain assumptions it is obtained that, for example, in the three-dimensional case for the continuous problem (2.12) the smallest $CN_h^{2/5}$ eigenvalues and the eigenfunctions associated to the smallest $CN_h^{1/4}$ eigenvalues can be well approximated by the finite element method using the space $\mathbb{X}^1_{h,0}$ (space of piecewise affine functions) where $N_h = \dim \mathbb{X}^1_{h,0}$.

Throughout this section we will use the following precondition.

Precondition 3.12 (FEM Setting) *We consider the variational eigenvalue problem* (2.12) *and its Ritz-Galerkin discretisation* (3.1) *associated to a family of finite-dimensional subspaces* $(V_h)_{h>0}$ *with* $V_h \subset H^1_0(\Omega)$*, where the following assumption are made:*

a) Assumptions on the data

We assume that Precondition 2.9 is fulfilled, in particular this means that L *is a uniformly elliptic PDE operator with analytic coefficients and that* $\Omega \subset \mathbb{R}^d$ *is a bounded Lipschitz domain with analytic boundary.*

b) Assumptions on the numbering of the eigensolutions

We assume that the eigensolutions of the continuous problem (2.12) *and the discrete problem* (3.1) *are ordered by the size and the geometric multiplicity of the eigenvalues, i.e., for the corresponding eigensolutions*

$$\big(\lambda_j, u_j\big)_{j=1}^{\infty} \in \mathbb{R}_{>0} \times H^1_0(\Omega) \setminus \{0\} \quad \text{and} \quad \big(\,\lambda_j^{(h)}, u_j^{(h)}\,\big)_{j=1}^{N} \in \mathbb{R}_{>0} \times V_h \setminus \{0\} \tag{3.13}$$

holds $\lambda_j \leq \lambda_{j+1}$ *for* $j \in \mathbb{N}$ *and* $\lambda_j^{(h)} \leq \lambda_{j+1}^{(h)}$ *for* $j = 1, \ldots, N_h - 1$ *where* $N_h := \dim V_h$*. Furthermore, we assume (without loss of generality) that the eigenfunctions of the continuous problem are orthonormal in* $L^2(\Omega)$*, i.e., it holds*

$$a(u_i, u_j) = \lambda_j (u_i, u_j)_0 = \lambda_j \delta_{ij} \qquad \text{for all } i, j \in \mathbb{N}.$$

c) Assumptions on the subspaces $(V_h)_{h>0}$

We assume that the finite element spaces $V_h = \mathbb{X}^p_{h,0}$ *from* (3.12) *are used for the Ritz-Galerkin discretisation with some fixed polynomial degree* $p \geq 1$*, where the associated family of admissible triangulations* $\{\mathcal{T}_h\}_{h>0}$ *uses the isoparametric finite elements described in Section 3.3.5, which admits domains with curved boundary. For the corresponding element maps it is assumed that for each* $T \in \mathcal{T}_h$ *the element map* $F_T : \hat{T} \to T$ *can be written as* $F_T = R_T \circ A_T$*, where* A_T *is affine and the maps* A_T*,* R_T *fulfil for constants* $C_A, C_R, \gamma_R > 0$ *independent of* $T \in \mathcal{T}_h$ *and*

independent of $h = h(\mathcal{T}_h)$ *that:*

$$i) \qquad \|DA_T\|_\infty \leq C_A h_T, \qquad \|(DA_T)^{-1}\|_\infty \leq C_A h_T^{-1}, \qquad \|(DR_T)^{-1}\|_\infty \leq C_R \tag{3.14}$$

where $h_T := \operatorname{diam}(T)$ and D denotes the total derivative.

$$ii) \qquad |R|_{n,\infty} := \left\| \left\{ \sum_{|\alpha|=n} \frac{n!}{\alpha!} |D^\alpha R|^2 \right\}^{1/2} \right\|_{L^\infty(A_T(\hat{T}))} \leq C_R \, n! \, (\gamma_R)^n \quad \text{for all } n \in \mathbb{N}_0,$$

where $|D^\alpha R(x)|^2 := \sum_{i=1}^{d} |D^\alpha R_i(x)|^2$ and R_i are the component functions of R.

Remark 3.13 *i) The assumptions made in Precondition 3.12 c) on the triangulations* $\{\mathcal{T}_h\}_{h>0}$ *appear technically, however, they generalise Definition 3.10: In the case that* Ω *is a polytope and all element maps* F_T *are affine (i.e.,* R_T *is the identity mapping) then the requirements on* A_T *in* (3.14) *are equivalent with the requirements on a regular family of triangulations from Definition 3.10. Furthermore, the assumptions made on* R_T *imply that* R_T *is an analytic diffeomorphism (cf.* inverse function theorem*).*

ii) A family of triangulations $\{\mathcal{T}_h\}_{h>0}$ *satisfying Precondition 3.12 c) can be obtained as follows (cf. [64]): Let* $\mathcal{T}_{\text{macro}}$ *be a fixed triangulation (possibly with curved elements) with analytic element maps* $F_T : \hat{T} \to T$ *that resolve the geometry of* Ω*. Furthermore, consider a regular family of triangulations* $\{\hat{\mathcal{T}}_{\hat{h}}\}_{\hat{h}>0}$ *(cf. Definition 3.10) of the reference element* $\hat{T}$ *(unit simplex). Then triangulations* $\mathcal{T}_h$ *of* Ω *satisfying Precondition 3.12 c) are obtained by mapping the triangulations* $\hat{\mathcal{T}}_{\hat{h}}$ *of* $\hat{T}$ *with the macro element maps* $F_T : \hat{T} \to T$*.*

The error estimates of $\lambda_j - \lambda_j^{(h)}$ and $u_j - u_j^{(h)}$ which are presented in the following have been original derived in [64]. However, in this work the estimates of [64] have been slightly adjusted (cf. Appendix D) with a special focus on the necessary conditions on h such that the predicted convergence rates get visible. Furthermore, certain assumptions on the asymptotic behaviour of the eigenvalues have been made in [64] in order to derive the error estimates: Firstly, it is assumed that there exists a constant $C_{\text{b}} > 0$ independent of j such that $j \leq C_{\text{b}} \lambda_j^{d/2}$, and secondly, it is assumed that the spectral gap is bounded by $\Delta_1(\lambda_j) \geq c_{\text{g}} \lambda_j^{-d/2}$, see (C.8) for details, where $c_{\text{g}} > 0$ is a constant independent of j (which has been motivated under quite technical assumptions for a special case). In this work, however, it could be shown that in general there exist constants $C_{\text{b}}, c_{\text{b}} > 0$ independent of j (cf. Theorem C.2) such that the eigenvalues of problem (2.12) can be bounded by

$$c_{\text{b}} \, \lambda_j^{d/2} \leq j \leq C_{\text{b}} \, \lambda_j^{d/2} \quad \text{and} \quad \left(\frac{j}{C_{\text{b}}}\right)^{2/d} \leq \lambda_j \leq \left(\frac{j}{c_{\text{b}}}\right)^{2/d} \qquad \text{for all } j \in \mathbb{N}. \tag{3.15}$$

Although result (3.15) was not hard to establish it is not described (to the best of the author's knowledge) in literature in this form. Furthermore, it could be shown in this work that in general the spectral gap of the eigenvalues can be bounded by $\Delta_1(\lambda_j) \geq c_{\text{g}} \lambda_j^{-d/2}$ with some constant $c_{\text{g}} > 0$ independent of j when assumptions on a certain remainder term are fulfilled (cf. Theorem C.4). Especially result (3.15) opens up the possibility to discuss not only the minimal

dimension of a finite element space to well approximate a given eigenvalue or eigenfunction, but also allows to measure asymptotically the number of well approximable eigenvalues and eigenfunctions by the finite element method using a given finite element space. The corresponding results are summarised in the following, however, the analysis and proofs can be found in Appendix C (asymptotic distribution of eigenvalues) and Appendix D (preliminary work FEM approximation).

Corollary 3.14 (Summary I) *Consider the variational eigenvalue problem* (2.12) *and its Ritz-Galerkin discretisation* (3.1)*, and assume in the following that Precondition 3.12 is satisfied and that* $j \in \{1, \dots, N_h\}$*.*

Eigenvalue Approximation:

Let $C_{\text{idx}}^{\text{EV}}, C_{\text{size}}^{\text{EV}} > 0$ *be some sufficiently small constants and let the mesh width* h *of the finite element space* V_h *be chosen such that* one *of the following conditions is fulfilled*

$$\text{i)} \qquad h^{2p}\, j^{1+\frac{2p}{d}} \;\leq\; C_{\text{idx}}^{\text{EV}}, \tag{3.16}$$

$$\text{ii)} \qquad h^{2p}\, \lambda_j^{p+\frac{d}{2}} \;\leq\; C_{\text{size}}^{\text{EV}}. \tag{3.17}$$

Then it holds

$$0 \leq \frac{\lambda_j^{(h)} - \lambda_j}{\lambda_j} \leq C_4'\, j \left(\left(\frac{C_2 h}{h+\sigma} \right)^{2p} + \left(\frac{\sqrt{\lambda_j} h}{\sigma p} \right)^{2p} \right) \quad \text{with } C_4' := \frac{2C_3}{(\,\min\{\lambda_1, 1\}\,)^3}. \tag{3.18}$$

where $C_2, C_3, \sigma > 0$ *are constants independent*[3] *of* j, h, p*.*

Eigenfunction Approximation:

Assume additionally that all continuous eigenvalues are simple, i.e., it holds

$$\lambda_1 < \lambda_2 < \lambda_3 < \dots, \tag{3.19}$$

and assume for $j \in \mathbb{N}$ *(in view of Theorem C.4) that the spectral gap satisfies*

$$\min_{i \in \{j_+, j+1\}} \frac{\lambda_i - \lambda_{i-1}}{2\lambda_i} \;\geq\; c_6 \lambda_j^{-\frac{d}{2}} \qquad \text{with } j_+ := \max\{j, 2\} \tag{3.20}$$

for some constant $c_6 > 0$ *independent of* j*. Furthermore, let* $C_{\text{idx}}^{\text{EF}}, C_{\text{size}}^{\text{EF}} > 0$ *be some sufficiently small constants and let the mesh width* h *of the finite element space* V_h *be chosen such that* one *of the following conditions is fulfilled*

$$\text{i)} \qquad h^{2p}\, j^{2+\frac{2p}{d}} \;\leq\; C_{\text{idx}}^{\text{EF}}, \tag{3.21}$$

$$\text{ii)} \qquad h^{2p}\, \lambda_j^{p+d} \;\leq\; C_{\text{size}}^{\text{EF}}. \tag{3.22}$$

Then there exist a discrete eigenfunction $\widetilde{u}_j^{(h)} \in E_h\big(\lambda_j^{(h)}\big)$ *such that*

$$\frac{\|u_j - \widetilde{u}_j^{(h)}\|_1}{\|u_j\|_1} \;\leq\; \frac{\widetilde{C}_3}{\min\{1, \lambda_1\}} \left(1 + \frac{\widetilde{C}_4}{c_6}\, \lambda_j^{1+\frac{d}{2}}\, h^{\min\{p,2\}} \right) \left(\frac{1}{\sqrt{\lambda_j}} \left(\frac{C_2 h}{h+\sigma} \right)^p + \left(\frac{\sqrt{\lambda_j} h}{\sigma p} \right)^p \right). \tag{3.23}$$

where $C_2, \widetilde{C}_3, \widetilde{C}_4, \sigma > 0$ *are constants independent*[4] *of* j, h, p*.*

[3] The constants C_2, C_3 depend on $C_{\text{idx}}^{\text{EV}}$ and $C_{\text{size}}^{\text{EV}}$ respectively.

[4] The constants $C_2, \widetilde{C}_3$ depend on $C_{\text{idx}}^{\text{EV}}$ and $C_{\text{size}}^{\text{EV}}$ respectively, and the constant σ is the same as in (3.18).

Proof: The proof can be found in the end of Appendix D. ■

For reasons of simplification the error analysis of the eigenfunction approximation has been restricted to the case that all continuous eigenvalues are simple. This restricting is quite strong, however, similar error estimates can be derived for the general case by combining the approximation properties between the continuous eigenfunctions and the finite element space V_h (which are described in [64]) with general results of the Ritz-Galerkin discretisation for multiple eigenvalues (which can found, e.g., in [6] and [48]).

Furthermore, it is noted that result (3.15) allows to derive a priori error estimates for the eigenvalue and eigenfunction approximation which are independent of the size of the (associated) eigenvalue, and which depend instead on the corresponding index[5]: For example, from (3.23) and (3.15) error estimate

$$\frac{\|u_j - \widetilde{u}_j^{(h)}\|_1}{\|u_j\|_1} \;\leq\; \left(\widetilde{C}_3' + \widetilde{C}_4'\, j^{1+\frac{2}{d}}\, h^{\min\{p,2\}}\right)\left(\frac{1}{j^{1/d}}\left(\frac{C_2 h}{h+\sigma}\right)^p + \left(\frac{j}{c_{\mathrm{b}}}\right)^{p/d}\left(\frac{h}{\sigma p}\right)^p\right)$$

is derived with constants $C_2, \widetilde{C}_3', \widetilde{C}_4', \sigma > 0$ independent of j, h, p.

The error estimates of Corollary 3.14 (and Theorem D.4) reveal the asymptotic dependence of the approximation error on the discretisation parameters h and p, on the size (and index) of the eigenvalue, and on the spectral gap. However, the error estimates become valid only when the mesh width h of V_h is small enough to fulfil the conditions (3.16), (3.17), (3.21) or (3.22). In particular, we observe that only the smallest eigenvalues λ_j and their associated eigenfunctions u_j can be well approximated by the finite element method: On one side the approximation error increases with increasing size of the eigenvalue, and on the other side, even more crucial, the necessary conditions on the mesh width h can be fulfilled more easily by the smallest eigenvalues.

Remark 3.15 (Refining the FEM Space) *In the concrete application of FEM the mesh width h is fixed. However, if the finite element space should be further refined the question arises which type of refinement is more efficient: Is it more efficient to (locally) reduce the mesh width h ("h-refinement"), or to (locally) increase the polynomial degree p ("p-refinement"), or to do both of it. Under the assumption that* $\dim V_h \in \mathcal{O}(\,(p/h)^d\,)$ *the observation of* (3.18) *and* (3.23) *shows that the p-refinement is more advantageous regarding a small dimension since (provided that h is sufficiently small) the discretisation error is decreasing exponentially in p.*

The conditions on the mesh width h in Corollary 3.14 allow to decide if an eigenvalue or eigenfunction of the continuous problem (2.12) is well approximable by the finite element method. In this context the question arises what is the maximal possible mesh width or the minimal dimension of a finite element space to well approximate a given eigenvalue or eigenfunction and, furthermore, how many eigenvalues and eigenfunctions can be well approximated by a given finite element space V_h. These questions are answered in the following by Theorem 3.17, but before, to ease the discussion, it is explicitly defined when an eigensolution is called well approximable.

Definition 3.16 (Well Approximable) *In view of Corollary 3.14 we call an eigenvalue of the continuous problem* (2.12) *as "well approximable by (the finite element method using the*

[5] In this section the *index* of an eigenvalue λ_j and eigenfunction u_j denotes always the numbering index j (not to be confused with the index of an eigenvalue defined in Theorem A.12).

finite element space) V_h" if condition (3.16) *or condition* (3.17) *is fulfilled. Analogically, we call an eigenfunction of problem* (2.12) *as well approximable by V_h if condition* (3.21) *or* (3.22) *is valid.*

In particular, this means that if an eigenvalue or eigenfunction is called well approximable the approximation error can be bounded by the right-hand side of (3.18) and (3.23).

Corollary 3.17 (Summary II) *Consider the variational eigenvalue problem* (2.12) *and its Ritz-Galerkin discretisation* (3.1)*, and assume that Precondition 3.12 is fulfilled. Furthermore, assume that $h \leq h_{\text{asymp}}$ for some $h_{\text{asymp}} > 0$ and that the dimension of the finite element space V_h is bounded*[6] *by*

$$c\,\frac{1}{h^d} \;\leq\; N_h \;\leq\; C\,\frac{1}{h^d} \qquad \text{for all } h \text{ with } 0 < h \leq h_{\text{asymp}}. \tag{3.24}$$

For the following results concerning the eigenfunction approximation it is additionally assumed that all eigenvalues are simple and that the spectral gap fulfils condition (3.20)*.*

Maximal Mesh Width of the Finite Element Space:

If we denote by $h^{\text{EV}}_{\max}(j)$ and $h^{\text{EV}}_{\max}(\lambda)$ the maximal mesh width of V_h in order to well approximate the continuous eigenvalue $\lambda_j = \lambda$, and by $h^{\text{EF}}_{\max}(j)$ and $h^{\text{EF}}_{\max}(\lambda)$ the maximal mesh width to well approximate the associated eigenfunction $u_j = u$ then these quantities are given by

$$\begin{aligned} h^{\text{EV}}_{\max}(j) \;&=\; \tilde{C}^{\text{EV}}_{\text{idx}}\, j^{-\frac{2p+d}{2pd}} \qquad \text{and} \qquad h^{\text{EF}}_{\max}(j) \;=\; \tilde{C}^{\text{EF}}_{\text{idx}}\, j^{-\frac{p+d}{pd}}, \\ h^{\text{EV}}_{\max}(\lambda) \;&=\; \tilde{C}^{\text{EV}}_{\text{size}}\, \lambda^{-\frac{2p+d}{4p}} \qquad \text{and} \qquad h^{\text{EF}}_{\max}(\lambda) \;=\; C^{\text{EF}}_{\text{size}}\, \lambda^{-\frac{p+d}{2p}} \end{aligned}$$

with suitable constants $\tilde{C}^{\text{EV}}_{\text{idx}}, \tilde{C}^{\text{EF}}_{\text{idx}}, \tilde{C}^{\text{EV}}_{\text{size}}, \tilde{C}^{\text{EF}}_{\text{size}} > 0$ independent of j, λ, h.

Minimal Dimension of the Finite Element Space:

If we denote by $N^{\text{EV}}_{\min}(j)$ and $N^{\text{EV}}_{\min}(\lambda)$ the minimal dimension of V_h in order to well approximate the continuous eigenvalue $\lambda_j = \lambda$, and by $N^{\text{EF}}_{\min}(j)$ and $N^{\text{EF}}_{\min}(\lambda)$ the minimal dimension to well approximate the associated eigenfunction $u_j = u$ then these quantities are asymptotically given by

$$N^{\text{EV}}_{\min}(j) \;=\; \Theta\Big(j^{1+\frac{d}{2p}} \Big) \qquad \text{and} \qquad N^{\text{EF}}_{\min}(j) \;=\; \Theta\Big(j^{1+\frac{d}{p}} \Big), \tag{3.25}$$

$$N^{\text{EV}}_{\min}(\lambda) \;=\; \Theta\Big(\lambda^{\frac{d}{2}+\frac{d^2}{4p}} \Big) \qquad \text{and} \qquad N^{\text{EF}}_{\min}(\lambda) \;=\; \Theta\Big(\lambda^{\frac{d}{2}+\frac{d^2}{2p}} \Big). \tag{3.26}$$

Maximal Number and Size of Well Approximable Eigensolutions:

If we denote by $j^{\text{EV}}_{\max}(N_h)$ the maximal index of the continuous eigenvalues and by $j^{\text{EF}}_{\max}(N_h)$ the

[6] The assumption (3.24) is not very restrictive. The assumption (3.24) is fulfilled, for example, when the triangulations $\{\mathcal{T}_h\}_{h>0}$ associated to finite element space V_h are constructed as described in Remark 3.13 ii) whereby the triangulation of the reference element $\hat{T}$ (unit simplex) is refined using *uniform* grids.

maximal index of the continuous eigenfunctions which are well approximable by V_h then these quantities are asymptotically given by

$$j^{\mathrm{EV}}_{\max}(N_h) \;=\; \Theta\left(N_h^{\frac{2p}{2p+d}} \right) \qquad \text{and} \qquad j^{\mathrm{EF}}_{\max}(N_h) \;=\; \Theta\left(N_h^{\frac{p}{p+d}} \right). \tag{3.27}$$

If we denote by $\lambda^{\mathrm{EV}}_{\max}(N_h)$ the maximal size of the continuous eigenvalues which are well approximable by V_h, and by $\lambda^{\mathrm{EF}}_{\max}(N_h)$ the maximal size of the eigenvalues associated to the continuous eigenfunctions which are well approximable by V_h, then these quantities are asymptotically given by

$$\lambda^{\mathrm{EV}}_{\max}(N_h) \;=\; \Theta\left(N_h^{\frac{4p}{d(2p+d)}} \right) \qquad \text{and} \qquad \lambda^{\mathrm{EF}}_{\max}(N_h) \;=\; \Theta\left(N_h^{\frac{2p}{d(p+d)}} \right). \tag{3.28}$$

Proof: The proof can be found in the end of Appendix D. ∎

From Corollary 3.17 follows that, for example, the eigenvalue λ_j is well approximable by all finite element spaces $\mathbb{X}^p_{h,0}$ when the underlying mesh width is fulfilling $h \leq h^{\mathrm{EV}}_{\max}(j)$ or when the dimension of the finite element space is fulfilling $N_h \geq C N^{\mathrm{EV}}_{\min}(j)/c$ [cf. proof of Corollary 3.17] for suitable constant $C, c > 0$ independent of j, h. The results of Corollary 3.17 are summarised In Table 3.1 and Table 3.2 for the dimensions $d = 1, 2, 3$ where the important space $\mathbb{X}^1_{h,0}$ (space of piecewise affine functions) is highlighted.

The maximal mesh widths $h^{\mathrm{EV}}_{\max}(\lambda), h^{\mathrm{EF}}_{\max}(\lambda)$ and the asymptotic behaviour of the minimal dimensions $N^{\mathrm{EV}}_{\min}(\lambda)$, $N^{\mathrm{EF}}_{\min}(\lambda)$ have been first derived in [64], however, by explicitly assuming that there exists a constant $C_{\mathrm{b}} > 0$ independent of j such that $j \leq C_{\mathrm{b}} \lambda_j^{d/2}$. In this work it is shown that this assumption is fulfilled anyway (cf. Theorem C.2). Furthermore, in this work the maximal mesh widths and the asymptotic behaviour of the minimal dimensions have been derived in dependence of the index j of the eigenvalue or eigenfunction. Moreover, the question on how many eigenvalues and eigenfunctions can be well approximated by a given finite element space V_h has been answered and how large the (associated) eigenvalue can get in order to be well approximable.

Remark 3.18 (Sharpness of the Results) *i) In [9] first systematic numerical experiments have been performed [for a 1D and 2D model problem with $V_h = \mathbb{X}^1_{h,0}$] in order to investigate the asymptotic sharpness of the error estimates* (3.18) *and* (3.23)*, and the asymptotic sharpness*[7] *of the maximal mesh widths $h^{\mathrm{EV}}_{\max}(\lambda)$ and $h^{\mathrm{EF}}_{\max}(\lambda)$. The sharpness of the results for the eigenfunction approximation could be largely validated. The measured error of the eigenvalue approximation, however, has been asymptotically slightly smaller than predicted [on the right-hand side of* (3.18) *the factor j could not be observed], and $h^{\mathrm{EV}}_{\max}(\lambda)$ has been slightly too strict, i.e., the asymptotic convergence rate became visible for mesh widths slightly larger then $h^{\mathrm{EV}}_{\max}(\lambda)$. However, the benchmarked model problems exhibit a special structure (cf. [9]) which facilitate the FEM approximation of the eigenvalues and eigenfunctions, so that according to [9] this might be the reason for the slightly better approximation properties.*

[7] Sharpness in the sense that the asymptotic convergence rates of the eigenvalue and eigenfunction approximation become visible if and only if $h \leq h^{\mathrm{EV}}_{\max}(\lambda)$ and $h \leq h^{\mathrm{EF}}_{\max}(\lambda)$.

ii) Since all other quantities described in Corollary 3.17 can be derived from $h_{\max}^{\mathrm{EV}}(\lambda)$ *and* $h_{\max}^{\mathrm{EF}}(\lambda)$ *using the estimates* (3.15) *and* (3.24) *the asymptotical sharpness of these quantities follows from the sharpness of* $h_{\max}^{\mathrm{EV}}(\lambda)$ *and* $h_{\max}^{\mathrm{EF}}(\lambda)$.

iii) If in Corollary 3.17 it is assumed that $\dim V_h \in \mathcal{O}(1/h^d)$ *instead of* $\dim V_h \in \Theta(1/h^d)$ *then the statements on the asymptotical behaviour of the minimal dimension, maximal index and maximal size have to be weakened: For the minimal dimension we obtain the same statements as in* (3.25) *and* (3.26) *but only of the form* $\mathcal{O}(\ldots)$*, and for the maximal index and the maximal size we derive the same statements as in* (3.27) *and* (3.28) *but only of the form* $\Omega(\ldots)$. [8]

Remark 3.19 (Analytic Data) *For the results presented in Corollary 3.14 and Corollary 3.17 it is essential that the eigenfunctions* u *of problem* (2.12) *fulfil* $u \in C^\infty(\Omega)$. *By assuming analytic data (Precondition 2.9) in Precondition 3.12a) the smoothness of the eigenfunctions can be guaranteed using Theorem 2.10. However, if Precondition 3.12a) is not fulfilled but it can be guaranteed anyway that for all eigenfunctions* u *of problem* (2.12) *it holds* $u \in C^\infty(\Omega)$*, then the results presented in Corollary 3.14 and Corollary 3.17 remain valid (cf. [64] and Appendix D).*

[8] See the *list of symbols* for the definition of the asymptotic notations $\mathcal{O}(\ldots)$, $\Theta(\ldots)$ and $\Omega(\ldots)$.

	FEM space $V_h = \mathbb{X}^1_{h,0}$			FEM space $V_h = \mathbb{X}^p_{h,0}$		
	$d=1$	$d=2$	$d=3$	$d=1$	$d=2$	$d=3$
$N^{\mathrm{EV}}_{\min}(j)$	$\Theta(j^{3/2})$	$\Theta(j^{2})$	$\Theta(j^{5/2})$	$\Theta\left(j^{\frac{2p+1}{2p}}\right)$	$\Theta\left(j^{\frac{2p+2}{2p}}\right)$	$\Theta\left(j^{\frac{2p+3}{2p}}\right)$
$N^{\mathrm{EF}}_{\min}(j)$	$\Theta(j^{2})$	$\Theta(j^{3})$	$\Theta(j^{4})$	$\Theta\left(j^{\frac{p+1}{p}}\right)$	$\Theta\left(j^{\frac{p+2}{p}}\right)$	$\Theta\left(j^{\frac{p+3}{p}}\right)$
$N^{\mathrm{EV}}_{\min}(\lambda)$	$\Theta(\lambda^{3/4})$	$\Theta(\lambda^{2})$	$\Theta(\lambda^{15/4})$	$\Theta\left(\lambda^{\frac{2p+1}{4p}}\right)$	$\Theta\left(\lambda^{\frac{4p+4}{4p}}\right)$	$\Theta\left(\lambda^{\frac{6p+9}{4p}}\right)$
$N^{\mathrm{EF}}_{\min}(\lambda)$	$\Theta(\lambda^{1})$	$\Theta(\lambda^{3})$	$\Theta(\lambda^{6})$	$\Theta\left(\lambda^{\frac{p+1}{2p}}\right)$	$\Theta\left(\lambda^{\frac{2p+4}{2p}}\right)$	$\Theta\left(\lambda^{\frac{3p+9}{2p}}\right)$

Table 3.1.: In view of Corollary 3.17 the asymptotical behaviour of the minimal dimension of the finite element space V_h in order to well approximate an eigenvalue (EV) or an eigenfunction (EF) of the continuous problem (2.12) by the finite element method.

	FEM space $V_h = \mathbb{X}^1_{h,0}$			FEM space $V_h = \mathbb{X}^p_{h,0}$		
	$d=1$	$d=2$	$d=3$	$d=1$	$d=2$	$d=3$
$j^{\mathrm{EV}}_{\max}(N_h)$	$\Theta(N_h^{2/3})$	$\Theta(N_h^{1/2})$	$\Theta(N_h^{2/5})$	$\Theta\left(N_h^{\frac{2p}{2p+1}}\right)$	$\Theta\left(N_h^{\frac{2p}{2p+2}}\right)$	$\Theta\left(N_h^{\frac{2p}{2p+3}}\right)$
$j^{\mathrm{EF}}_{\max}(N_h)$	$\Theta(N_h^{1/2})$	$\Theta(N_h^{1/3})$	$\Theta(N_h^{1/4})$	$\Theta\left(N_h^{\frac{p}{p+1}}\right)$	$\Theta\left(N_h^{\frac{p}{p+2}}\right)$	$\Theta\left(N_h^{\frac{p}{p+3}}\right)$
$\lambda^{\mathrm{EV}}_{\max}(N_h)$	$\Theta(N_h^{4/3})$	$\Theta(N_h^{1/2})$	$\Theta(N_h^{4/15})$	$\Theta\left(N_h^{\frac{4p}{2p+1}}\right)$	$\Theta\left(N_h^{\frac{4p}{4p+4}}\right)$	$\Theta\left(N_h^{\frac{4p}{6p+9}}\right)$
$\lambda^{\mathrm{EF}}_{\max}(N_h)$	$\Theta(N_h^{1})$	$\Theta(N_h^{1/3})$	$\Theta(N_h^{1/6})$	$\Theta\left(N_h^{\frac{2p}{p+1}}\right)$	$\Theta\left(N_h^{\frac{2p}{2p+4}}\right)$	$\Theta\left(N_h^{\frac{2p}{3p+9}}\right)$

Table 3.2.: In view of Corollary 3.17 the asymptotical behaviour of the maximal index and the maximal size of the (associated) eigenvalue to well approximate an eigenvalue (EV) or an eigenfunction (EF) of the continuous problem (2.12) by the finite element method.

4. Summary and Problem Description

The objective of this work and the results of the previous chapters can be summarised as follows: This work is focused on the efficient solution of the continuous eigenvalue problem

$$\begin{cases} Lu = \lambda u & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega \end{cases} \tag{4.1}$$

where $\Omega \subset \mathbb{R}^d$ is a bounded domain ($d = 2, 3$) with Lipschitz boundary $\partial\Omega$ and L is a linear uniformly elliptic second order PDE operator (cf. Definition 2.1) in divergence form

$$Lu = -\mathrm{div}\big(A\nabla u\big) + cu = -\sum_{i,j=1}^{d} \frac{\partial}{\partial x_i}\left(a_{ij}\frac{\partial}{\partial x_j}u\right) + cu$$

with $L^\infty(\Omega)$-functions a_{ij}, c where $A := (a_{ij})_{i,j=1}^d$ and $c \geq 0$. To solve problem (4.1) the corresponding weak formulation

$$\begin{cases} \text{find } (\lambda, u) \in \mathbb{R} \times H_0^1(\Omega) \setminus \{0\} \text{ such that} \\ a(u,v) \;=\; \lambda\,(u,v)_0 \qquad \forall\, v \in H_0^1(\Omega) \end{cases} \tag{4.2}$$

is derived where $a(u,v) = \int_\Omega \nabla u^T A \nabla v + cuv \,\mathrm{d}x$ is a symmetric elliptic bilinear form (cf. proof of Theorem 2.5) and $(u,v)_0 = \int_\Omega uv \,\mathrm{d}x$ is the inner product of $L^2(\Omega)$. According to Theorem 2.6 the continuous eigenvalue problem (4.2) possesses a countable family of eigensolutions

$$\big(\lambda_j, u_j\big)_{j=1}^\infty \in \mathbb{R}_{>0} \times H_0^1(\Omega) \setminus \{0\} \qquad \text{with } \lambda_j \leq \lambda_{j+1} \tag{4.3}$$

with positive real eigenvalues.

Solutions of the continuous eigenvalue problem (4.3) are approximated by discretisation: Using a conforming finite element space $V_h \subset H_0^1(\Omega)$ [e.g., the space $\mathbb{X}_{h,0}^p$ from Section 3.3] with dimension N_h and nodal basis $\big(\varphi_i^{(h)}\big)_{i=1}^{N_h}$ the eigenvalue problem (4.2) is discretised by

$$\begin{cases} \text{find } (\lambda^{(h)}, x^{(h)}) \in \mathbb{R} \times \mathbb{R}^{N_h} \setminus \{0\} \text{ with} \\ K^{(h)}\, x^{(h)} \;=\; \lambda^{(h)}\, M^{(h)}\, x^{(h)} \end{cases} \tag{4.4}$$

(cf. Section 3.1) where the stiffness and mass matrix

$$K^{(h)} := \Big(a(\varphi_j^{(h)}, \varphi_i^{(h)})\Big)_{i,j=1}^{N_h} \in \mathbb{R}^{N_h \times N_h} \quad \text{and} \quad M^{(h)} := \Big((\varphi_j^{(h)}, \varphi_i^{(h)})_0\Big)_{i,j=1}^{N_h} \in \mathbb{R}^{N_h \times N_h} \tag{4.5}$$

are both sparse, symmetric and positive definite. The eigenvalues of (4.4) are positive real, and the corresponding eigenpairs are given by

$$\big(\lambda_j^{(h)}, x_j^{(h)}\big)_{j=1}^{N_h} \in \mathbb{R}_{>0} \times \mathbb{R}^{N_h} \setminus \{0\} \qquad \text{with } \lambda_j^{(h)} \leq \lambda_{j+1}^{(h)}. \tag{4.6}$$

As described in Section 3.2 the discrete eigenpairs (4.6) are approximating the continuous eigensolutions (4.3): According to Theorem 3.2 it holds $\lambda_j^{(h)} \to \lambda_j$ for $h \to 0$; and for the discrete eigenfunctions $u_j^{(h)} := \mathcal{P} x_j^{(h)} \in V_h$ [with prolongation operator $\mathcal{P}$ from (3.3) and assuming that $\|u_j^{(h)}\|_1 = 1$] exists a subsequence $\left(u_j^{(h_k)}\right)_{k\in\mathbb{N}}$ which converges in $H_0^1(\Omega)$ to an eigenfunction $\widetilde{u}_j \in E(\lambda_j)$ when $h_k \xrightarrow{k\to\infty} 0$ [where $E(\lambda_j)$ is the eigenspace of λ_j defined in (2.14)].

However, as discussed in Section 3.4 only the smaller eigenvalues λ_j and their corresponding eigenfunctions u_j can be well approximated by the finite element space V_h (see, e.g., [9] for experimental studies) because the approximation error increases with increasing size of the eigenvalue. Furthermore, the eigenvalues λ_j have to be small enough so that necessary conditions on the discretisation mesh width h of the finite element space are fulfilled and thus the corresponding convergence rates become valid (see, e.g., Corollary 3.14). If, for example, the assumptions of Corollary 3.17 are fulfilled (i.e., analytic data is assumed) then, e.g., for three-dimensional problems only the first $\Theta(N_h^{2/5})$ eigenvalues and only the first $\Theta(N_h^{1/4})$ eigenfunctions (under the assumption that all eigenvalues are simple) are well approximable by the finite element method using the finite element spaces $V_h = \mathbb{X}_{h,0}^1$. If some of the assumptions of Corollary 3.17 are not fulfilled it can be expected that even less eigenvalues and less eigensolutions can be well approximated.

Correspondingly, in the following we are only interested in computing a portion of the smallest eigenpairs of the discrete problem (4.4), e.g., the first

$$n_{\text{es}} = CN_h^{2/5} \in \mathbb{N} \quad \text{or} \quad n_{\text{es}} = CN_h^{1/4} \in \mathbb{N}$$

eigenpairs, for some constant $C > 0$. The computation of the remaining eigenpairs of (4.4) associated to larger eigenvalues is not reasonable because typically they do not provide useful approximations of continuous eigensolutions.

The solution of the algebraic eigenvalue problem (4.4) is typically performed by a classical approach, i.e., by some iterative algebraic eigensolver such as the Lanczos method [8, 36] or the subspace iteration [10]. Classical iterative approaches are well suited if the number of sought eigensolutions n_{es} is rather small, e.g., if $n_{\text{es}} = 5$. However, because we are interested in a large number of eigensolutions, we will use the AMLS method to solve the eigenvalue problem (4.2), respectively (4.4). The AMLS method, which is presented in the next chapter, has proven to be very efficient when a large number of eigensolutions is sought (cf. [42]). If, however, the number of sought discrete eigenpairs approaches N_h, it is advisable to use instead of AMLS either a cubic scaling direct method or an iterative method like shift-invert Lanczos (cf. [36]) with a good shift strategy and an efficient solver for the arising shifted linear systems.

5. Automated Multi-Level Substructuring

The so-called *automated multi-level substructuring* (short AMLS) method is a very efficient approach to solve an elliptic PDE eigenvalue problem. The AMLS method was mainly developed by Bennighof and co-authors [14, 16, 47], and is based on the classical *component mode synthesis* (short CMS).

The CMS is as a substructuring method which was already developed in 1960 by Hurty [44] for the solution of large scale eigenvalue problems arising in structural engineering analysis. The idea behind CMS is to substructure the spatial domain of the PDE eigenvalue problem into subdomains, and to approximate the sought eigensolutions of the global problem by using eigensolutions of problems that are defined on the smaller subdomains. The method was further improved by Craig and Bampton [26], and during the years CMS became very popular and was studied by many researchers which developed refined versions. An overview over different CMS versions can be found in [65]. The first mathematical analysis of CMS, however, has been performed only in the early 1990s by Bourquin and d'Hennezel (see, e.g., [20, 21, 22]).

Bennighof extended and generalised in the AMLS method the single-level substructuring of CMS to a multi-level version. The procedure of AMLS is as follows: In the first step the spatial domain of the PDE eigenvalue problem is recursively subdivided into several subdomains. In the next step, on each of these subdomains and additionally on the interfaces, which are separating the subdomains, certain eigenvalue problems are defined that are induced by the global problem, and which are typically small and easy to solve. From each of these subproblems a few eigensolutions are computed which are meant to represent the global problem on the corresponding subdomain or interface. In the next step, the computed eigensolutions of the subproblems are used to form a subspace onto which the global eigenvalue problem is projected. The projection results in a *reduced eigenvalue problem* which is of much smaller size than the original problem and typically easy to solve. Finally, the eigenpairs of the reduced eigenvalue problem are computed which deliver approximations of the sought eigenpairs of the global eigenvalue problem.

The AMLS method has proven to be very efficient for solving large-scale eigenvalue problems arising in structural engineering analysis (see, e.g., [15, 47, 55]). Especially when a large number of eigenpair approximations is required, AMLS has shown to be more efficient than classical approaches using iterative algebraic eigensolvers that are coupled with a preconditioner or a linear solver (cf. [42]). The big advantage of the AMLS method is that its computational costs increase only slightly with the number of sought eigenpairs, and hence a large amount of eigenpairs can be computed at once. A very popular classical approach is the *shift-invert block Lanczos* (short block-SIL) algorithm [36] which is commonly used in structural engineering. Kropp and Heiserer could present in [55] breakthrough calculations when they benchmarked AMLS against block-SIL within a vibro-acoustic analysis of an automobile body. In these benchmarks the AMLS method running on a commodity workstation has been several times faster than block-SIL running on a supercomputer.

When the AMLS method is applied to a discrete eigenvalue problem it computes only approx-

imations of the discrete eigenpairs whereas block-SIL computes the discrete eigenpairs almost numerically exact. This seems to be disadvantageous, but in the setting of this work a discrete eigenvalue problem results always from a finite element discretisation of a continuous eigenvalue problem. Hence, all computed eigenpairs of the discrete problem are related to a discretisation error. This means that, as long as the projection error caused by AMLS is of the same order as the discretisation error, the computed eigenpair approximations of AMLS are of comparable quality as the eigenpairs computed by block-SIL or some other classical approach.

Although AMLS has proven to be very efficient, one problem is the computation of the interface eigenvalue problems via dense matrix operations. In the three-dimensional case the complexity is dominated by this part.

In the following sections the AMLS method is described in detail. Although the method can be described in a purely algebraic way without any geometry information of the underlying partial differential equation the method is first explained, in Section 5.1, in a continuous setting. In the continuous setting it is easier to understand the idea behind AMLS and why the method is working. However, for reasons of simplification the discussion in the continuous setting is restricted to the single-level version of AMLS, for the description of the corresponding multi-level version it is referred to [16]. After that, in Section 5.2, the AMLS method is described in an algebraic setting to show how the method is applied in practice. For ease of understanding it is started with the description of the single-level version of AMLS which is extended to the multi-level version afterwards. Finally, in Section 5.3 it is outlined why the AMLS method is getting expensive for three-dimensional problems.

The AMLS method has been already described in [31, Section 3] by the author. This chapter is a strongly revised version of [31, Section 3] and provides a more detailed discussion of the topic.

5.1. The AMLS Method in the Continuous Setting

The initial point of AMLS in the continuous setting is the variational eigenvalue problem (4.2) which will be denoted as *global eigenvalue problem* in this particular section. The single-level version of the AMLS method, which is described in the following, is a generalisation of the classical CMS.

In the first step of AMLS the Lipschitz domain Ω is subdivided into two non-overlapping subdomains Ω_1 and Ω_2 such that both subdomains have as well a Lipschitz boundary. In particular, we have

$$\overline{\Omega} = \overline{\Omega}_1 \cup \overline{\Omega}_2 \qquad \text{and} \qquad \Omega_1 \cap \Omega_2 = \emptyset \tag{5.1}$$

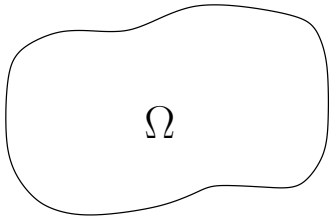
whereby the subdomains share the interface $\Gamma := \overline{\Omega}_1 \cap \overline{\Omega}_2$. Examples of such a domain substructuring are given in Figure 5.1 for a two-dimensional domain.

After this suitable subspaces of $H_0^1(\Omega)$ are defined which are associated with the subdomains Ω_i $(i = 1, 2)$ and the interface Γ: For the subdomains Ω_i the subspaces

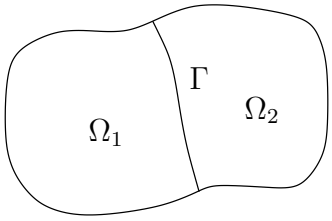
$$V_{\Omega_i} := \left\{ v \in H_0^1(\Omega) \ : \ v_{|\Omega \setminus \Omega_i} = 0 \right\}$$

are defined, which are built of all admissible functions that are equal to zero on $\Omega \setminus \Omega_i$. For the interface Γ the subspace

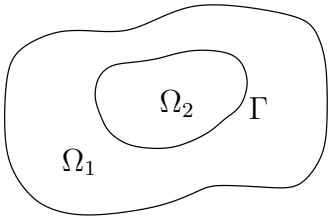
$$V_\Gamma := \left\{ E_\Omega \tau \ : \ \tau \in H_{00}^{1/2}(\Gamma) \right\}$$



(a) Underlying domain Ω of the PDE eigenvalue problem (4.1)



(b) Example of a possible domain substructuring.



(c) Example of a possible domain substructuring.

Figure 5.1.: Substructuring of the domain Ω into two non-overlapping subdomains Ω_1 and Ω_2 with interface $\Gamma := \overline{\Omega}_1 \cap \overline{\Omega}_2$.

is defined, where $H_{00}^{1/2}(\Gamma)$ denotes the trace space of $H_0^1(\Omega)$ on the interface Γ and $E_\Omega \tau \in H_0^1(\Omega)$ is the extension of the trace function $\tau \in H_{00}^{1/2}(\Gamma)$ which is defined as the unique solution of the variational problem

$$\begin{cases} \text{find } \; E_\Omega \tau \in \left\{ u \in H_0^1(\Omega) \,:\, u_{|\Gamma} = \tau \right\} \; \text{ such that} \\ a(E_\Omega \tau, v) \;=\; 0 \qquad \forall\, v \in \left\{ u \in H_0^1(\Omega) \,:\, u_{|\Gamma} = 0 \right\}. \end{cases} \tag{5.2}$$

Basic properties of the extension operator E_Ω are summarised in the following lemma.

Lemma 5.1 (Extension Operator) *The extension operator $E_\Omega : H_{00}^{1/2}(\Gamma) \to H_0^1(\Omega)$ is linear and injective. Furthermore, the operator is continuous, i.e., there exists a constant $\widehat{C} > 0$ such that*

$$\|E_\Omega \tau\|_1 \;\le\; \widehat{C}\, \|\tau\|_{H_{00}^{1/2}(\Gamma)} \qquad \text{for all } \tau \in H_{00}^{1/2}(\Gamma) \tag{5.3}$$

where $\|\cdot\|_{H_{00}^{1/2}(\Gamma)}$ is the the norm on $H_{00}^{1/2}(\Gamma)$ [cf. Remark B.20].

Proof: i) For the sake of completeness it is first proven that problem (5.2) has for each $\tau \in H_{00}^{1/2}(\Gamma)$ a unique solution $E_\Omega \tau$. Basic properties of trace spaces are presented in Remark B.20 [in Remark B.20 the trace space $H^{1/2}(\partial\Omega)$ is discussed, however, the analogue results hold as well for $H_{00}^{1/2}(\Gamma)$]. It holds that for all $\tau \in H_{00}^{1/2}(\Gamma)$ there exists some $\widetilde{\tau} \in H_0^1(\Omega)$ such that

$$\widetilde{\tau}_{|\Gamma} \;=\; \tau \qquad \text{and} \qquad \|\widetilde{\tau}\|_{H_0^1(\Omega)} \;\le\; C \|\tau\|_{H_{00}^{1/2}(\Gamma)} \tag{5.4}$$

with some constant $C > 0$ independent of τ. Furthermore, it holds that $E_\Omega \tau \in H_0^1(\Omega)$ is a solution of problem (5.2) iff $\widetilde{E}_\Omega \tau := E_\Omega \tau - \widetilde{\tau}$ is a solution of the variational problem

$$\begin{cases} \text{find } \; \widetilde{E}_\Omega \tau \in U_0 \; \text{ such that} \\ a(\widetilde{E}_\Omega \tau, v) = -a(\widetilde{\tau}, v) \quad \forall\, v \in U_0 \end{cases} \tag{5.5}$$

where $U_0 := \left\{ u \in H_0^1(\Omega) : u_{|\Gamma} = 0 \right\}$. Since $U_0 \subset H_0^1(\Omega)$ is a Hilbert space and $l(v) := -a(\widetilde{\tau}, v)$ defines a continuous linear functional on U_0, it follows from Theorem A.18 that problem (5.5) has a unique solution $\widetilde{E}_\Omega \tau \in U_0$. Thus, a solution of problem (5.2) exists as well. Moreover, if it is assumed that u_1 and u_2 are solutions of (5.2) it follows that $u_1 - u_2 \in U_0$ and $a(u_1 - u_2, v) = 0$ for all $v \in U_0$. Since $a(\cdot,\cdot)$ is $H_0^1(\Omega)$-elliptic we conclude that $u_1 - u_2 = 0$ in $H_0^1(\Omega)$ which proves the uniqueness of the solution $E_\Omega \tau$.

ii) The linearity of the operator $E_\Omega : H_{00}^{1/2}(\Gamma) \to H_0^1(\Omega)$ is proven using the uniqueness of the solution of problem (5.2), and the injectivity of E_Ω is obvious. Moreover, it holds $a(E_\Omega\tau, \widetilde{E}_\Omega\tau) = 0$ because of (5.2) and $\widetilde{E}_\Omega\tau \in U_0$, and it follows

$$\begin{aligned}\|E_\Omega\tau\|_1^2 &\underset{\substack{a(\cdot,\cdot) \text{ is}\\ \text{elliptic}}}{\leq} \frac{1}{C_E} a(E_\Omega\tau, E_\Omega\tau) \;=\; \frac{1}{C_E}\, a(E_\Omega\tau, \widetilde{E}_\Omega\tau + \widetilde{\tau}) \;=\; \frac{1}{C_E}\, a(E_\Omega\tau, \widetilde{\tau}) \\ &\underset{\substack{a(\cdot,\cdot) \text{ is}\\ \text{continuous}}}{\leq} \frac{C_B}{C_E}\, \|E_\Omega\tau\|_1 \|\widetilde{\tau}\|_1 \underset{(5.4)}{\leq} \frac{C\, C_B}{C_E}\, \|E_\Omega\tau\|_1 \|\tau\|_{H_{00}^{1/2}(\Gamma)}.\end{aligned}$$

Hence, for all $\tau \in H_{00}^{1/2}(\Gamma)$ there exists some $\widehat{C} > 0$ with $\|E_\Omega\tau\|_1 \leq \widehat{C}\, \|\tau\|_{H_{00}^{1/2}(\Gamma)}$, i.e., the extension operator E_Ω is continuous.

■

For the subspaces V_{Ω_1}, V_{Ω_2} and V_Γ defined above the following relation is valid:

Theorem 5.2 *The direct sum*

$$V_{\Omega_1} \oplus V_{\Omega_2} \oplus V_\Gamma \tag{5.6}$$

is an a-orthogonal decomposition of $H_0^1(\Omega)$.

Proof: The proof of this result is outlined in [16], however, for a better understanding of the AMLS theory the proof is performed here as well: Let $u_\Gamma \in V_\Gamma$ be given then there exists a $\tau \in H_{00}^{1/2}(\Gamma)$ with $u_\Gamma = E_\Omega\tau$. According to the definition of $E_\Omega\tau$ it holds $a(E_\Omega\tau, v) = 0$ for all $v \in U_0 := \left\{u \in H_0^1(\Omega) : u_{|\Gamma} = 0\right\}$. It follows that for all $u_i \in V_{\Omega_i}$ $(i = 1, 2)$ it holds $a(u_\Gamma, u_i) = 0$ since $V_{\Omega_i} \subset U_0$, and that $a(u_1, u_2) = 0$ since $\Omega_1 \cap \Omega_2 = \emptyset$, which proves the orthogonality statement. Furthermore, for $u \in H_0^1(\Omega)$ we define the functions

$$\mathcal{P}_\Gamma u := E_\Omega(u_{|\Gamma}), \quad \mathcal{P}_{\Omega_1} u := \begin{cases} u - \mathcal{P}_\Gamma u & \text{in } \Omega_1, \\ 0 & \text{in } \Omega_2 \end{cases} \quad \text{and} \quad \mathcal{P}_{\Omega_2} u := \begin{cases} 0 & \text{in } \Omega_1, \\ u - \mathcal{P}_\Gamma u & \text{in } \Omega_2. \end{cases}$$

From the definition of V_{Ω_i} and from the substructuring (5.1) of Ω it follows that $\mathcal{P}_\Gamma u \in V_\Gamma$ and $\mathcal{P}_{\Omega_i} u \in V_{\Omega_i}$. Finally, from the orthogonality result from above and from the identity

$$\mathcal{P}_\Gamma u + \mathcal{P}_{\Omega_1} u + \mathcal{P}_{\Omega_2} u = \mathcal{P}_\Gamma u + u - \mathcal{P}_\Gamma u = u$$

we conclude that (5.6) is valid. ■

In the second step of AMLS for each subspace separate eigenvalue problems are defined: For V_{Ω_i} $(i = 1, 2)$ the so-called *fixed-interface eigenvalue problem*

$$\begin{cases} \text{find } (\lambda^{\Omega_i}, u^{\Omega_i}) \in \mathbb{R} \times V_{\Omega_i} \setminus \{0\} \text{ such that} \\ a(u^{\Omega_i}, v) \;=\; \lambda^{\Omega_i}\, (u^{\Omega_i}, v)_0 \qquad \forall\, v \in V_{\Omega_i} \end{cases} \tag{5.7}$$

is defined, and for V_Γ the so-called *coupling mode eigenvalue problem*

$$\begin{cases} \text{find } (\lambda^\Gamma, u^\Gamma) \in \mathbb{R} \times V_\Gamma \setminus \{0\} \text{ such that} \\ a(u^\Gamma, v) \;=\; \lambda^\Gamma\, (u^\Gamma, v)_0 \qquad \forall\, v \in V_\Gamma. \end{cases} \tag{5.8}$$

Note that the only difference to the global eigenvalue problem (4.2) is that the functions u and v in (5.7) and (5.8) are elements of V_{Ω_i} or V_Γ instead of $H_0^1(\Omega)$. Since V_{Ω_i} and V_Γ are closed subspaces in $H_0^1(\Omega)$ it follows from Remark 2.7 that each of these problems possesses a countable family of eigensolutions which are given by

$$\big(\lambda_j^{\Omega_i}\,,\, u_j^{\Omega_i}\big)_{j=1}^{\infty} \in \mathbb{R}_{>0} \times V_{\Omega_i} \setminus \{0\} \qquad \text{with } \lambda_j^{\Omega_i} \leq \lambda_{j+1}^{\Omega_i}$$

for the fixed-interface eigenvalue problem (5.7) and by

$$\big(\lambda_j^{\Gamma}\,,\, u_j^{\Gamma}\big)_{j=1}^{\infty} \in \mathbb{R}_{>0} \times V_{\Gamma} \setminus \{0\} \qquad \text{with } \lambda_j^{\Gamma} \leq \lambda_{j+1}^{\Gamma}$$

for the coupling mode eigenvalue problem (5.8). The eigenfunctions (5.7) and (5.8) form a basis of V_{Ω_i} and V_Γ. Furthermore, from Theorem 5.2 it follows that the eigenfunctions of (5.7) and (5.8) are a-orthogonal to each other, and that they form a basis of $H_0^1(\Omega)$ with

$$H_0^1(\Omega) \;=\; \bigoplus_{i=1}^{2} \operatorname{span}\Big\{ u_j^{\Omega_i} \;:\; j \in \mathbb{N} \Big\} \;\oplus\; \operatorname{span}\Big\{ u_j^{\Gamma} \;:\; j \in \mathbb{N} \Big\}. \tag{5.9}$$

It is important to note that even if the eigensolutions of the problems (5.7) and (5.8) are known, the global eigenvalue problem (4.2) is not solved. However, the eigenfunctions of (5.7) and (5.8) belonging to the smallest eigenvalues are well suited to approximate the sought eigensolutions $(\lambda_j, u_j)_{j=1}^{n_{\text{es}}}$ of the global problem (4.2). This issue is reasoned by various numerical studies (see, e.g., [16]) and is motivated by the error analysis done in [21, 22] for a method quite similar to AMLS. Correspondingly, to approximate the sought eigensolutions of problem (4.2), in the third step of AMLS the finite dimensional subspace $U_{\Bbbk} \subset H_0^1(\Omega)$ is defined by

$$U_{\Bbbk} \;:=\; \bigoplus_{i=1}^{2} \operatorname{span}\Big\{ u_j^{\Omega_i} \;:\; j = 1, \ldots, k_i \Big\} \;\oplus\; \operatorname{span}\Big\{ u_j^{\Gamma} \;:\; j = 1, \ldots, k_\Gamma \Big\} \tag{5.10}$$

which is obtained by applying a modal truncation in (5.9) and selecting only those eigenfunctions which belong to the smallest k_1, k_2 and k_Γ eigenvalues for given $k_1, k_2, k_\Gamma \in \mathbb{N}$ and multi-index $\Bbbk := (k_1, k_2, k_\Gamma)$.

Using the finite dimensional subspace $U_{\Bbbk}$ the so-called *reduced eigenvalue problem*

$$\begin{cases} \text{find } \; (\lambda^{(\Bbbk)}, u^{(\Bbbk)}) \;\in\; \mathbb{R} \times U_{\Bbbk} \setminus \{0\} \;\text{ such that} \\ a(u^{(\Bbbk)}, v) \;=\; \lambda^{(\Bbbk)}\, (u^{(\Bbbk)}, v)_0 \qquad \forall\, v \in U_{\Bbbk} \end{cases} \tag{5.11}$$

is defined which possesses the eigensolutions

$$\big(\lambda_j^{(\Bbbk)}, u_j^{(\Bbbk)}\big)_{j=1}^{|\Bbbk|} \in \mathbb{R}_{>0} \times U_{\Bbbk} \setminus \{0\} \qquad \text{with} \quad \lambda_j^{(\Bbbk)} \leq \lambda_{j+1}^{(\Bbbk)}. \tag{5.12}$$

The reduced problem (5.11) is a conforming Ritz-Galerkin approximation of the global eigenvalue problem (4.2). According to Theorem 3.2 the eigensolutions of the reduced problem are converging to eigensolutions of the global eigenvalue problem when $k_1, k_2, k_\Gamma \to \infty$.

Correspondingly in the fourth and last step of the AMLS method the first n_{es} eigensolutions (5.12) are computed (with $n_{\text{es}} \leq |\Bbbk|$) which are approximating the sought eigensolutions $(\lambda_j, u_j)_{j=1}^{n_{\text{es}}}$ of the global problem (4.2).

Theorem 5.3 *The coupling mode eigenvalue problem* (5.8) *is equivalent to the eigenvalue problem*

$$\begin{cases} \text{find } (\lambda, u) \in \mathbb{R} \times H_{00}^{1/2}(\Gamma) \setminus \{0\} \text{ such that} \\ \langle \mathcal{S}u, v \rangle \;=\; \lambda \, \langle \mathcal{M}u, v \rangle \qquad \forall\, v \in H_{00}^{1/2}(\Gamma), \end{cases} \tag{5.13}$$

where $\mathcal{S}$ and $\mathcal{M}$ are operators acting on the trace space $H_{00}^{1/2}(\Gamma)$ which are given in strong form by

$$\mathcal{S}\tau \;:=\; \sum_{i=1}^{2} \big((A \nabla E_{\Omega_i} \tau) \cdot \mathbf{n}^i \big)_{|\Gamma} \qquad \text{and} \qquad \mathcal{M}\tau \;:=\; \sum_{i=1}^{2} - \big((A \nabla \mathcal{G}_{\Omega_i}(E_{\Omega_i}\tau)) \cdot \mathbf{n}^i \big)_{|\Gamma}$$

for $\tau \in H_{00}^{1/2}(\Gamma)$, and where $\langle \cdot, \cdot \rangle$ denotes the duality pairing between $H_{00}^{1/2}(\Gamma)$ and its dual space. Furthermore, $\mathbf{n}^i$ denotes the outward normal unit vector on Γ for the subdomain Ω_i, E_{Ω_i} is the subdomain extension operator defined by $E_{\Omega_i}\tau := (E_\Omega \tau)_{|\Omega_i}$, and for $f \in L^2(\Omega_i)$ the function $\mathcal{G}_{\Omega_i}(f)$ is defined as the solution of the variational problem

$$\begin{cases} \text{find} \quad \mathcal{G}_{\Omega_i}(f) \;\in\; H_0^1(\Omega_i) \quad \text{such that} \\ a(\mathcal{G}_{\Omega_i}(f), v)_{\Omega_i} = (f, v)_{\Omega_i} \quad \forall\, v \in H_0^1(\Omega_i) \end{cases} \tag{5.14}$$

with the restricted bilinear forms

$$\begin{aligned} a(u, v)_{\Omega_i} &:= \int_{\Omega_i} \nabla u^T A \nabla v + cuv \; \mathrm{d}x \qquad && \forall\, u, v \in H^1(\Omega_i), \\ (u, v)_{\Omega_i} &:= \int_{\Omega_i} uv \; \mathrm{d}x && \forall\, u, v \in L^2(\Omega_i). \end{aligned}$$

Properties of $\mathcal{S}$ and $\mathcal{M}$ are listed in the end of the following proof.

Proof: The proof of this theorem can be found in [16] in the context of an eigenvalue problem from linear elastodynamics. However, for a better understanding of the AMLS theory the proof is performed here as well for generic $H^1(\Omega)$-elliptic bilinear forms: First of all, it is noted that the coupling mode eigenvalue problem (5.8) is equivalent to the eigenvalue problem

$$\begin{cases} \text{find} \;\; (\lambda, \tau) \;\in\; \mathbb{R} \times H_{00}^{1/2}(\Gamma) \setminus \{0\} \;\; \text{such that} \\ a(E_\Omega \tau, E_\Omega \eta) \;=\; \lambda \, (E_\Omega \tau, E_\Omega \eta)_0 \qquad \forall\, \eta \in H_{00}^{1/2}(\Gamma) \end{cases} \tag{5.15}$$

since each element of V_Γ is determined by its trace on Γ.

Derivation of operator $\mathcal{S}$:

Note that for $\tau \in H_{00}^{1/2}(\Gamma)$ the subdomain extension $E_{\Omega_i}\tau := (E_\Omega \tau)_{|\Omega_i}$ is the unique solution[1] of the variational problem

$$\begin{cases} \text{find} \;\; E_{\Omega_i}\tau \in \big\{ u \in H^1(\Omega_i) \,:\, u_{|\Gamma} = \tau \text{ and } u_{|\partial\Omega_i \setminus \Gamma} = 0 \big\} \;\; \text{such that} \\ a(E_{\Omega_i}\tau, v)_{\Omega_i} \;=\; 0 \qquad \forall\, v \in H_0^1(\Omega_i). \end{cases} \tag{5.16}$$

[1] The existence of a unique solution for problem (5.16) is proven in the same way as for problem (5.2) [cf. part i) of the proof of Lemma 5.1].

Since $E_{\Omega_i}\tau$ is the solution of problem (5.16) we obtain for $v \in H^1(\Omega_i)$ the identity[2]

$$a(\, E_{\Omega_i}\tau, v\,)_{\Omega_i} \;=\; \int_{\partial\Omega_i} v_{|\partial\Omega_i}\,(\,(A\nabla E_{\Omega_i}\tau)\cdot\mathbf{n}^i\,)_{|\partial\Omega_i}\;\mathrm{d}s, \tag{5.17}$$

where the surface integral $\int_{\partial\Omega_i}\ldots\,\mathrm{d}s$ on the right-hand side of (5.17) is formally interpreted as a functional on $H^{1/2}_{00}(\Gamma)$. Correspondingly, for $\tau,\eta\in H^{1/2}_{00}(\Gamma)$ it follows that

$$\begin{aligned} a(\, E_\Omega\tau, E_\Omega\eta\,) \;&=\; \sum_{i=1}^{2} a(\, E_{\Omega_i}\tau, E_{\Omega_i}\eta\,)_{\Omega_i} \underset{(5.17)}{=} \sum_{i=1}^{2}\int_{\partial\Omega_i}(E_{\Omega_i}\eta)_{|\partial\Omega_i}\,(\,(A\nabla E_{\Omega_i}\tau)\cdot\mathbf{n}^i\,)_{|\partial\Omega_i}\;\mathrm{d}s \\ &=\; \sum_{i=1}^{2}\int_{\Gamma}\eta\,(\,(A\nabla E_{\Omega_i}\tau)\cdot\mathbf{n}^i\,)_{|\Gamma}\;\mathrm{d}s \;=\; \sum_{i=1}^{2}\langle\,(A\nabla E_{\Omega_i}\tau)\cdot\mathbf{n}^i,\eta\,\rangle \;=\; \langle\mathcal{S}\tau,\eta\rangle. \end{aligned}$$

Derivation of operator $\mathcal{M}$:

Since $\mathcal{G}_{\Omega_i}(f)$ is the solution of problem (5.14) we obtain for $v\in H^1(\Omega_i)$ the identity[3]

$$a(\,\mathcal{G}_{\Omega_i}(f), v\,)_{\Omega_i} \;=\; (\,f, v\,)_{\Omega_i} + \int_{\partial\Omega_i} v_{|\partial\Omega_i}\,(\,(A\nabla\mathcal{G}_{\Omega_i}(f))\cdot\mathbf{n}^i\,)_{|\partial\Omega_i}\;\mathrm{d}s, \tag{5.18}$$

where the surface integral $\int_{\partial\Omega_i}\ldots\,\mathrm{d}s$ on the right-hand side of (5.18) is formally interpreted as a functional on $H^{1/2}_{00}(\Gamma)$. Using $f := E_{\Omega_i}\tau$ and $v := E_{\Omega_i}\eta$ in (5.18) we conclude that for $\tau,\eta\in H^{1/2}_{00}(\Gamma)$ it holds

$$a(\,\mathcal{G}_{\Omega_i}(E_{\Omega_i}\tau), E_{\Omega_i}\eta\,)_{\Omega_i} = (\,E_{\Omega_i}\tau, E_{\Omega_i}\eta\,)_{\Omega_i} + \int_{\partial\Omega_i}(E_{\Omega_i}\eta)_{|\partial\Omega_i}\,(\,(A\nabla\mathcal{G}_{\Omega_i}(E_{\Omega_i}\tau))\cdot\mathbf{n}^i\,)_{|\partial\Omega_i}\;\mathrm{d}s$$

which leads to

$$\begin{aligned} (\,E_\Omega\tau, E_\Omega\eta\,)_0 \;&=\; \sum_{i=1}^{2}(\,E_{\Omega_i}\tau, E_{\Omega_i}\eta\,)_{\Omega_i} \\ &=\; \sum_{i=1}^{2} a(\,\mathcal{G}_{\Omega_i}(E_{\Omega_i}\tau), E_{\Omega_i}\eta\,)_{\Omega_i} - \int_{\partial\Omega_i}(E_{\Omega_i}\eta)_{|\partial\Omega_i}\,(\,(A\nabla\mathcal{G}_{\Omega_i}(E_{\Omega_i}\tau))\cdot\mathbf{n}^i\,)_{|\partial\Omega_i}\;\mathrm{d}s \\ &=\; \sum_{i=1}^{2} a(\,\mathcal{G}_{\Omega_i}(E_{\Omega_i}\tau), E_{\Omega_i}\eta\,)_{\Omega_i} - \int_{\Gamma}(E_{\Omega_i}\eta)_{|\Gamma}\,(\,(A\nabla\mathcal{G}_{\Omega_i}(E_{\Omega_i}\tau))\cdot\mathbf{n}^i\,)_{|\Gamma}\;\mathrm{d}s. \end{aligned}$$

[2]Identity (5.17) is proven using the following result: Let $u\in H^1_0(\Omega)$ be the solution of the variational problem $a(u,v)=(f,v)_0$ for all $v\in H^1_0(\Omega)$ with some given $f\in L^2(\Omega)$. In the case that the coefficients of the associated PDE operator L fulfil $a_{ij}\in H^1(\Omega)$ and if $u\in H^2(\Omega)$ holds, then we obtain by partial integration (P.I.) that

$$(f,v)_0 \;=\; \int_\Omega fv\;\mathrm{d}x \;=\; \int_\Omega Lu\,v\;\mathrm{d}x \overset{\text{P.I.}}{=} a(u,v) - \int_{\partial\Omega} v_{|\partial\Omega}\,(\,(A\nabla u)\cdot\mathbf{n}\,)_{|\partial\Omega}\;\mathrm{d}s \qquad \forall\, v\in H^1(\Omega)$$

where $\mathbf{n}$ denotes the outward normal unit vector on $\partial\Omega$. Using this result, it can be shown that also in the case when only $a_{ij}\in L^2(\Omega)$ and $u\in H^1(\Omega)$ is valid, that then the functional $l(v) := a(u,v)-(f,v)_0$ fulfils the equality $l(v)=\int_{\partial\Omega} v_{|\partial\Omega}((A\nabla u)\cdot\mathbf{n})_{|\partial\Omega}\;\mathrm{d}s$ for all $v\in H^1(\Omega)$ where the surface integral is formally interpreted as a functional on $H^{1/2}(\partial\Omega)$.

[3]Identity (5.18) is derived in the same way as identity (5.17).

Furthermore, using the function

$$\mathcal{G}_i(E_{\Omega_i}\tau) := \begin{cases} \mathcal{G}_{\Omega_i}(E_{\Omega_i}\tau) & \text{in } \Omega_i, \\ 0 & \text{in } \Omega \setminus \Omega_i \end{cases}$$

we obtain from the orthogonality result of Theorem 5.2 that

$$a(\, \mathcal{G}_{\Omega_i}(E_{\Omega_i}\tau), E_{\Omega_i}\eta \,)_{\Omega_i} \;=\; a(\, \mathcal{G}_i(E_{\Omega_i}\tau), E_\Omega\eta \,) \;=\; 0$$

since $\mathcal{G}_i(E_{\Omega_i}\tau) \in V_{\Omega_i}$ and $E_\Omega\eta \in V_\Gamma$. Altogether, we obtain

$$(\, E_\Omega\tau, E_\Omega\eta \,)_0 \;=\; \sum_{i=1}^{2} - \int_\Gamma \eta \; (\, (A\nabla\mathcal{G}_{\Omega_i}(E_{\Omega_i}\tau)) \cdot \mathbf{n}^i \,)_{|\Gamma} \; \mathrm{d}s \;=\; \langle \mathcal{M}\tau, \eta \rangle.$$

Properties of $\mathcal{S}$ and $\mathcal{M}$:

$\mathcal{S}$ and $\mathcal{M}$ are linear operators of the form $H_{00}^{1/2}(\Gamma) \to H^{-1/2}(\Gamma)$ where $H^{-1/2}(\Gamma)$ denotes the dual space of $H_{00}^{1/2}(\Gamma)$. Since the variational representation of $\mathcal{S}$ and $\mathcal{M}$ is given by

$$\langle \mathcal{S}\tau, \eta \rangle = a(\, E_\Omega\tau, E_\Omega\eta \,) \quad \text{and} \quad \langle \mathcal{M}\tau, \eta \rangle = (\, E_\Omega\tau, E_\Omega\eta \,)_0 \qquad \text{for } \tau, \eta \in H_{00}^{1/2}(\Gamma)$$

it follows that the operators $\mathcal{S}$ and $\mathcal{M}$ (and the bilinear forms associated to $\mathcal{S}$ and $\mathcal{M}$) are selfadjoint (symmetric). From

$$|\langle \mathcal{S}\tau, \eta \rangle| = |a(\, E_\Omega\tau, E_\Omega\eta \,)| \underset{\substack{a(\cdot,\cdot) \text{ is} \\ \text{continuous}}}{\leq} C_B \|E_\Omega\tau\|_1 \|E_\Omega\eta\|_1 \underset{(5.3)}{\leq} \widehat{C}^2 \, C_B \|\tau\|_{H_{00}^{1/2}(\Gamma)} \|\eta\|_{H_{00}^{1/2}(\Gamma)}$$

and

$$|\langle \mathcal{M}\tau, \eta \rangle| = |(\, E_\Omega\tau, E_\Omega\eta \,)_0| \leq \|E_\Omega\tau\|_0 \|E_\Omega\eta\|_0 \leq \|E_\Omega\tau\|_1 \|E_\Omega\eta\|_1 \underset{(5.3)}{\leq} \widehat{C}^2 \, \|\tau\|_{H_{00}^{1/2}(\Gamma)} \|\eta\|_{H_{00}^{1/2}(\Gamma)}$$

we conclude that the operators $\mathcal{S}$ and $\mathcal{M}$ are continuous on $H_{00}^{1/2}(\Gamma)$, and that the associated bilinear forms are continuous on $H_{00}^{1/2}(\Gamma) \times H_{00}^{1/2}(\Gamma)$. Basic properties of trace spaces are discussed in Remark B.20 [in Remark B.20 the trace space $H^{1/2}(\partial\Omega)$ is discussed, however, the analogue results hold as well for $H_{00}^{1/2}(\Gamma)$]. In particular, the trace operator $\gamma : H_0^1(\Omega) \to H_{00}^{1/2}(\Gamma)$ is continuous, i.e., there exists a constant $C > 0$ such that $\|\gamma(u)\|_{H_{00}^{1/2}(\Gamma)} \leq C\|u\|_1$ for all $u \in H_0^1(\Omega)$. For $\tau \in H_{00}^{1/2}(\Gamma)$ we have

$$\langle \mathcal{S}\tau, \tau \rangle \;=\; a(\, E_\Omega\tau, E_\Omega\tau \,) \underset{\substack{a(\cdot,\cdot) \text{ is} \\ \text{elliptic}}}{\geq} C_E \, \|E_\Omega\tau\|_1^2 \;\geq\; \frac{C_E}{C^2} \, \|\tau\|_{H_{00}^{1/2}(\Gamma)}^2$$

and conclude that $\mathcal{S}$ (respectively, the associated bilinear form) is $H_{00}^{1/2}(\Gamma)$-elliptic. Furthermore, from the injectivity of the extension operator E_Ω (cf. Lemma 5.1) we conclude that $E_\Omega\tau \neq 0$ for all $\tau \in H_{00}^{1/2}(\Gamma) \setminus \{0\}$. It follows that for $\tau \in H_{00}^{1/2}(\Gamma) \setminus \{0\}$ it holds

$$\langle \mathcal{M}\tau, \tau \rangle \;=\; (\, E_\Omega\tau, E_\Omega\tau \,)_0 \;=\; \|E_\Omega\tau\|_0^2 > 0,$$

i.e., the operator $\mathcal{M}$ (and the associated bilinear form) is positive definite. ∎

Remark 5.4 *i) $\mathcal{S}$ is the so-called* Steklov-Poincaré operator *which is determined by the bilinear form $a(\cdot,\cdot)$. The Steklov-Poincaré operator is known, for example, from domain decomposition methods for elliptic boundary value problems. See [60] for further informations. The operator $\mathcal{M}$ has been firstly described in [16] and is called* mass operator*. This operator is determined by both the bilinear form $a(\cdot,\cdot)$ and the bilinear form $(\cdot,\cdot)_0$.*

ii) The operators $\mathcal{S}$ and $\mathcal{M}$ are symmetric and continuous on $H_{00}^{1/2}(\Gamma)$, moreover, $\mathcal{S}$ is elliptic and $\mathcal{M}$ is positive definite (cf. proof of Theorem 5.3). These properties (respectively, the properties of the associated bilinear forms) are important for the numerical solution of the coupling mode eigenvalue problem (5.13)*.*

iii) The fixed-interface eigenvalue problem (5.7) *is equivalent to the eigenvalue problem*

$$\begin{cases} \text{find } (\lambda, u) \in \mathbb{R} \times H_0^1(\Omega_i) \setminus \{0\} \text{ such that} \\ a(u,v) = \lambda\,(u,v)_0 \qquad \forall\, v \in H_0^1(\Omega_i). \end{cases} \tag{5.19}$$

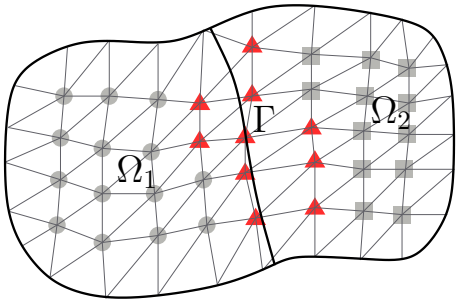
iv) The benefit of the representation (5.19) *and* (5.13) *compared to the representation* (5.7) *and* (5.8) *is that the eigenvalue problems are solely solved and evaluated on the subdomains Ω_i respectively the interface Γ.*

In this section we have seen that, in order to solve the global eigenvalue problem, the domain Ω is subdivided into two subdomains which are separated by an interface. On the subdomains and on the interface suitable eigenvalue problems are defined which, however, do not solve the global problem but whose eigenfunctions are well suited to approximate the sought eigensolutions of the global problem. In particular eigenfunctions belonging to the smallest eigenvalues are selected from each subproblem to form a suitable subspace which is used for a Ritz-Galerkin approximation of the global problem. Finally, we obtain from the resulting reduced eigenvalue problem approximations of the sought eigensolutions of the global problem.

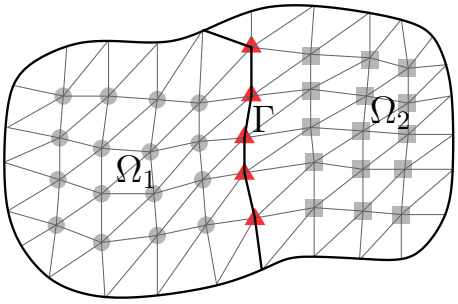
Remark 5.5 (Modal Truncation) *To apply AMLS in practice the fixed-interface eigenvalue problems* (5.19) *and the coupling mode eigenvalue problem* (5.13) *have to be discretised, e.g., by the finite element method, and eigensolutions of* (5.19) *and* (5.13) *associated to the smallest eigenvalues are approximated. As already mentioned, the modal truncation performed in* (5.10) *is motivated by theoretical and numerical results. However, there is a further reason for the performed modal truncation: According to the approximation properties of finite element spaces (see Section 3.4) only the first $j_{\max}^{\mathrm{EF}}(N_h) \ll N_h$ eigenfunctions can be well approximated by a given finite element space V_h with $N_h = \dim V_h$* degrees of freedom *(short DOF). For example, under certain smoothness conditions on the data (cf. Corollary 3.17) the number of well approximable eigenfunctions is given, e.g., only by $j_{\max}^{\mathrm{EF}}(N_h) = CN_h^{1/3}$ or $j_{\max}^{\mathrm{EF}}(N_h) = CN_h^{1/4}$. This issue motivates the modal truncation in* (5.10) *from another point of view.*

5.2. The AMLS Method in the Algebraic Setting

In this section the AMLS method is described in the algebraic setting to show how the method is applied in practice. The initial point is the algebraic eigenvalue problem (4.4) which is the finite element discretisation of the continuous problem (4.2). For reasons of convenience the upper



(a) DOF are associated to the interface Γ when their basis functions have supports that are intersecting the interface Γ.



(b) Interface Γ is chosen in such a way that it does not cut any finite element of the triangulation of Ω.

Figure 5.2.: Single-level substructuring of the domain Ω with triangulation. DOF of the FEM space $\mathbb{X}^1_{h,0}$ are indicated by circles if they are associated to Ω_1, by squares if associated to Ω_2, and by triangles if associated to Γ.

index of $\lambda^{(h)}, x^{(h)}, K^{(h)}$ and $M^{(h)}$, which is indicating in (4.4) the underlying mesh width h of the finite element discretisation, is left out in this particular section and the eigenvalue problem

$$\begin{cases} \text{find } (\lambda, x) \in \mathbb{R} \times \mathbb{R}^N \setminus \{0\} \text{ with} \\ K\,x \;=\; \lambda\, M\,x \end{cases} \tag{5.20}$$

is considered with the eigenpairs

$$\left(\lambda_j, x_j\right)_{j=1}^{N} \in \mathbb{R}_{>0} \times \mathbb{R}^N \setminus \{0\} \qquad \text{and} \quad \lambda_j \le \lambda_{j+1}$$

where $N := N_h = \dim V_h$. To avoid misunderstandings, it is explicitly noted that in this section λ and λ_j are interpreted as the eigenvalues of the discrete problem (5.20) and not as the eigenvalues of the continuous problem (4.2).

The description starts in Section 5.2.1 with AMLS in the single-level version and is extended in Section 5.2.2 to the multi-level version. In Section 5.2.3 a recursive version of AMLS is presented, which has been first described in [31] by the author (to the best of his knowledge), and which reduces the computational costs of the classical (multi-level) AMLS version.

5.2.1. Single-Level Version

In the first step of AMLS the domain Ω is substructured into two non-overlapping subdomains Ω_1 and Ω_2 as done in (5.1) where the subdomains share the interface $\Gamma := \overline{\Omega}_1 \cap \overline{\Omega}_2$. Since the matrices $K \in \mathbb{R}^{N\times N}$ and $M \in \mathbb{R}^{N\times N}$ in (5.20) result from a finite element discretisation each row and column index is associated with a basis function which has typically a small support (see, e.g., Figure 3.2). Using the substructuring of Ω the row and column indices of K and M are reordered in such a way that

$$K = \begin{array}{c} \\ \Omega_1 \\ \Omega_2 \\ \Gamma \end{array} \begin{array}{c} \begin{array}{ccc} \Omega_1 & \Omega_2 & \Gamma \end{array} \\ \begin{bmatrix} K_{11} & & K_{13} \\ & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix} \end{array} \quad \text{and} \quad M = \begin{array}{c} \\ \Omega_1 \\ \Omega_2 \\ \Gamma \end{array} \begin{array}{c} \begin{array}{ccc} \Omega_1 & \Omega_2 & \Gamma \end{array} \\ \begin{bmatrix} M_{11} & & M_{13} \\ & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \end{array} \tag{5.21}$$

holds with $K_{ij}, M_{ij} \in \mathbb{R}^{N_i \times N_j}$ and $N_1 + N_2 + N_3 = N$. The labels Ω_1, Ω_2 and Γ in (5.21) are indicating to which subset the indices are associated, i.e., if the supports of the corresponding basis functions are inside Ω_i or intersecting Γ [cf. Figure 5.2(a)].

Remark 5.6 *It is more convenient to substructure Ω in such a way that the resulting interface does not cut any finite element of the triangulation $\mathcal{T}_h$ of $\overline{\Omega}$ [cf. Figure 5.2(b)]. This minimizes the number of DOF associated to the interface, leading to smaller submatrices in* (5.21) *that are associated to Γ, and correspondingly reduces the computational costs when the problem (K, M) is transformed in the next steps of AMLS. Furthermore, when the interface Γ does not cut the triangulation $\mathcal{T}_h$ of $\overline{\Omega}$ then the substructuring induces triangulations $\mathcal{T}_{h,1}$ of $\overline{\Omega}_1$ and $\mathcal{T}_{h,2}$ of $\overline{\Omega}_2$ that share the same edges on Γ [cf. Figure 5.2(b)]. In particular, it follows that the eigenvalue problems (K_{11}, M_{11}) and (K_{22}, M_{22}) are the discrete equivalent of the continuous eigenvalue problems* (5.19)*, which eases the proof that AMLS in the algebraic setting is the discrete equivalent of AMLS in the continuous setting (see Section 5.1).*

In the next step of AMLS a block $\mathrm{LDL}^{\mathrm{T}}$-decomposition is performed in order to block diagonalise the matrix K by $K = L\widetilde{K}L^T$ where

$$L := \begin{bmatrix} \mathrm{Id} & & \\ & \mathrm{Id} & \\ K_{31}K_{11}^{-1} & K_{32}K_{22}^{-1} & \mathrm{Id} \end{bmatrix} \in \mathbb{R}^{N \times N} \qquad \text{and} \qquad \widetilde{K} = \operatorname{diag}\big[K_{11}, K_{22}, \widetilde{K}_{33}\big]. \tag{5.22}$$

The submatrix $\widetilde{K}_{33}$ given by

$$\widetilde{K}_{33} = K_{33} - K_{31}K_{11}^{-1}K_{13} - K_{32}K_{22}^{-1}K_{23}$$

is the *Schur complement* of $\operatorname{diag}[K_{11}, K_{22}]$ in K and it is typically dense. The matrix M is transformed correspondingly by computing $\widetilde{M} := L^{-1}ML^{-T}$ with

$$\widetilde{M} = \begin{bmatrix} M_{11} & & \widetilde{M}_{13} \\ & M_{22} & \widetilde{M}_{23} \\ \widetilde{M}_{31} & \widetilde{M}_{32} & \widetilde{M}_{33} \end{bmatrix} \tag{5.23}$$

where the submatrices of $\widetilde{M}$ are given by

$$\widetilde{M}_{3i} \;=\; M_{3i} - K_{3i}K_{ii}^{-1}M_{ii}, \qquad \text{and} \qquad \widetilde{M}_{i3} \;=\; \widetilde{M}_{3i}^T \qquad \text{for } i = 1, 2,$$

and

$$\widetilde{M}_{33} \;=\; M_{33} - \sum_{i=1}^{2} \big(K_{3i}K_{ii}^{-1}M_{i3} + M_{3i}K_{ii}^{-1}K_{i3} - K_{3i}K_{ii}^{-1}M_{ii}K_{ii}^{-1}K_{i3} \big).$$

A part of the sparsity structure is lost in $\widetilde{K}$ and $\widetilde{M}$. All submatrices $\widetilde{K}_{ii}$ and $\widetilde{M}_{ij}$ whose row or column indices are associated with the interface Γ are now typically dense. The eigenvalue problems (K, M) and $(\widetilde{K}, \widetilde{M})$ are equivalent, i.e., the eigenvalues of both problems are equal and if $\widetilde{x}$ is an eigenvector of $(\widetilde{K}, \widetilde{M})$ then $x = L^{-T}\widetilde{x}$ is an eigenvector of (K, M).

At first glance, the reason for the performed eigenvalue problem transformation from (K, M) to $(\widetilde{K}, \widetilde{M})$ is not obvious. But it can be shown, cf. [16] and [60, Section 2], that the eigenvalue problem $(\widetilde{K}_{33}, \widetilde{M}_{33})$ is the discrete equivalent of the continuous coupling mode eigenvalue

problem (5.13), and that the eigenvalue problems (K_{11}, M_{11}) and (K_{22}, M_{22}) are the discrete equivalents of the continuous fixed-interface problems (5.19). The initial point of the corresponding proof is the triangulation of Ω described in Remark 5.6 and Figure 5.2(b).

As in the continuous setting the global eigenvalue problem (K, M), respectively $(\widetilde{K}, \widetilde{M})$, is not solved just by computing the eigensolution of the subproblems (K_{11}, M_{11}), (K_{22}, M_{22}) and $(\widetilde{K}_{33}, \widetilde{M}_{33})$. However, the eigenvectors of these three subproblems are well suited to approximate the sought eigenvectors of (K, M) and $(\widetilde{K}, \widetilde{M})$. As in the continuous setting, cf. (5.10), only those subproblem eigenvectors are of interest which belong to the smallest eigenvalues.

Correspondingly in the next step of AMLS partial eigensolutions of the subproblems are computed, i.e., only those eigenpairs of (K_{11}, M_{11}), (K_{22}, M_{22}) and $(\widetilde{K}_{33}, \widetilde{M}_{33})$ are computed which belong to the smallest $k_i \in \mathbb{N}$ eigenvalues for given $k_i \leq N_i$ and $i = 1, 2, 3$. In the following these partial eigensolutions are

$$K_{ii}\,\widetilde{S}_i \;=\; M_{ii}\,\widetilde{S}_i\,\widetilde{D}_i \quad \text{for } i = 1, 2 \qquad \text{and} \qquad \widetilde{K}_{33}\,\widetilde{S}_3 \;=\; \widetilde{M}_{33}\,\widetilde{S}_3\,\widetilde{D}_3 \tag{5.24}$$

where the diagonal matrix $\widetilde{D}_i \in \mathbb{R}^{k_i \times k_i}$ contains the k_i smallest eigenvalues and the matrix $\widetilde{S}_i \in \mathbb{R}^{N_i \times k_i}$ column-wise the associated eigenvectors $(i = 1, 2, 3)$. Furthermore, the eigenvectors of the subproblems are normalised by $\widetilde{S}_i^T M_{ii} \widetilde{S}_i = \mathrm{Id}$ $(i = 1, 2)$ and $\widetilde{S}_3^T \widetilde{M}_{33} \widetilde{S}_3 = \mathrm{Id}$.

Remark 5.7 (Mode Selection) *How many eigenvectors have to be selected in* (5.24) *from each subproblem is not easy to answer. On the one hand enough spectral information has to be kept to obtain sufficiently good eigenpair approximations from the reduced problem. Selecting all (discrete) eigenvectors from each subproblem would lead to exact eigenpairs of the discrete global eigenvalue problem* (K, M)*. On the other hand* k_i *should be small to obtain in the further proceeding of AMLS a reduced problem of small size which can be easily solved.*

In the literature [28, 67] several heuristic approaches have been derived on how to select eigenpairs. These heuristics are based purely on the analysis of the algebraic eigenvalue problem $(\widetilde{K}, \widetilde{M})$ *without using any geometry information of the underlying partial differential equation* (4.1)*. One possible strategy for the eigenpair selection in* (5.24) *is as follows: Select in each subproblem only those eigenpairs whose eigenvalues are smaller than a given* truncation bound $\omega > 0$*.*

In this work a different approach is pursued. As already described the three subproblems (K_{11}, M_{11})*,* (K_{22}, M_{22}) *and* $(\widetilde{K}_{33}, \widetilde{M}_{33})$ *correspond to finite element discretisations of the continuous problems* (5.19) *and* (5.13)*. Therefore and because of the discussed approximation properties of finite element spaces (see Chapter 4 for summary), all eigenvectors in* (5.24) *are computed which still lead to reasonable approximations of the corresponding continuous eigenfunctions. Correspondingly only the eigenvectors belonging, e.g., to the smallest*

$$k_i = C N_i^{1/3} \in \mathbb{N} \quad \text{or} \quad k_i = C N_i^{1/2} \in \mathbb{N} \tag{5.25}$$

eigenvalues are computed with some constant $C > 0$*. The size of the constant* C *and the used exponent in* (5.25) *depend, among other things, on the spatial dimension* d*, on the polynomial degree* p *of the used finite element space* V_h*, on the type of the subproblem (subdomain or interface eigenvalue problem), and on the number* n_{es} *of sought eigenpairs. The constant* C *and the exponent from* (5.25) *will be specified in Chapter 9 where numerical results are presented. The results of Corollary 3.17 where asymptotics on the number of well approximable eigenfunctions are presented can be used as orientation, however, it might be useful to select slightly more*

eigenvectors which are possibly bad approximations of continuous eigenfunctions but still have enough spectral information to approximate the associated eigenvalue.

In the next step the block diagonal matrix

$$Z := \operatorname{diag}\left[\widetilde{S}_1, \widetilde{S}_2, \widetilde{S}_3\right] \in \mathbb{R}^{N\times\bar{k}} \qquad \text{with } \bar{k} := k_1 + k_2 + k_3 \ll N$$

is defined which is built of all selected subproblem eigenvectors. The $\bar{k}$-dimensional subspace spanned by the columns of the matrix Z respectively of the matrix $L^{-T}Z$ is well suited to approximate the sought eigenvectors of $(\widetilde{K}, \widetilde{M})$ respectively (K, M). In particular, the columns of the matrix

$$L^{-T}Z = \begin{bmatrix} \widetilde{S}_1 & & -(K_{11}^{-1}K_{13})\widetilde{S}_3 \\ & \widetilde{S}_2 & -(K_{22}^{-1}K_{23})\widetilde{S}_3 \\ & & \widetilde{S}_3 \end{bmatrix} \in \mathbb{R}^{N\times\bar{k}}$$

are the discrete equivalent of the selected eigenfunctions contained in subspace (5.10) from the continuous setting (cf. [16, 60]). To be more precise, the columns vectors in the first and second block column of the matrix $L^{-T}Z$ are the discrete analogue of the selected eigenfunctions from the continuous problems (5.7), and the column vectors in the third block-column of $L^{-T}Z$ are the discrete analogue of the selected eigenfunctions from the continuous problem (5.8).

In order to approximate the sought eigenpairs of (K, M) in the next step of AMLS, the matrices $\widehat{K} := Z^T\,\widetilde{K}\,Z$ and $\widehat{M} := Z^T\,\widetilde{M}\,Z$ are computed where it holds

$$\widehat{K} = \operatorname{diag}\left[\widetilde{D}_1, \widetilde{D}_2, \widetilde{D}_3\right] \in \mathbb{R}^{\bar{k}\times\bar{k}} \qquad \text{and} \qquad \widehat{M} = \begin{bmatrix} \mathrm{Id} & & \widehat{M}_{13} \\ & \mathrm{Id} & \widehat{M}_{23} \\ \widehat{M}_{31} & \widehat{M}_{32} & \mathrm{Id} \end{bmatrix} \in \mathbb{R}^{\bar{k}\times\bar{k}},$$

and a *reduced eigenvalue problem*

$$\begin{cases} \text{find } (\widehat{\lambda}, \widehat{x}\,) \in \mathbb{R}\times\mathbb{R}^{\bar{k}} \setminus \{0\} \text{ with} \\ \widehat{K}\,\widehat{x} \;=\; \widehat{\lambda}\,\widehat{M}\,\widehat{x} \end{cases} \tag{5.26}$$

with eigenpairs

$$\big(\widehat{\lambda}_j, \widehat{x}_j\big)_{j=1}^{\bar{k}} \in \mathbb{R}_{>0}\times\mathbb{R}^{\bar{k}} \setminus \{0\} \qquad \text{and} \quad \widehat{\lambda}_j \le \widehat{\lambda}_{j+1}$$

is obtained. In particular, the reduced eigenvalue problem (5.26) is the discrete equivalent of the reduced problem (5.11) from the continuous setting.

At the end of AMLS the smallest n_{es} eigenpairs of (5.26) are computed. The vectors

$$\widehat{y}_j := L^{-T}Z\,\widehat{x}_j \qquad \text{with } j = 1, \ldots, \bar{k} \tag{5.27}$$

are Ritz-vectors of the original eigenvalue problem (K, M) respective to the subspace spanned by the columns of the matrix $L^{-T}Z$, and $\widehat{\lambda}_j$ are the respective Ritz-values. Finally, the pairs

$$\big(\widehat{\lambda}_j, \widehat{y}_j\big)_{j=1}^{n_{\mathrm{es}}} \in \mathbb{R}_{>0}\times\mathbb{R}^{N} \setminus \{0\} \qquad \text{with} \quad \widehat{\lambda}_j \le \widehat{\lambda}_{j+1} \tag{5.28}$$

are approximating the sought smallest n_{es} eigenpairs of the eigenvalue problem (K, M). But note that in the beginning of AMLS the original index ordering of the matrices K and M has

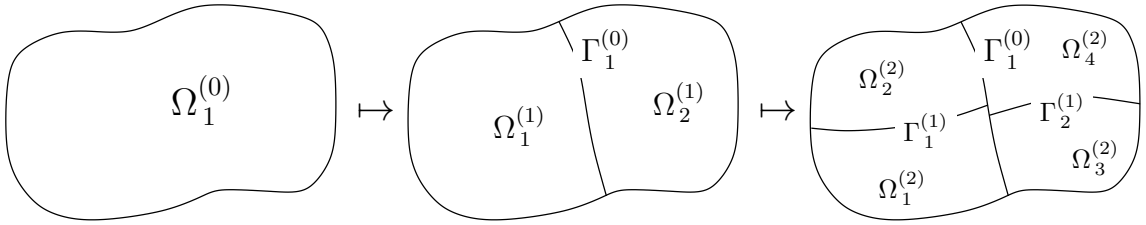


Figure 5.3.: Extending the single-level substructuring of $\Omega_1^{(0)} := \Omega$ to a two-level substructuring.

been changed by some index permutation $\pi : I \to I$ with $I := \{1, \ldots, N\}$ in order to obtain the block-structure form (5.21). To obtain eigenpair approximations of (K, M) with the original ordering of the row and column indices the inverse index permutation π^{-1} has to be applied to the vectors $\widehat{y}_j$ for $j = 1, \ldots, n_{\text{es}}$.

Remark 5.8 (Reduced Eigenvalue Problem) *Because the eigenpairs of the reduced eigenvalue problem $(\widehat{K}, \widehat{M})$ are primarily used to approximate the eigensolutions of the continuous problem* (4.2) *and not the eigenpairs of the discretised problem (K, M), the approximation error of AMLS is influenced by the finite element discretisation and the modal truncation applied in* (5.24)*. As long as the error caused by the modal truncation is of the same order as the discretisation error, the eigenpair approximations derived from the reduced problem $(\widehat{K}, \widehat{M})$ are of comparable quality as the eigenpair approximations derived from the problem (K, M).*

The reduced eigenvalue problem $(\widehat{K}, \widehat{M})$ is much easier to solve than the original eigenvalue problem (K, M) because the number of selected eigenpairs in (5.24) *is typically quite small and therefore the order of the reduced problem is much smaller than the order of the original problem. If for example the mode selection strategy described in Remark 5.7 is used with $k_i = CN_i^{1/3}$ then the size of the reduced problem can be bounded by $\mathcal{O}(N^{1/3})$ and the problem can be solved by dense linear algebra routines in $\mathcal{O}(N)$.*

5.2.2. Multi-Level Version

The single-level version of the AMLS method explained in the previous section can easily be extended to a multi-level version. Using the substructuring from the single-level version we recursively subdivide the subdomains Ω_1 and Ω_2 into additional levels. For this purpose a more appropriate notation is introduced: The subdomain on level $l \in \mathbb{N}_0$ with numbering index $j \in \mathbb{N}$ is denoted by $\Omega_j^{(l)}$ where $\Omega_1^{(0)}$ corresponds to the global domain Ω. Each subdomain $\Omega_j^{(l)}$ is further subdivided into two non-overlapping subdomains on level $l + 1$ which share the interface $\Gamma_j^{(l)}$. This substructuring can be applied recursively to the resulting subdomains until a certain level is exceeded or the size of the subdomains falls below some given limit. This type of domain substructuring is also known as *nested dissection*. In Figure 5.3 the described domain substructuring has been illustrated until the level $l = 2$ is reached. The performed substructuring results in a subdivision of Ω into m_{dom} subdomains which are separated by m_{int} interfaces (for the two-level substructuring from Figure 5.3 we have $m_{\text{dom}} = 4$ and $m_{\text{int}} = 3$). The subdomains and interfaces will constitute in the following, similar to the single-level version of AMLS, in total $m = m_{\text{dom}} + m_{\text{int}}$ subproblems.

The further proceeding of AMLS in the multi-level version is analogous to the single-level version. As in (5.21) the row and column indices of the matrices K and M are reordered to achieve a matrix partitioning according to the performed domain substructuring. For example the matrix partitioning of K corresponding to the two-level domain substructuring applied in Figure 5.3 is

$$K = \begin{array}{c} \\ \Omega_1^{(2)} \\ \Omega_2^{(2)} \\ \Gamma_1^{(1)} \\ \Omega_3^{(2)} \\ \Omega_4^{(2)} \\ \Gamma_2^{(1)} \\ \Gamma_1^{(0)} \end{array} \begin{array}{c} \begin{array}{ccccccc} \Omega_1^{(2)} & \Omega_2^{(2)} & \Gamma_1^{(1)} & \Omega_3^{(2)} & \Omega_4^{(2)} & \Gamma_2^{(1)} & \Gamma_1^{(0)} \end{array} \\ \begin{bmatrix} K_{11} & & K_{13} & & & & K_{17} \\ & K_{22} & K_{23} & & & & K_{27} \\ K_{31} & K_{32} & K_{33} & & & & K_{37} \\ & & & K_{44} & & K_{46} & K_{47} \\ & & & & K_{55} & K_{56} & K_{57} \\ & & & K_{64} & K_{65} & K_{66} & K_{67} \\ K_{71} & K_{72} & K_{73} & K_{74} & K_{75} & K_{76} & K_{77} \end{bmatrix} \end{array} \tag{5.29}$$

where $K_{ij} \in \mathbb{R}^{N_i \times N_j}$ is the submatrix of K in block-row i and block-column j for $i, j = 1, \ldots, m$ and with $N = \sum_{i=1}^{m} N_i$. It is explicitly noted that the multi-level version of AMLS does not correspond to a recursive call of the single-level version. Instead the different matrix operations, done in the single-level version, are applied analogously to the matrices from the multi-level version, i.e., to matrices of the form (5.29) for example.

In the next step the eigenvalue problem (K, M) is transformed equivalently to $(\widetilde{K}, \widetilde{M})$, i.e., K is block diagonalised via $K = L\widetilde{K}L^T$ by performing a block $\mathrm{LDL^T}$-decomposition and M is transformed correspondingly by $\widetilde{M} = L^{-1}ML^{-T}$. Due to the transformation a part of the sparsity structure is lost in $\widetilde{K}$ and $\widetilde{M}$. All submatrices $\widetilde{K}_{ij}$ and $\widetilde{M}_{ij}$ are now typically dense if their respective row or column indices are associated with an interface. Furthermore, it holds $\widetilde{K}_{ii} = K_{ii}$ and $\widetilde{M}_{ii} = M_{ii}$ if the respective row and column indices are associated with a subdomain. For example, for the two-level substructuring described in Figure 5.3 the transformed matrices are of the form $\widetilde{K} = \mathrm{diag}\big[K_{11}, K_{22}, \widetilde{K}_{33}, K_{44}, K_{55}, \widetilde{K}_{66}, \widetilde{K}_{77}\big]$ and

$$\widetilde{M} = \begin{bmatrix} M_{11} & & \widetilde{M}_{13} & & & & \widetilde{M}_{17} \\ & M_{22} & \widetilde{M}_{23} & & & & \widetilde{M}_{27} \\ \widetilde{M}_{31} & \widetilde{M}_{32} & \widetilde{M}_{33} & & & & \widetilde{M}_{37} \\ & & & M_{44} & & \widetilde{M}_{46} & \widetilde{M}_{47} \\ & & & & M_{55} & \widetilde{M}_{56} & \widetilde{M}_{57} \\ & & & \widetilde{M}_{64} & \widetilde{M}_{65} & \widetilde{M}_{66} & \widetilde{M}_{67} \\ \widetilde{M}_{71} & \widetilde{M}_{72} & \widetilde{M}_{73} & \widetilde{M}_{74} & \widetilde{M}_{75} & \widetilde{M}_{76} & \widetilde{M}_{77} \end{bmatrix} \tag{5.30}$$

where the transformation matrix L is of the form

$$L = \begin{bmatrix} \mathrm{Id} & & & & & & \\ & \mathrm{Id} & & & & & \\ L_{31} & L_{32} & \mathrm{Id} & & & & \\ & & & \mathrm{Id} & & & \\ & & & & \mathrm{Id} & & \\ & & & L_{64} & L_{65} & \mathrm{Id} & \\ L_{71} & L_{72} & L_{73} & L_{74} & L_{75} & L_{76} & \mathrm{Id} \end{bmatrix}.$$

In the next step the partial eigensolutions of the subproblems $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ are computed. Let the partial eigensolution be given again by

$$\widetilde{K}_{ii}\,\widetilde{S}_i \;=\; \widetilde{M}_{ii}\,\widetilde{S}_i\,\widetilde{D}_i \qquad \text{with} \quad \widetilde{S}_i^T\,\widetilde{M}_{ii}\,\widetilde{S}_i \;=\; \mathrm{Id} \tag{5.31}$$

for $i = 1, \ldots, m$, where the diagonal matrix $\widetilde{D}_i \in \mathbb{R}^{k_i \times k_i}$ contains the $k_i \leq N_i$ smallest eigenvalues and $\widetilde{S}_i \in \mathbb{R}^{N_i \times k_i}$ column-wise the associated eigenvectors. In the next step the reduced matrices $\widehat{K} := Z^T \widetilde{K} Z \in \mathbb{R}^{\bar{k} \times \bar{k}}$ and $\widehat{M} := Z^T \widetilde{M} Z \in \mathbb{R}^{\bar{k} \times \bar{k}}$ are computed where $Z := \mathrm{diag}\big[\widetilde{S}_1, \ldots, \widetilde{S}_m\big]$ and $\bar{k} := \sum_{i=1}^m k_i$. For example, for the two-level substructuring described in Figure 5.3 the reduced matrices are of the form

$$\widehat{K} = \mathrm{diag}\big[\widetilde{D}_1, \ldots, \widetilde{D}_7\big] \quad \text{and} \quad \widehat{M} = \begin{bmatrix} \mathrm{Id} & & \widehat{M}_{13} & & & & \widehat{M}_{17} \\ & \mathrm{Id} & \widehat{M}_{23} & & & & \widehat{M}_{27} \\ \widehat{M}_{31} & \widehat{M}_{32} & \mathrm{Id} & & & & \widehat{M}_{37} \\ & & & \mathrm{Id} & & \widehat{M}_{46} & \widehat{M}_{47} \\ & & & & \mathrm{Id} & \widehat{M}_{56} & \widehat{M}_{57} \\ & & & \widehat{M}_{64} & \widehat{M}_{65} & \mathrm{Id} & \widehat{M}_{67} \\ \widehat{M}_{71} & \widehat{M}_{72} & \widehat{M}_{73} & \widehat{M}_{74} & \widehat{M}_{75} & \widehat{M}_{76} & \mathrm{Id} \end{bmatrix}.$$

Finally, the n_{es} smallest eigenpairs $(\widehat{\lambda}_j, \widehat{x}_j)$ of the reduced eigenvalue problem $(\widehat{K}, \widehat{M})$ are computed where eigenpair approximations of the original eigenvalue problem (K, M) are obtained by computing $\widehat{y}_j := L^{-T} Z\, \widehat{x}_j$. To summarise the AMLS method an overview of all necessary operations is given in Table 5.1 where the different tasks of the method are denoted by $(\mathbb{T}1)$–$(\mathbb{T}8)$.

Remark 5.9 (Purely Algebraic Setting) *The matrix partitioning in* (5.21) *can be obtained also in a purely algebraic way by applying graph partitioning algorithms like nested dissection to the graph of the matrix*[4] $|K|+|M|$*, and AMLS is applied to the pure algebraic eigenvalue problem* $Kx = \lambda Mx$ *where symmetric sparse matrices* K*,* M *are given and where* M *is additionally positive definite. For this reason, the AMLS method is often analysed in literature (e.g., in [28, 30, 67]) only in a pure algebraic setting.*

[4] For a given matrix $A = (a_{ij})_{i,j=1}^N \in \mathbb{R}^{N \times N}$ the matrix $|A| \in \mathbb{R}^{N \times N}$ is defined by $|A| = (\,|a_{ij}|\,)_{i,j=1}^N$.

Task	Matrix Operations AMLS
($\mathbb{T}1$) partition matrices K and M	apply nested dissection reordering as done, for example, in (5.21) and (5.29)
($\mathbb{T}2$) block diagonalise the matrix K	$K = L\,\widetilde{K}\,L^T$
($\mathbb{T}3$) transform M	$\widetilde{M} = L^{-1}\,M\,L^{-T}$
($\mathbb{T}4$) compute partial eigensolutions (for $i = 1, \ldots, m$)	$\widetilde{K}_{ii}\,\widetilde{S}_i = \widetilde{M}_{ii}\,\widetilde{S}_i\,\widetilde{D}_i$ with $\widetilde{S}_i \in \mathbb{R}^{N_i \times k_i}$ and $\widetilde{D}_i \in \mathbb{R}^{k_i \times k_i}$
($\mathbb{T}5$) define subspace	$Z := \operatorname{diag}\left[\widetilde{S}_1, \ldots, \widetilde{S}_m\right] \in \mathbb{R}^{N \times \bar{k}}$ with $\bar{k} = \sum_{i=1}^m k_i$
($\mathbb{T}6$) compute matrices of the reduced eigenvalue problem	$\widehat{K} := Z^T\,\widetilde{K}\,Z \in \mathbb{R}^{\bar{k}\times\bar{k}}$, $\widehat{M} := Z^T\,\widetilde{M}\,Z \in \mathbb{R}^{\bar{k}\times\bar{k}}$
($\mathbb{T}7$) solve reduced eigenvalue problem	$\widehat{K}\,\widehat{x}_j = \widehat{\lambda}_j\,\widehat{M}\,\widehat{x}_j$ for $j = 1, \ldots, n_{\text{es}}$
($\mathbb{T}8$) transform eigenvectors	$\widehat{y}_j := L^{-T} Z\,\widehat{x}_j$ for $j = 1, \ldots, n_{\text{es}}$ (to restore the original index-ordering for the approximated eigenvector the inverse of the in task ($\mathbb{T}1$) performed index permutation π has to be applied to $\widehat{y}_j$)

Table 5.1.: Overview of the AMLS method to compute eigenpair approximations $(\widehat{\lambda}_j, \widehat{y}_j)$ for the smallest n_{es} eigenpairs of (K, M).

The benefit of the multi-level approach is that the substructuring of the domain Ω or respectively the partitioning of the matrices K and M can be applied recursively until eventually in (5.31) the size of the subproblems $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ associated to subdomains is small enough to be solved easily. However, if more and more levels are used in the multi-level approach of AMLS, then the size of the reduced eigenvalue problem increases as $\bar{k} = \sum_{i=1}^m k_i$ grows with the number m of subproblems. Although the reduced problem is partially structured ($\widehat{K}$ is a diagonal matrix and the structure of $\widehat{M}$ is inherited from the block-sparsity of M), eventually the total complexity is dominated by this part. As a consequence, the number of levels has to be controlled so that at most $\mathcal{O}(n_{\text{es}})$ eigenvectors are used from all subproblems together. Altogether, to achieve both, i.e., subproblems that are easy to solve and a reduced eigenvalue problem of size $\mathcal{O}(n_{\text{es}})$, it is proposed to apply the AMLS method only with a few levels and to apply the method recursively to the large subdomain eigenvalue problems. Details of this approach are presented in the following section.

5.2.3. Recursive AMLS

In the following we denote a subproblem $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ that is associated to an interface simply as *interface eigenvalue problem* and a subproblem $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ associated to a subdomain as *subdomain eigenvalue problems.*

The recursive version of AMLS has been first discussed in [31] by the author (to the best of his

knowledge). In the recursive approach the classical (multi-level) AMLS method is applied (as described in Section 5.2.1 and Section 5.2.2) with only a few levels, and with the special feature that in task (𝕋4) large subdomain eigenvalue problems $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ are solved recursively by the AMLS method. Since in this approach the number of levels from the multi-level substructuring is kept small the number of subdomains (from the multi-level substructuring) can be bounded by $\mathcal{O}(1)$.

As already discussed only $j_{\max}^{\mathrm{EF}}(N)$ eigenfunctions (cf. Corollary 3.17) of the continuous problem (4.2) can be well approximated by the discretised problem (K, M) using a finite element space V_h with $N = \dim V_h$ degrees of freedom and where $j_{\max}^{\mathrm{EF}}(N) \leq n_{\mathrm{es}} \ll N$. Furthermore, each subdomain eigenvalue problem $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ is associated to a subdomain $\Omega_j^{(l)}$ (with suitable $l \in \mathbb{N}$ and $j \in \mathbb{N}$), in particular it holds $(\widetilde{K}_{ii}, \widetilde{M}_{ii}) = (K_{ii}, M_{ii})$, i.e., the subproblem corresponds to the finite element discretisation of the continuous eigenvalue problem on $\Omega_j^{(l)}$ (cf. Remark 5.6) using the finite element space V_h restricted[5] to $\Omega_j^{(l)}$ with $N_i < N$ degrees of freedom. Correspondingly, in each of the subdomains $\Omega_j^{(l)}$ there are $k_i < n_{\mathrm{es}}$ eigenfunctions that can be represented well in the associated finite element space. Since in the recursive approach the number of subdomains (from the multi-level substructuring) is in $\mathcal{O}(1)$ it follows[6] that the size of the reduced problem is at most $\mathcal{O}(n_{\mathrm{es}})$. If it holds, for example, that $n_{\mathrm{es}} \in \mathcal{O}(N^{1/3})$ then the reduced eigenvalue problem $(\widehat{K}, \widehat{M})$ can be handled by a standard dense linear algebra solver with cubic complexity whereby the computational costs for task (𝕋7) still remain in $\mathcal{O}(N)$.

When in task (𝕋4) the AMLS method is used for the solution of a subdomain problem $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ then an approximated eigensolution

$$\widetilde{K}_{ii}\,\widetilde{S}_i \;\approx\; \widetilde{M}_{ii}\,\widetilde{S}_i\,\widetilde{D}_i \qquad \text{with} \quad \widetilde{S}_i^T\,\widetilde{M}_{ii}\,\widetilde{S}_i \;=\; \mathrm{Id} \tag{5.32}$$

is obtained. Although it is primarily aimed to approximate the continuous eigenfunctions and not the discrete eigenvectors of $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$, it is reasonable to compute slightly more eigenvectors than in the case when, e.g., a direct solver is used for the solution of $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ [which computes almost numerically exact eigenvectors], so that a possibly lower approximation quality of AMLS can be compensated and enough spectral information of $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ is provided for the subspace in task (𝕋5). This issue is discussed in more detail in Section 8.7 and Section 9.2 where the implementation is described and numerical results are presented.

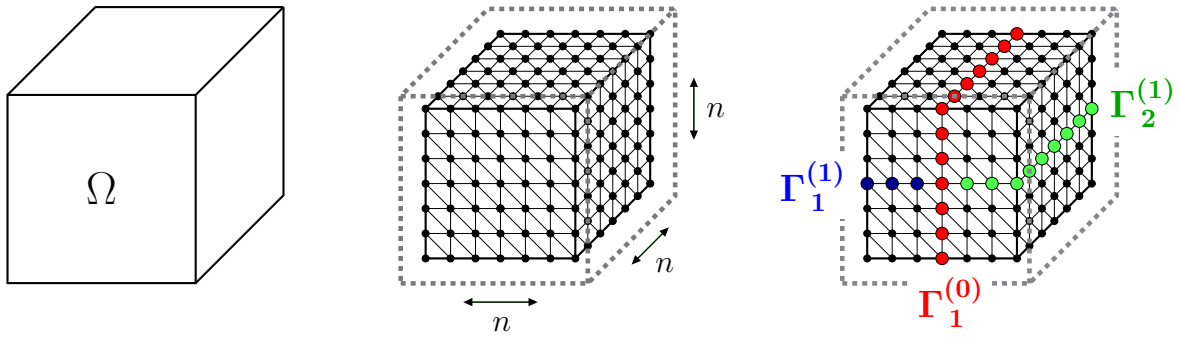
The recursive approach and the multi-level approach of AMLS affect the size of the subdomain eigenvalue problems, however, both approaches do not effect the size of subproblems that are related to interfaces. When the spatial domain Ω of the problem is three-dimensional this is a bottleneck, as it is shown in the following section.

5.3. Efficiency Problems in the Three-Dimensional Case

In the following we refer to a submatrix whose row or column indices are associated with an interface as an *interface matrix*. When AMLS is applied to a three-dimensional problem the interface matrices are getting relatively large which is leading to very high computational costs.

[5] More precisely, this is the finite element space which is spanned by the basis functions $(\varphi_j^{(h)})_{j=1}^N$ of V_h whose index j is associated with the rows and columns of $\widetilde{K}_{ii}$ or $\widetilde{M}_{ii}$.

[6] Provided that for interface problems the number of selected eigenvectors k_i is bounded as well by $k_i < n_{\mathrm{es}}$.



(a) Domain $\Omega = (0,1)^3$.

(b) Triangulation of Ω with n^3 DOF; Only the grid associated to the DOF is marked where DOF are indicated by small dots.

(c) Two-level substructuring of the discretised domain leading to interfaces of the size $\mathcal{O}(n^2)$.

Figure 5.4.: Two-level substructuring of the discretised domain $\Omega = (0,1)^3$.

As discussed in the previous section further substructuring reduces only the size of submatrices which are associated only with subdomains, but the substructuring reduces not the size of the interface matrices.

To illustrate this bottleneck we consider the initial eigenvalue problem (4.2) with the underlying domain $\Omega = (0,1)^3$. To solve this problem with the AMLS method it has to be discretised first using finite elements. A triangulation of $\Omega = (0,1)^3$ can be obtained, for example, by decomposing $\Omega = (0,1)^3$ into $n+1$ equispaced subintervals in each direction and using the finite element space of piecewise affine functions $\mathbb{X}^1_{h,0}$ with mesh width $h = 1/(n+1)$ [see Figure 5.4(b) for illustration]. The discretisation results in the algebraic eigenvalue problem (5.20) where the matrices K and M are of size $N \times N$ with $N = n^3$. If we assume that, for example, a two-level substructuring is performed in AMLS then we obtain a matrix partitioning of the form (5.29) where the number of rows or columns of the interface matrices are $\mathcal{O}(N^{2/3})$ [see Figure 5.4(c) for illustration]. The size of these interface matrices is relatively large and cannot be reduced by further substructuring.

During the procedure of AMLS a couple of matrix operations have to be performed on the interface matrices, e.g, computing the inverse, the matrix product or the partial eigensolution. The interface matrices are not only relatively large, they become dense as well when task ($\mathbb{T}2$) and ($\mathbb{T}3$) is performed. For example in the two-level version of AMLS the inverse of the interface matrices $\widetilde{K}_{33}$ and $\widetilde{K}_{66}$ has to be computed when in task ($\mathbb{T}2$) the block $\mathrm{LDL}^{\mathrm{T}}$-decomposition $K = L\widetilde{K}L^T$ is performed. These operations alone lead to costs of $\mathcal{O}\big((N^{2/3})^3\big) = \mathcal{O}\big(N^2\big)$, and hence AMLS is getting too expensive for three-dimensional problems. The so-called hierarchical matrices which are introduced in the next chapter are a possibility to resolve this problem.

6. Hierarchical Matrices

The so-called hierarchical matrices (short $\mathcal{H}$-matrices) [38, 39] are data-sparse but possibly dense matrices. The basic idea is to reorder the rows and columns of a matrix such that certain submatrices can be represented or approximated by low rank matrices. Using the low rank approximation, a fully populated but data-sparse matrix of size $N \times N$ can be represented using only $\mathcal{O}(N \log^\alpha N)$ data instead of storing N^2 entries where $\alpha = 1, \ldots, 4$ (cf. [33, 35]). Most importantly, $\mathcal{H}$-matrices provide exact matrix-vector multiplication and approximated matrix(-matrix) operations (e.g. multiplication, addition, inversion, LU-factorisation) which are performed in almost linear complexity $O(N \log^\alpha N)$.

As described in Section 3.3, the stiffness matrix resulting from the finite element discretisation of an elliptic PDE operator is sparse. However, the inverse and the LU-factors of the stiffness matrix are in general fully populated. In [13, 29] and [12, 29, 34, 59] it is shown that the inverse and the LU-factors can be well approximated by $\mathcal{H}$-matrices and that these approximations can be computed with almost linear complexity. This motivates to use the fast $\mathcal{H}$-matrix arithmetic in task ($\mathbb{T}2$) and ($\mathbb{T}2$) of the AMLS method for the computation of the block diagonalisation $K = L\widetilde{K}L^T$ and the matrix transformation $\widetilde{M} = L^{-1}ML^{-T}$.

To use the fast $\mathcal{H}$-arithmetic the sparse matrices K and M have to be converted into $\mathcal{H}$-matrices. For this purpose a suitable $\mathcal{H}$-matrix format has to be provided which is based on the geometry information of the partial differential equation (4.1). To introduce this $\mathcal{H}$-matrix format and the basics of $\mathcal{H}$-matrices it is first explained how the inverse of a stiffness matrix is approximated by an $\mathcal{H}$-matrix.

A description of the $\mathcal{H}$-matrices by the author has been already given in [31, Section 5]. This chapter is a strongly revised version of [31, Section 5] and provides a more detailed discussion of the topic.

6.1. $\mathcal{H}$-Matrix Approximation of the Inverse Stiffness Matrix

Assume $G \in \mathbb{R}^{N\times N}$ is the stiffness matrix[1] resulting from the finite element discretisation of an elliptic partial differential operator. The matrix G is sparse, however, its inverse G^{-1} is fully populated. Recalling the definition of the stiffness matrix in (4.5) each row and column index $i \in I := \{1, \ldots, N\}$ of G and respectively of G^{-1} is associated with a basis function $\varphi_i^{(h)}$ of the underlying finite element space V_h with $N = \dim V_h$ degrees of freedom. To emphasise that the row and column indices of G and G^{-1} are associated with the index set I the notation $G, G^{-1} \in \mathbb{R}^{I\times I}$ is used. The support of each index set $t \subset I$ is defined by

$$\Omega_t := \bigcup_{i\in t} \operatorname{supp}\big(\varphi_i^{(h)}\big). \tag{6.1}$$

[1] To avoid misunderstandings, the stiffness matrix is denoted in this particular section by G and not by K since the $\mathcal{H}$-matrix structure used to approximate the inverse G^{-1} differs from the $\mathcal{H}$-matrix structure which will be used later in the AMLS method in order to approximate the factorisation $K = L\widetilde{K}L^T$.

Correspondingly each submatrix

$$G^{-1}{}_{|s\times t} := \big((G^{-1})_{ij} \big)_{i\in s, j\in t} \qquad \text{with } s,t \subset I$$

of G^{-1} is associated with the geometry information of Ω_s and Ω_t. Based on the geometric separation of the supports Ω_s and Ω_t certain subblocks $s \times t \subset I \times I$ can be identified that allow a low rank approximation of the respective submatrices $G^{-1}{}_{|s\times t}$. More precisely, submatrices $G^{-1}{}_{|s\times t}$ whose index sets s and t fulfil the so-called *admissibility condition*

$$\min\big\{ \operatorname{diam}(\Omega_s), \operatorname{diam}(\Omega_t) \big\} \;\le\; \eta \operatorname{dist}(\Omega_s, \Omega_t) \tag{6.2}$$

are well suited for a low rank approximation (cf. [13]). The quantities

$$\operatorname{diam}(\Omega_t) := \max\big\{ \|x-y\|_2 \;:\; x,y \in \Omega_t \big\}, \tag{6.3}$$

$$\operatorname{dist}(\Omega_s, \Omega_t) := \min\big\{ \|x-y\|_2 \;:\; x \in \Omega_s, y \in \Omega_t \big\} \tag{6.4}$$

are the diameter and the distance of the supports of s and t, and the parameter $\eta > 0$ controls the number of admissible subblocks $s \times t$ and is typically set to $\eta = 1$ (see, e.g., [33]). Subblocks $s \times t$ fulfilling the admissibility condition (6.2) are called *admissible* and submatrices associated to these subblocks are approximated by so-called R(k)-matrices which are defined as follows.

Definition 6.1 (R(k)-Matrix Representation) *Let $k, m, n \in \mathbb{N}_0$ and $R \in \mathbb{R}^{n\times m}$ be a matrix of rank at most k. If the matrix R is stored in factorised form*

$$R = AB^T \qquad \text{with } A \in \mathbb{R}^{n\times k} \text{ and } B \in \mathbb{R}^{m\times k} \tag{6.5}$$

and when A, B are stored in full matrix representation[2] then R is called an R(k)*-matrix. Furthermore,* (6.5) *is called the* R(k)*-matrix representation of R.*

When the rank k is small compared to n and m the representation of an R(k)-matrix of size $n \times m$ is much cheaper than in full matrix representation because only $k(n+m)$ entries have to be stored instead of nm. Furthermore, when k is small basic matrix operations can be evaluated much more efficiently in R(k)-matrix representation than in full matrix representation. More precisely, for an R(k)-matrix $R = AB^T \in \mathbb{R}^{n\times m}$ it holds:

- Matrix-Vector Multiplication: The matrix-vector product $y := Rx$ for $x \in \mathbb{R}^m$ can be computed in only $\mathcal{O}(k(n+m))$ by first computing $z := B^T x \in \mathbb{R}^k$ and afterwards $y = Az$.
- Matrix Addition: Let $\tilde{R} = \tilde{A}\tilde{B}^T$ be an R(k)-matrix of size $n \times m$ then the sum

$$R + \tilde{R} \;=\; AB^T + \tilde{A}\tilde{B}^T \;=\; \underbrace{[A\,\tilde{A}]}_{n\times 2k}\,\underbrace{[B\,\tilde{B}]^T}_{2k\times n} \tag{6.6}$$

  is an R($2k$)-matrix and no computation is necessary.

[2] A matrix $B \in \mathbb{R}^{m\times k}$ is said to be stored in *full matrix representation* if the matrix entries are stored (column-wise) in an array of real numbers of length mk.

- Matrix Multiplication: Let $R_1 = A_1 B_1^T \in \mathbb{R}^{l\times n}$ and $R_2 = A_2 B_2^T \in \mathbb{R}^{m\times l}$ be R(k)-matrices. Then the matrix products

$$R_1 R = \tilde{A} B^T \quad \text{with} \quad \tilde{A} := A_1 (B_1^T A),$$
$$R R_2 = A \tilde{B}^T \quad \text{with} \quad \tilde{B} := B_2 (A_2^T B)$$

  can be represented in the R(k)-matrix format where $\tilde{A} \in \mathbb{R}^{l\times k}$ and $\tilde{B} \in \mathbb{R}^{l\times k}$ can be computed in $\mathcal{O}(k^2(l+n))$ and $\mathcal{O}(k^2(l+m))$. Also, when R_1 and R_2 are in full matrix representation the products $R_1 R$ and $R R_2$ can be represented as R(k)-matrices with $R_1 R = \tilde{A} B^T$ and $R R_2 = A \tilde{B}^T$ where $\tilde{A} := R_1 A$ and $\tilde{B} := R_2^T B$ can be computed in $\mathcal{O}(lnk)$ and $\mathcal{O}(lmk)$.

Algorithm 1 Geometric Bisection of an Index Set

```
procedure GeometricBisection( t, (ξ_i)_{i∈t})
    for j = 1,...,d do                          ▷ determine a bounding box containing all ξ_i
        a_j := min{ ⟨e_j, ξ_i⟩ : i ∈ t };        ▷ ⟨·,·⟩ denotes the dot product in R^d
        b_j := max{ ⟨e_j, ξ_i⟩ : i ∈ t };
    end for
    j_max := argmax{b_j − a_j : j = 1,...,d};   ▷ j_max is the direction of maximal extent
    c := (a_{j_max} + b_{j_max})/2;             ▷ split the bounding box in the chosen direction
    t_1 := ∅, t_2 := ∅;
    for all i ∈ t do                            ▷ distribute the indices i ∈ t to the sons t_1 and t_2
        if ⟨e_{j_max}, ξ_i⟩ ≤ c then
            t_1 := t_1 ∪ {i};
        else
            t_2 := t_2 ∪ {i};
        end if
    end for
    return cluster t with S(t) := {t_1, t_2}
end procedure
```

In order to exploit the low rank approximation property of submatrices $G^{-1}{}_{|s\times t}$ fulfilling (6.2) the row and column indices of G^{-1} have to be reordered according to a suitable partitioning of the product index set $I \times I$. How a suitable partitioning of $I \times I$ is constructed is described in the following.

At first the index set I is divided according to a geometric bisection of its support into two disjoint index sets $s, t \subset I$ with $I = s \dot{\cup} t$. This geometric bisection is applied recursively to the index sets s and t until the cardinality of an index set falls below some given limit $n_{\min} \in \mathbb{N}$. Using this approach a hierarchy of disjoint partitions of the index set I is obtained where the corresponding subsets of the partitioning tend to be geometrically separated. In Figure 6.1(a) and 6.1(b) such a partitioning of I is illustrated for a two-dimensional problem. The partitioning of I can be obtained, for example, by applying Algorithm 1 recursively to the index set I (cf. [18]), where for each index $i \in I$ a geometric representative

$$\xi_i \in \operatorname{supp}\big(\varphi_i^{(h)}\big) \subset \mathbb{R}^d \tag{6.7}$$

is chosen which can be, e.g., the nodal point of the basis function $\varphi_i^{(h)}$ or the geometric centre of the corresponding support. Let $\{e_1, ..., e_d\}$ be an arbitrary orthonormal basis in $\mathbb{R}^d$ (e.g.,

the standard basis) then Algorithm 1 will split a given index set $t \subset I$ into disjoint index sets $t_1, t_2 \subset t$ such that the associated geometric representatives ξ_i are separated by a hyper-plane. The described geometric partitioning of I is organised in a so-called *cluster tree*:

Definition 6.2 (Cluster Tree) *Let $\mathcal{T}_I = (V, E)$ be a tree with vertex set V and edge set $E \subset V \times V$. For a vertex $v \in V$ the set of its sons is defined by $S(v) := \{w \in V : (v, w) \in E\}$. The tree $\mathcal{T}_I$ is called a cluster tree over the index set I if the following conditions are fulfilled:*

i) For all $v \in V$ it holds $v \subset I$ and $v \neq \emptyset$.

ii) $I \in V$ is the root of $\mathcal{T}_I$.

iii) For all $v \in V$ it holds $S(v) = \emptyset$ or $v = \dot{\bigcup}_{w \in S(v)} w$.

The nodes $v \in V$ are called clusters. For reasons of simplification, we identify V and $\mathcal{T}_I$, i.e., we just write $v \in \mathcal{T}_I$ instead of $v \in V$. Furthermore, we call a cluster without sons a leaf and define the set of leaves of $\mathcal{T}_I$ by $\mathcal{L}(\mathcal{T}_I) := \{v \in \mathcal{T}_I : S(v) = \emptyset\}$.

Algorithm 2 Construction of the Block Cluster Tree

```
procedure ConstructBlockClusterTree( s × t, n_min, η )
    if s × t is admissible then
        S(s × t) := ∅;
    else if min{#s, #t} ≤ n_min then
        S(s × t) := ∅;                    ▷ n_min affects the minimal size of the block clusters
    else
        S(s × t) := {s' × t' : s' ∈ S(s), t' ∈ S(t)};
        for all s' × t' ∈ S(s × t) do
            ConstructBlockClusterTree( s' × t', n_min, η );
        end for
    end if
    return s × t;
end procedure
```

To obtain a partitioning of the product index set $I \times I$ the so-called *block cluster tree* $\mathcal{T}_{I \times I}$ is introduced. The block cluster tree corresponding to the cluster tree T_I and admissible condition (6.2) is obtained by applying Algorithm 2 to the product index set $I \times I$. Using this algorithm, $I \times I$ is recursively subdivided into subblocks $s \times t$ until the subblock gets admissible (which is controlled by the parameter η) or the size of the subblock falls below the limit $n_{\min}$ as it is illustrated in Figure 6.1(c). The block cluster tree $\mathcal{T}_{I \times I}$ itself is a cluster tree over the product index set $I \times I$ (cf. Definition 6.2) and provides a hierarchy of disjoint block partitions of the product index set $I \times I$. In particular, it holds

$$I \times I = \dot{\bigcup}_{s \times t \in \mathcal{L}(\mathcal{T}_{I \times I})} s \times t \tag{6.8}$$

where $\mathcal{L}(\mathcal{T}_{I \times I})$ is the set of leaves of $\mathcal{T}_{I \times I}$. Using this block partitioning of $I \times I$ a hierarchical matrix format can be defined which is well suited for the approximation of G^{-1}:

Definition 6.3 ($\mathcal{H}$-matrix Representation) *Let $k \in \mathbb{N}_0$. The set of $\mathcal{H}$-matrices induced by the block cluster tree $\mathcal{T}_{I\times I}$ with block-wise rank k is defined by*

$$\mathcal{H}(\mathcal{T}_{I\times I}, k) := \Big\{ A \in \mathbb{R}^{I\times I} \ : \ \operatorname{rank}(A_{|s\times t}) \leq k \quad \text{for all admissible leaves } s\times t \text{ of } \mathcal{T}_{I\times I} \Big\}.$$

We say a matrix $A \in \mathcal{H}(\mathcal{T}_{I\times I}, k)$ is stored in $\mathcal{H}$-matrix representation if the submatrices $A_{|s\times t}$ associated to admissible leaves $s\times t$ of $\mathcal{T}_{I\times I}$ are stored as $\mathrm{R}(k)$-matrices whereas submatrices associated to inadmissible leaves are stored as full matrices.

$\mathcal{H}$-matrices typically consist of large low rank matrices and small full matrices. The size of the full matrices is controlled by the parameter $n_{\min}$ which has been used for the construction of the trees $\mathcal{T}_I$ and $\mathcal{T}_{I\times I}$. For inadmissible leaves $s\times t$ of $\mathcal{T}_{I\times I}$ it holds $\min\{\#s, \#t\} \leq n_{\min}$. Standard values for $n_{\min}$ are in the range of 20 to 40 (cf. [50]).

Finally, when the rows and columns of G^{-1} are reordered according to the applied partitioning of I, and G^{-1} is partitioned into blocks according to $\mathcal{L}(\mathcal{T}_{I\times I})$ [as illustrated in Figure 6.1(c)], then the reordered matrix $G^{-1} \in \mathbb{R}^{N\times N}$, which is in general fully populated, can be well approximated in the matrix format $\mathcal{H}(\mathcal{T}_{I\times I}, k)$ and only $\mathcal{O}(N \log^\alpha N)$ data is necessary instead of storing N^2 entries (cf. [13, 29]). Even more important: The $\mathcal{H}$-matrix approximation of G^{-1} can be computed in $\mathcal{O}(N \log^\alpha N)$ by an algorithm requiring only the matrix G and the used $\mathcal{H}$-matrix format. How this $\mathcal{H}$-matrix approximation is computed is briefly described in the following.

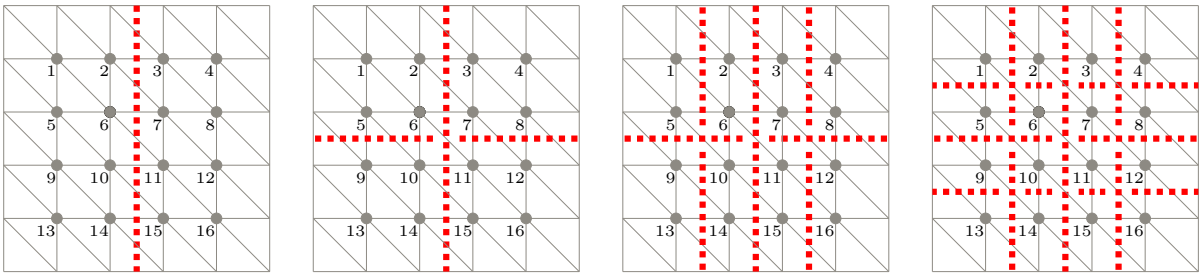
First of all, the rows and columns of the stiffness matrix $G \in \mathbb{R}^{I\times I}$ are reordered according to the partitioning of I, and the matrix is partitioned into blocks according to $\mathcal{L}(\mathcal{T}_{I\times I})$. Since for a block cluster $s\times t \in \mathcal{L}(\mathcal{T}_{I\times I})$ fulfilling admissibility condition (6.2) the supports of the basis functions $\varphi_i^{(h)}$ with $i \in s$ are geometrically separated from these of $\varphi_j^{(h)}$ with $j \in t$ it follows from Lemma 3.6 that the respective submatrix $G_{|s\times t}$ has only entries equal to zero and, therefore, $G_{|s\times t}$ can be represented exactly by an $\mathrm{R}(k)$-matrix with rank zero. Correspondingly, no approximation is necessary to represent the reordered matrix G in the matrix format $\mathcal{H}(\mathcal{T}_{I\times I}, k)$. The (exact) $\mathcal{H}$-matrix representation of G will be denoted in the following by $G^{\mathcal{H}}$.

$\mathcal{H}$-matrices are implemented in a structured way which is guided by the block cluster tree (cf. [39]) in order to ease the implementation of a corresponding $\mathcal{H}$-matrix arithmetic: For the matrix $G^{\mathcal{H}} = G^{\mathcal{H}}{}_{|I\times I}$ and a block cluster $s\times t \in \mathcal{T}_{I\times I}$ it holds

- if $s\times t \in \mathcal{L}(\mathcal{T}_{I\times I})$ then $G^{\mathcal{H}}{}_{|s\times t}$ is represented as a full matrix or an $\mathrm{R}(k)$-matrix,
- if $s\times t \notin \mathcal{L}(\mathcal{T}_{I\times I})$ then $G^{\mathcal{H}}{}_{|s\times t}$ is decomposed into a block matrix consisting of submatrices that are associated to the sons of $s\times t$, i.e., if $S(s) = \{s_1, \ldots, s_r\}$ and $S(t) = \{t_1, \ldots, t_c\}$ with $r, c \in \mathbb{N}$ it holds[3]

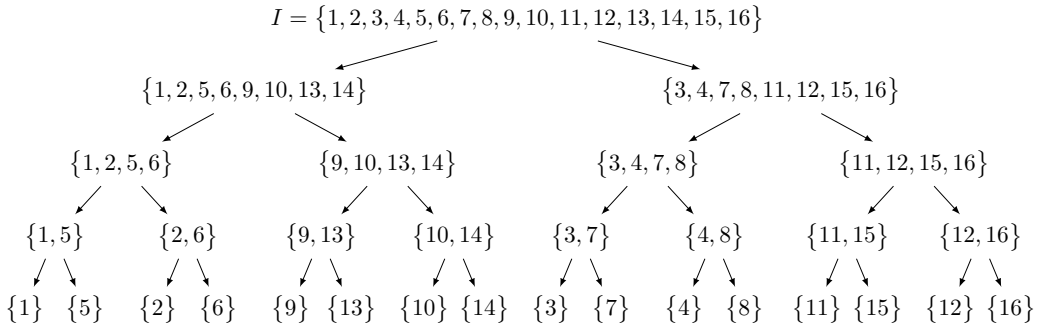
$$G^{\mathcal{H}}{}_{|s\times t} = \begin{bmatrix} G^{\mathcal{H}}{}_{|s_1\times t_1} & \cdots & G^{\mathcal{H}}{}_{|s_1\times t_c} \\ \vdots & \ddots & \vdots \\ G^{\mathcal{H}}{}_{|s_r\times t_1} & \cdots & G^{\mathcal{H}}{}_{|s_r\times t_c} \end{bmatrix}. \tag{6.9}$$

[3]Using a geometric bisection for the construction of $\mathcal{T}_I$ and Algorithm 2 for the construction of $\mathcal{T}_{I\times I}$ we obtain in (6.9) a 2×2 block structure.

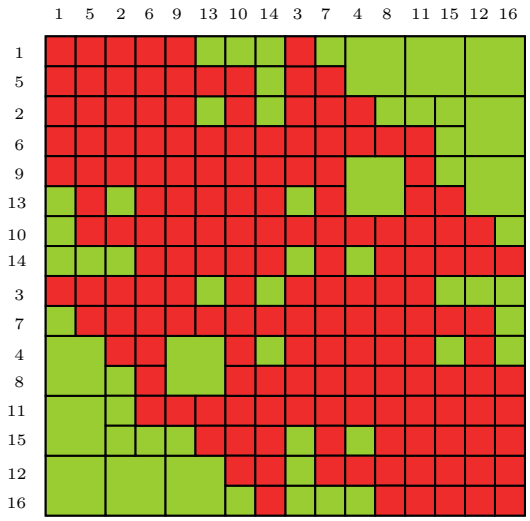


one-level bisection | two-level bisection | three-level bisection | four-level bisection

(a) Geometric bisection of the domain $\Omega = (0,1)^2$ using $n_{\min} = 1$. The indices $i \in I = \{1, \dots, 16\}$ of the nodal points of the basis functions are enumerated from 1 in the upper left to 16 in the lower right corner.



(b) Disjoint partitioning of the index set I corresponding to the applied geometric bisection.



(c) $\mathcal{H}$-Matrix format for $G^{-1} \in \mathbb{R}^{16\times 16}$ according to the applied partitioning of I using admissibility condition (6.2) with $\eta = 50$ and where $n_{\min} = 1$; admissible blocks are coloured green, inadmissible ones are red.

Figure 6.1.: Construction of the $\mathcal{H}$-matrix format for the inverse of the stiffness matrix resulting from a finite element discretisation of an elliptic partial differential operator on $\Omega = (0,1)^2$, where the finite element space of piecewise affine functions $\mathbb{X}^1_{h,0}$ defined on an equispaced grid with 16 DOF is used for the discretisation.

The computation of the $\mathcal{H}$-matrix approximation of G^{-1} is based on the following approach: The inverse of a regular matrix A can be computed by using the identify

$$A^{-1} = \begin{bmatrix} A_{11}^{-1} + A_{11}^{-1}A_{12}S^{-1}A_{21}A_{11}^{-1} & -A_{11}^{-1}A_{12}S^{-1} \\ -S^{-1}A_{21}A_{11}^{-1} & S^{-1} \end{bmatrix} \quad \text{with} \quad A := \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \tag{6.10}$$

and *Schur complement* $S := A_{22} - A_{21}A_{11}^{-1}A_{12}$. After the computation of A_{11}^{-1} and S^{-1} only multiplications and additions of submatrices have to be performed in order to obtain A^{-1}.

Applying identity (6.10) recursively to the block structure (6.9) of $G^{\mathcal{H}}$ its inverse can be computed. In particular, the computation can be performed very efficiently when the R(k)-matrix representation is exploited for the multiplication and addition of submatrices fulfilling condition (6.2). However, since the sum of two R(k)-matrices leads to an R(k)-matrix of rank $2k$ [see (6.6)] the problem arises that during the computation of the inverse the rank of submatrices can become unacceptably large. Correspondingly, submatrices have to be replaced during the computation by approximations with a reduced rank. Beside the possibility to reduce to a fixed rank k (i.e., the matrix format $\mathcal{H}(\mathcal{T}_{I\times I}, k)$ is retained) also an adaptive rank can be used in order to control the approximation quality of the R(k)-matrix approximations: For a desired approximation accuracy $\varepsilon > 0$ an arbitrary matrix M can be approximated by an R(k)-matrix R such that

$$\frac{\| M - R \|_2}{\| M \|_2} \leq \varepsilon \tag{6.11}$$

where the rank $k \in \mathbb{N}_0$ is as small as possible (cf. [33]). There are many different approaches for the efficient computation of such a low rank approximation (see, e.g., [39]). For example, a low rank approximation with accuracy ε of the sum (6.6) of the two R(k)-matrices can be computed in $\mathcal{O}(k^2(n+m))$ by the so-called *truncated singular value decomposition* (see [33]).

The corresponding $\mathcal{H}$-matrix format using an adaptive rank with accuracy ε for the R(k)-matrix approximation is denoted by $\mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$. Furthermore, let $(G^{\mathcal{H}})^{-1} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$ denote the $\mathcal{H}$-matrix approximation of G^{-1} which is computed by the recursive approach outlined above using an adaptive rank with accuracy ε. Then this approximated inverse is computed in $\mathcal{O}(N \log^{\alpha} N)$, see [33], and the error[4] $\|G^{-1} - (G^{\mathcal{H}})^{-1}\|_2$ is controlled by the chosen accuracy ε in (6.11).

6.2. $\mathcal{H}$-Matrix Format for AMLS

To use the fast $\mathcal{H}$-matrix arithmetic in task ($\mathbb{T}$2) and ($\mathbb{T}$3) of the AMLS method for the computation of the block diagonalisation $K = L\widetilde{K}L^T$ and the matrix transformation $\widetilde{M} = L^{-1}ML^{-T}$ the described $\mathcal{H}$-matrix format has to be changed slightly. First of all, a nested dissection is applied as in the classical AMLS method, i.e., the domain Ω is recursively subdivided into several subdomains which are separated by interfaces. The row and column indices of K and M are reordered according to the performed substructuring of Ω and a matrix partitioning, e.g., of the form (5.21) or (5.29) is obtained. As discussed in Section 5.2 some of the submatrices $\widetilde{K}_{ij}$ and $\widetilde{M}_{ij}$ are fully populated, however, they can be approximated in the $\mathcal{H}$-matrix format.

[4] To compare $(G^{\mathcal{H}})^{-1}$ with G^{-1} and to compute the error $\|G^{-1} - (G^{\mathcal{H}})^{-1}\|_2$, both matrices need to have the same ordering of the row and column indices.

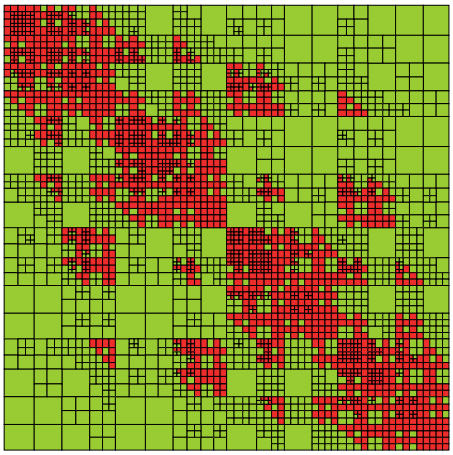


Figure 6.2: $\mathcal{H}$-matrix format for $G^{-1} \in \mathbb{R}^{N\times N}$ with $N = 2500$ using admissibility condition (6.2) with $\eta = 1$ and $n_{\min} = 40$ where G is the stiffness matrix resulting from a finite element discretisation of an elliptic partial differential operator on $\Omega = (0, 1)^2$.

For this purpose the index sets associated to the subdomains and interfaces are additionally partitioned according to a recursive geometric bisection, and the row and column indices of the submatrices $\widetilde{K}_{ij}$ and $\widetilde{M}_{ij}$ are reordered correspondingly. The described domain substructuring has been illustrated in Figure 6.3.

To compute the transformed eigenvalue problem by the fast $\mathcal{H}$-matrix arithmetic the reordered matrices K and M have to be represented in the matrix format $\mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$ where $\mathcal{T}_{I\times I}$ is the block cluster tree corresponding to the domain substructuring described above. Examples for the matrix format $\mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$ are given in Figure 6.4. The resulting $\mathcal{H}$-matrix representations of K and M will be denoted in the following by $K^{\mathcal{H}}$ and $M^{\mathcal{H}}$ (note that the representations are exact, cf. Section 6.1). Using these $\mathcal{H}$-matrix representations the block diagonalisation of K and the transformation of M can be computed by efficient algorithms (see, e.g., [34, 35, 39]), similar to the recursive algorithm used for the computation of $(G^{\mathcal{H}})^{-1}$, in $\mathcal{O}(N \log^{\alpha} N)$ leading to

$$K^{\mathcal{H}} \;\approx\; L^{\mathcal{H}}\, \widetilde{K}^{\mathcal{H}}\, (L^{\mathcal{H}})^T \qquad \text{and} \qquad \widetilde{M}^{\mathcal{H}} \;\approx\; (L^{\mathcal{H}})^{-1}\, M^{\mathcal{H}}\, (L^{\mathcal{H}})^{-T} \tag{6.12}$$

where $\widetilde{K}^{\mathcal{H}}, \widetilde{M}^{\mathcal{H}}, L^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$. The computation in (6.12) is performed not exactly but only approximatively, in particular the approximation errors $\|L - L^{\mathcal{H}}\|_2$, $\|\widetilde{K} - \widetilde{K}^{\mathcal{H}}\|_2$ and $\|\widetilde{M} - \widetilde{M}^{\mathcal{H}}\|_2$ are influenced by the chosen accuracy ε in (6.11). In Section 8 the computation in (6.12) is discussed in more detail, especially an improved $\mathcal{H}$-matrix format is introduced which is much more efficient for the computation of (6.12).

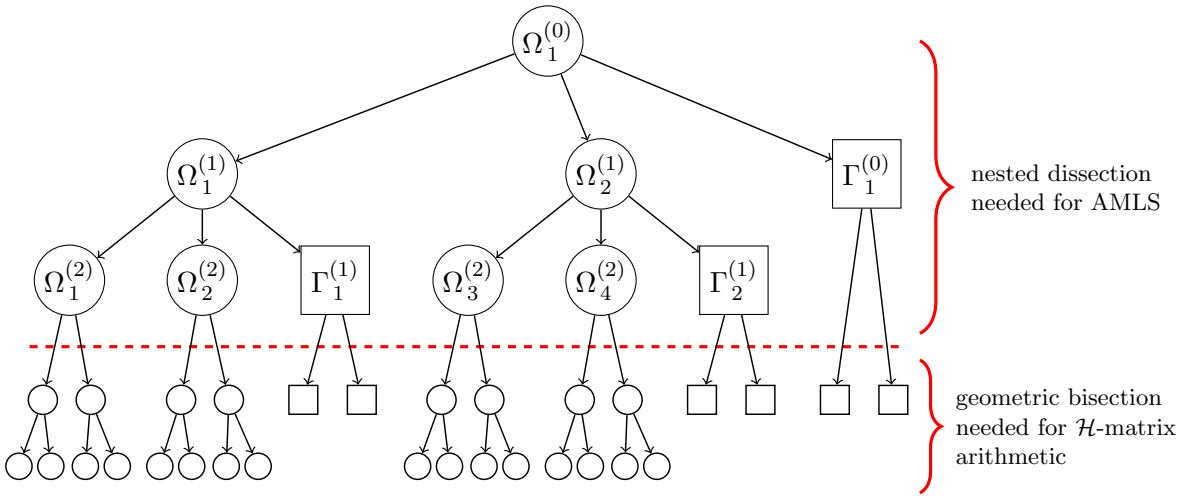
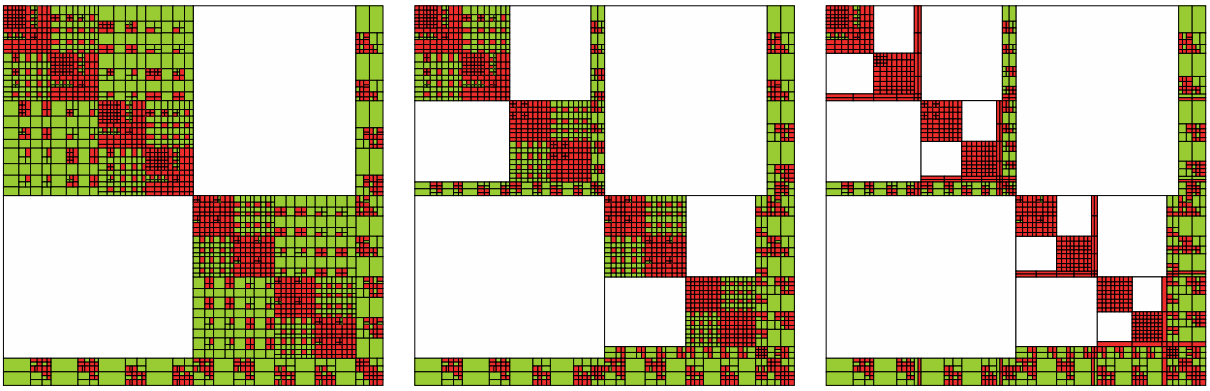


Figure 6.3.: Schematic example of the in $\mathcal{H}$-AMLS applied domain substructuring: A two-level nested dissection (necessary for AMLS, cf. Figure 5.3) is performed followed by an additional two-level geometric bisection of the subdomains and a one-level geometric bisection of the interfaces (necessary for the approximative $\mathcal{H}$-matrix arithmetic).



(a) one-level nested dis. **(b)** two-level nested dis. **(c)** three-level nested dis.

Figure 6.4.: $\mathcal{H}$-matrix format $\mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$ used in $\mathcal{H}$-AMLS for the finite element discretisation of an elliptic PDE eigenvalue problem on $\Omega = (0,1)^3$ with $\#I = 2744$ degrees of freedom. The matrix format $\mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$ is based on a one, two and three-level nested dissection with a subsequent geometric bisection. Furthermore, $\eta = 50$ and $n_{\min} = 40$ have been used for the construction of $\mathcal{T}_{I\times I}$. Red blocks represent full matrices, green blocks $\mathrm{R}(k)$-matrices and white blocks submatrices equal to zero which retain zero during the computation of (6.12) and don't cause computational costs.

7. Combination of AMLS and $\mathcal{H}$-Matrices

In this chapter a more refined version of the AMLS method is presented which is using the fast $\mathcal{H}$-matrix arithmetic. The benefit of the use of $\mathcal{H}$-matrices is a substantial reduction in computational time and storage requirements. However, due to the use of the approximative $\mathcal{H}$-matrix arithmetic, an additional error occurs which can influence the quality of the computed eigenpair approximations.

This chapter is organised as follows: In Section 7.1 the new method, called $\mathcal{H}$-AMLS, is introduced. The computational costs of the method are analysed in Section 7.2 and the accuracy of the computed eigenpair approximations is discussed in Section 7.3. Finally, in Section 7.4 an additional task is introduced which is basically one (approximative) iteration step of the subspace iteration and which further improves the accuracy of the $\mathcal{H}$-AMLS method.

The $\mathcal{H}$-AMLS method has been introduced in [31, Section 6] by the author. This chapter is a revised version of [31, Section 6]. It contains a more detailed description of the method, and it introduces the new *improvement task* from Section 7.4.

7.1. Introduction of the $\mathcal{H}$-AMLS method

As already described in Section 6.2, in the first step of the $\mathcal{H}$-AMLS method the domain Ω is substructured according to a nested dissection just like in the classical AMLS method, which results in a substructuring of Ω with $m \in \mathbb{N}$ subdomains and interfaces. To use the fast $\mathcal{H}$-matrix arithmetic additionally a geometric bisection of the subdomains and interfaces is performed. After the row and column indices of K and M are reordered corresponding to the performed domain substructuring, the exact $\mathcal{H}$-matrix representations $K^{\mathcal{H}}$ and $M^{\mathcal{H}}$ are constructed (cf. Section 6.2). In the next step the block diagonalisation of $K^{\mathcal{H}}$ is computed and the corresponding matrix transformation of $M^{\mathcal{H}}$, see (6.12), using the fast $\mathcal{H}$-matrix arithmetic.

The further proceeding of $\mathcal{H}$-AMLS is analogous to the classical AMLS method. At first the $m \times m$ block partitioning of the matrices $\widetilde{K}^{\mathcal{H}}$ and $\widetilde{M}^{\mathcal{H}}$ is introduced corresponding to the performed nested dissection [e.g., as in (5.21) or (5.29)], where $\widetilde{K}_{ij}^{\mathcal{H}} \in \mathbb{R}^{N_i \times N_j}$ and $\widetilde{M}_{ij}^{\mathcal{H}} \in \mathbb{R}^{N_i \times N_j}$ denote the corresponding submatrices in block row i and block column j. In the next step of $\mathcal{H}$-AMLS the partial eigensolutions of the subproblems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ are computed for $i = 1, \ldots, m$ which are given by

$$\widetilde{K}_{ii}^{\mathcal{H}} \, \widetilde{\mathbf{S}}_i \;=\; \widetilde{M}_{ii}^{\mathcal{H}} \, \widetilde{\mathbf{S}}_i \, \widetilde{\mathbf{D}}_i \qquad \text{with} \quad \widetilde{\mathbf{S}}_i^T \, \widetilde{M}_{ii}^{\mathcal{H}} \, \widetilde{\mathbf{S}}_i \;=\; \mathrm{Id}, \tag{7.1}$$

where the diagonal matrix $\widetilde{\mathbf{D}}_i \in \mathbb{R}^{k_i \times k_i}$ contains the $k_i \leq N_i$ smallest eigenvalues and the matrix $\widetilde{\mathbf{S}}_i \in \mathbb{R}^{N_i \times k_i}$ column-wise the associated eigenvectors. Because in general the matrices $\widetilde{K}_{ii}^{\mathcal{H}}$ and $\widetilde{M}_{ii}^{\mathcal{H}}$ slightly differ from $\widetilde{K}_{ii}$ and $\widetilde{M}_{ii}$, the corresponding eigensolutions (5.31) and (7.1) can differ as well. To indicate this difference in the $\mathcal{H}$-AMLS method bold symbols are used for the corresponding matrices and symbols. In the next step the block diagonal matrix

$\mathbf{Z} := \operatorname{diag}\left[\widetilde{\mathbf{S}}_1, \ldots, \widetilde{\mathbf{S}}_m\right]$ is defined and the reduced matrices

$$\widehat{\mathbf{K}} := \mathbf{Z}^T \widetilde{K}^{\mathcal{H}} \mathbf{Z} \in \mathbb{R}^{\bar{k}\times\bar{k}} \qquad \text{and} \qquad \widehat{\mathbf{M}} := \mathbf{Z}^T \widetilde{M}^{\mathcal{H}} \mathbf{Z} \in \mathbb{R}^{\bar{k}\times\bar{k}}$$

are computed where $\bar{k} = \sum_{i=1}^{m} k_i$. These matrices lead to the so-called *$\mathcal{H}$-reduced eigenvalue problem*

$$\begin{cases} \text{find } (\widehat{\boldsymbol{\lambda}}, \widehat{\mathbf{x}}) \in \mathbb{R} \times \mathbb{R}^{\bar{k}} \setminus \{0\} \text{ with} \\ \widehat{\mathbf{K}}\,\widehat{\mathbf{x}} = \widehat{\boldsymbol{\lambda}}\,\widehat{\mathbf{M}}\,\widehat{\mathbf{x}} \end{cases} \tag{7.2}$$

which possesses the eigenpairs

$$\left(\widehat{\boldsymbol{\lambda}}_j, \widehat{\mathbf{x}}_j\right)_{j=1}^{\bar{k}} \in \mathbb{R}_{>0} \times \mathbb{R}^{\bar{k}} \setminus \{0\} \qquad \text{with} \quad \widehat{\boldsymbol{\lambda}}_j \le \widehat{\boldsymbol{\lambda}}_{j+1}. \tag{7.3}$$

In the end of $\mathcal{H}$-AMLS the smallest n_{es} eigenpairs of (7.2) are computed and the eigenvectors $\widehat{\mathbf{x}}_j$ of the reduced problem are transformed via

$$\widehat{\mathbf{y}}_j \;:=\; (L^{\mathcal{H}})^{-T}\, \mathbf{Z}\, \widehat{\mathbf{x}}_j \qquad \text{for } j = 1, \ldots, n_{\text{es}}.$$

The computed pairs $\left(\widehat{\boldsymbol{\lambda}}_j, \widehat{\mathbf{y}}_j\right)_{j=1}^{n_{\text{es}}}$ are approximating the n_{es} smallest eigenpairs of the original problem (K, M). But note that in the beginning of $\mathcal{H}$-AMLS the original index ordering of the matrices K and M has been changed by some permutation $\pi : I \to I$ in order to obtain the needed matrix partitioning and to derive the corresponding $\mathcal{H}$-matrix representations. To receive eigenpair approximations of the problem (K, M) with the original ordering of the row and column indices, the inverse index permutation π^{-1} has to be applied to the vectors $\widehat{\mathbf{y}}_j$ for $j = 1, \ldots, n_{\text{es}}$.

In contrast to the classical AMLS method, in general $\widehat{\boldsymbol{\lambda}}_j$ is not equal to the Rayleigh quotient

$$\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})} \;:=\; \frac{\widehat{\mathbf{y}}_j^T\, K\, \widehat{\mathbf{y}}_j}{\widehat{\mathbf{y}}_j^T\, M\, \widehat{\mathbf{y}}_j} \tag{7.4}$$

since the matrix operations in (6.12) are performed only approximatively. Typically the Rayleigh quotients $\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}$ deliver better approximations of the sought eigenvalues λ_j than $\widehat{\boldsymbol{\lambda}}_j$, especially when the chosen accuracy ε of the $\mathcal{H}$-matrix arithmetic is coarse. Correspondingly, the Rayleigh quotients are computed as well in $\mathcal{H}$-AMLS.

Furthermore, as already described in Section 5.2.3, it is intended to apply a low-level nested dissection in the $\mathcal{H}$-AMLS method in order to keep the number of subproblems small so that the size $\bar{k}$ of the $\mathcal{H}$-reduced problem can be bounded by $\mathcal{O}(n_{\text{es}})$. In the case that in task ($\mathbb{T}4$) subdomain eigenvalue problems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ occur that are too large to be solved by a direct solver, the $\mathcal{H}$-AMLS method is applied recursively to this subproblem. However, note that in this case instead of an (almost) exact eigensolution of the discrete problem, as obtained by an classical eigensolver, an approximated eigensolution

$$\widetilde{K}_{ii}^{\mathcal{H}}\, \widetilde{\mathbf{S}}_i \;\approx\; \widetilde{M}_{ii}^{\mathcal{H}}\, \widetilde{\mathbf{S}}_i\, \widetilde{\mathbf{D}}_i \tag{7.5}$$

is obtained, and correspondingly an additional error is introduced which can influence the quality of the computed eigenpair approximations $\left(\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}, \widehat{\mathbf{y}}_j\right)_{j=1}^{n_{\text{es}}}$.

In the following the recursive version of $\mathcal{H}$-AMLS is simply called *recursive $\mathcal{H}$-AMLS*. In order to compare classical AMLS with the new (recursive) $\mathcal{H}$-AMLS method an overview of both methods is given in Table 7.1 where the tasks ($\mathbb{T}1$)–($\mathbb{T}8$) and a new task, denoted by task ($\mathbb{T}9$), are summarised.

Remark 7.1 *If the accuracy of the approximative $\mathcal{H}$-arithmetic is not fine enough it can potentially happen that the computed matrices $\widetilde{K}^{\mathcal{H}}$ and $\widetilde{M}^{\mathcal{H}}$ lose their positive definiteness (note that the FEM matrices K,M and their corresponding $\mathcal{H}$-matrix representations $K^{\mathcal{H}}$,$M^{\mathcal{H}}$ are positive definite), and that therefore the matrices $\widetilde{K}_{ii}^{\mathcal{H}}$, $\widetilde{M}_{ii}^{\mathcal{H}}$ and $\widehat{\mathbf{K}}, \widehat{\mathbf{M}}$ may lose their positive definiteness as well. In this case the problems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ and $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$ may become defective and eigensolutions of the form* (7.1) *and* (7.3) *do not exist. However, in numerical tests this problem did not occur, even when the accuracy of the $\mathcal{H}$-arithmetic has been chosen very coarse the matrices $\widetilde{K}^{\mathcal{H}}$ and $\widetilde{M}^{\mathcal{H}}$ retained the positive definiteness.*

7.2. Computational Costs

Beside the number of degrees of freedom N and the number of sought eigenpairs n_{es} the computational costs of $\mathcal{H}$-AMLS depend on the chosen accuracy ε of the $\mathcal{H}$-matrix operations in (6.12) and the applied modal truncation in (7.1), i.e, the number of selected eigenvectors k_i. A coarser accuracy ε and smaller k_i result in faster computations and reduced memory requirements of $\mathcal{H}$-AMLS. Of course these parameters can be chosen arbitrarily, however, their choice influences the approximation accuracy of the sought n_{es} eigenpairs. This issue is discussed in the next section and in Chapter 9. In the following the computational costs of recursive $\mathcal{H}$-AMLS are discussed in detail and compared with the costs of classical AMLS. The discussion is restricted to three-dimensional problems where $n_{\text{es}} \sim N^{\beta}$ eigenpairs are sought with some $\beta \in (0, 1/3]$.

The recursive version of $\mathcal{H}$-AMLS is applied in detail as follows: In task ($\mathbb{T}1$) a multi-level substructuring is performed with a fixed number of few levels $l_{\text{loc}} \in \mathbb{N}$. The substructuring results in $2^{l_{\text{loc}}}$ subdomain problems of approximate size $N/2^{l_{\text{loc}}}$, and $2^{l_{\text{loc}}} - 1$ interface eigenvalue problems of size $\mathcal{O}(N^{2/3})$ [cf. Section 5.3]. In task ($\mathbb{T}4$) the mode selection strategy proposed by Remark 5.7 is applied, i.e., the eigenvectors of $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ associated to the k_i smallest eigenvalues are computed where $k_i \in \mathcal{O}(N_i^{\beta_{\text{dom}}})$ for subdomain problems and $k_i \in \mathcal{O}(N_i^{\beta_{\text{int}}})$ for interface problems with suitable constants $\beta_{\text{dom}}, \beta_{\text{int}} \in (0, 1)$ fulfilling $\beta_{\text{dom}} \leq \beta$ and $\beta_{\text{int}} \leq 3\beta/2$. This mode selection strategy guarantees for three-dimensional problems that the size of the $\mathcal{H}$-reduced problem is bounded by $\mathcal{O}(n_{\text{es}})$, and is applied as well in Chapter 9 where numerical results are presented. In the case that in task ($\mathbb{T}4$) the size N_i of a subdomain problem $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ is larger than a given threshold $n_{\min}^{\text{AMLS}} \in \mathbb{N}$ then, instead of using a direct solver, the subdomain problem is solved recursively by $\mathcal{H}$-AMLS. The depth of the recursion $l_{\text{rec}} \in \mathbb{N}$ depends on the size N of the problem (K, M) and is described by

$$N \approx 2^{l_{\text{rec}}\, l_{\text{loc}}}\, n_{\min}^{\text{AMLS}} \;\Rightarrow\; l_{\text{rec}} \in \mathcal{O}(\log N).$$

Comparing the different tasks it can be seen that the recursive $\mathcal{H}$-AMLS method is much faster than the classical AMLS method:

- The computational costs of task ($\mathbb{T}1$) are negligible, $\mathcal{O}(N \log N)$. The computational costs for task ($\mathbb{T}2$) and ($\mathbb{T}3$) are of the order $\mathcal{O}(N \log^{\alpha} N)$ in $\mathcal{H}$-AMLS whereas in classical AMLS they are at least of the order $\mathcal{O}(N^2)$ for three-dimensional problems (cf. Section 5.3).

Task	Matrix Operations classical AMLS	Matrix Operations (recursive) $\mathcal{H}$-AMLS
($\mathbb{T}1$) partition K and M	nested dissection reordering, cf. (5.21) and (5.29)	nested dissection reordering, cf. (5.21) and (5.29), with subsequent geometric bisection (cf. Section 6.2)
($\mathbb{T}2$) block diagonalise the matrix K	$K = L\widetilde{K}L^T$ → **expensive** because of large-sized, dense interface matrices	$K^{\mathcal{H}} \approx L^{\mathcal{H}}\,\widetilde{K}^{\mathcal{H}}\,(L^{\mathcal{H}})^T$ → using **fast $\mathcal{H}$-matrix arithmetic** done in $\mathcal{O}(N\log^{\alpha}N)$
($\mathbb{T}3$) transform M	$\widetilde{M} = L^{-1}ML^{-T}$ → **expensive** because of large-sized, dense interface matrices	$\widetilde{M}^{\mathcal{H}} \approx (L^{\mathcal{H}})^{-1}\,M^{\mathcal{H}}\,(L^{\mathcal{H}})^{-T}$ → using **fast $\mathcal{H}$-matrix arithmetic** done in $\mathcal{O}(N\log^{\alpha}N)$
($\mathbb{T}4$) comp. partial eigensolutions (for $i = 1,\dots,m$)	$\widetilde{K}_{ii}\,\widetilde{S}_i = \widetilde{M}_{ii}\,\widetilde{S}_i\,\widetilde{D}_i$ with $\widetilde{S}_i \in \mathbb{R}^{N_i\times k_i}$ and $\widetilde{D}_i \in \mathbb{R}^{N_i\times k_i}$	$\widetilde{K}^{\mathcal{H}}_{ii}\,\widetilde{\mathbf{S}}_i = \widetilde{M}^{\mathcal{H}}_{ii}\,\widetilde{\mathbf{S}}_i\,\widetilde{\mathbf{D}}_i$ (… ≈ … *if recursive call*) with $\widetilde{\mathbf{S}}_i \in \mathbb{R}^{N_i\times k_i}$ and $\widetilde{\mathbf{D}}_i \in \mathbb{R}^{N_i\times k_i}$
if interface EVP	$\widetilde{K}_{ii}$ and $\widetilde{M}_{ii}$ are dense → **expensive**	$\widetilde{K}^{\mathcal{H}}_{ii}$ and $\widetilde{M}^{\mathcal{H}}_{ii}$ are $\mathcal{H}$-matrices → use **fast $\mathcal{H}$-matrix arithmetic** for eigensolver
if subdomain EVP	$\widetilde{K}_{ii} = K_{ii}$ and $\widetilde{M}_{ii} = M_{ii}$ are sparse → if small problem: **cheap** → if large problem: **expensive**	$\widetilde{K}^{\mathcal{H}}_{ii}$ and $\widetilde{M}^{\mathcal{H}}_{ii}$ are are $\mathcal{H}$-matrices → if small problem: **cheap** → if large problem: when $\mathcal{H}$-AMLS is used **cheap**, otherwise **expensive**
($\mathbb{T}5$) define subspace	$Z := \operatorname{diag}\left[\widetilde{S}_1,\dots,\widetilde{S}_m\right] \in \mathbb{R}^{N\times\bar{k}}$ with $\bar{k} = \sum_{i=1}^{m} k_i$	$\mathbf{Z} := \operatorname{diag}\left[\widetilde{\mathbf{S}}_1,\dots,\widetilde{\mathbf{S}}_m\right] \in \mathbb{R}^{N\times\bar{k}}$ with $\bar{k} = \sum_{i=1}^{m} k_i$
($\mathbb{T}6$) comp. matrices of reduced EVP	$\widehat{K} := Z^T\,\widetilde{K}\,Z \in \mathbb{R}^{\bar{k}\times\bar{k}}$, $\widehat{M} := Z^T\,\widetilde{M}\,Z \in \mathbb{R}^{\bar{k}\times\bar{k}}$	$\widehat{\mathbf{K}} := \mathbf{Z}^T\,\widetilde{K}^{\mathcal{H}}\,\mathbf{Z} \in \mathbb{R}^{\bar{k}\times\bar{k}}$, $\widehat{\mathbf{M}} := \mathbf{Z}^T\,\widetilde{M}^{\mathcal{H}}\,\mathbf{Z} \in \mathbb{R}^{\bar{k}\times\bar{k}}$ → use **fast $\mathcal{H}$-matrix arithmetic** for computation
($\mathbb{T}7$) solve the reduced EVP	$\widehat{K}\,\widehat{x}_j = \widehat{\lambda}_j\,\widehat{M}\,\widehat{x}_j \quad$ for $j = 1,\dots,n_{\text{es}}$	$\widehat{\mathbf{K}}\,\widehat{\mathbf{x}}_j = \widehat{\boldsymbol{\lambda}}_j\,\widehat{\mathbf{M}}\,\widehat{\mathbf{x}}_j \quad$ for $j = 1,\dots,n_{\text{es}}$
($\mathbb{T}8$) transform vectors	$\widehat{y}_j := L^{-T}Z\,\widehat{x}_j \quad$ for $j = 1,\dots,n_{\text{es}}$	$\widehat{\mathbf{y}}_j := (L^{\mathcal{H}})^{-T}\,\mathbf{Z}\,\widehat{\mathbf{x}}_j \quad$ for $j = 1,\dots,n_{\text{es}}$
($\mathbb{T}9$) comp. Rayl. quot.	—	$\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})} := \widehat{\mathbf{y}}_j^T K\widehat{\mathbf{y}}_j / \widehat{\mathbf{y}}_j^T M\widehat{\mathbf{y}}_j, \quad j = 1,\dots,n_{\text{es}}$
final eigenpair approx.	$(\widehat{\lambda}_j, \widehat{y}_j)$ with $j = 1,\dots,n_{\text{es}}$	$(\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}, \widehat{\mathbf{y}}_j)$ with $j = 1,\dots,n_{\text{es}}$

Table 7.1.: Overview of the classical AMLS and the new $\mathcal{H}$-AMLS method (*EVP* is the abbreviation for eigenvalue problem).

- Also the computation of the partial eigensolutions [task (𝕋4)] is faster in the $\mathcal{H}$-AMLS method: The submatrices $\widetilde{K}_{ii}^{\mathcal{H}}$ and $\widetilde{M}_{ii}^{\mathcal{H}}$ whose row and column indices are associated to an interface are data-sparse $\mathcal{H}$-matrices and not unstructured dense matrices as assumed in the classical AMLS method. Correspondingly an eigensolver exploiting the $\mathcal{H}$-matrix structure of $\widetilde{K}_{ii}^{\mathcal{H}}$ and $\widetilde{M}_{ii}^{\mathcal{H}}$ can be applied in (7.1) instead of an eigensolver for dense matrices as it is done in classical AMLS. Since interface eigenvalue problems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ are of size $N_i \in \mathcal{O}(N^{2/3})$ for three-dimensional problems, the almost linear scaling of $\mathcal{H}$-matrices allows us to solve for $k_i \in \mathcal{O}(n_{\text{es}})$ eigenvectors in complexity

$$\mathcal{O}\big(\, k_i\, N_i \log^{\alpha} N_i \,+\, k_i^2\, N_i \,\big) \;\leq\; \mathcal{O}\big(\, n_{\text{es}}\, N^{2/3} \log^{\alpha} N \,+\, n_{\text{es}}^2\, N^{2/3} \,\big), \tag{7.6}$$

  e.g. by using shift-invert Lanczos (see $\mathcal{H}$-SIL in Section 8.4). Since we assume $n_{\text{es}} \in \mathcal{O}(N^{1/3})$ and since the number of interface eigenvalue problems is bounded by $\mathcal{O}(1)$, it follows from (7.6) that the computational costs for the solution of all interface eigenvalue problems are bounded by $\mathcal{O}(n_{\text{es}}N)$.

  The number of subdomain problems is bounded as well by $\mathcal{O}(1)$. In the case that the subdomain problems are small enough (i.e., it holds $N_i \leq n_{\min}^{\text{AMLS}}$) the subproblems are solved by a direct solver leading to costs of the order $\mathcal{O}(1)$, otherwise the subdomain problems are solved recursively by the $\mathcal{H}$-AMLS method.

- Also the $\mathcal{H}$-matrix structure of $\widetilde{K}^{\mathcal{H}}$ and $\widetilde{M}^{\mathcal{H}}$ can be exploited in $\mathcal{H}$-AMLS using the fast $\mathcal{H}$-matrix-vector multiplication for the computation of the reduced matrices $\widehat{\mathbf{K}}, \widehat{\mathbf{M}} \in \mathbb{R}^{\bar{k}\times\bar{k}}$: The multiplications $\mathbf{S}^T(\widetilde{K}^{\mathcal{H}}\mathbf{S})$ and $\mathbf{S}^T(\widetilde{M}^{\mathcal{H}}\mathbf{S})$ involve $2\bar{k}$ $\mathcal{H}$-matrix times vector multiplications in $\mathcal{O}(N \log^{\alpha} N)$ plus $2\bar{k}^2$ scalar products of length N. Both together sum up to costs of the order

$$\mathcal{O}\big(\, \bar{k}\, N \log^{\alpha} N \,+\, \bar{k}^2\, N \,\big) \;\leq\; \mathcal{O}\big(\, n_{\text{es}}\, N \log^{\alpha} N \,+\, n_{\text{es}}^2\, N \,\big)$$

  for task (𝕋6). Since the scalar products can be computed with peak performance on todays workstations and compute servers, the costs for these are invisible in practice. The $2\bar{k}$ $\mathcal{H}$-matrix times vector multiplications are as well harmless since the logarithms and constants involved in the matrix vector multiplications are much smaller than for the $\mathcal{H}$-matrix operations in (6.12).

- The computational costs of task (𝕋7) are the same in both methods. The reduced eigenvalue problems $(\widehat{K}, \widehat{M})$ and $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$ are both of the same structure. Since it holds $\bar{k} \in \mathcal{O}(n_{\text{es}})$ and since we aim at $n_{\text{es}} \in \mathcal{O}(N^{1/3})$ eigenvalues, the size of the reduced problem allows us to use a dense linear algebra solver with cubic complexity whereby the computational costs still remain in $\mathcal{O}(N)$.

- Finally, for task (𝕋8) we can again exploit the fast $\mathcal{H}$-matrix times vector multiplication [backward substitution in $(L^{\mathcal{H}})^T$] to complete this task in

$$\mathcal{O}\big(\, n_{\text{es}}\, (\, \bar{k}\, N \,+\, N \log^{\alpha} N \,)\, \big) \;\leq\; \mathcal{O}\big(\, n_{\text{es}}\, N \log^{\alpha} N \,+\, n_{\text{es}}^2\, N \,\big)$$

  and for the Rayleigh Quotients in task (𝕋9) it is enough to use the sparsity of K and M to perform the computation in $\mathcal{O}(n_{\text{es}}N)$.

Hence, the computational costs of recursive $\mathcal{H}$-AMLS — without the costs of the recursive calls of $\mathcal{H}$-AMLS in task ($\mathbb{T}4$) — are bounded by $\mathcal{O}(n_{\text{es}} N \log^\alpha N + n_{\text{es}}^2 N)$. Defining $\widetilde{m} := 2^{l_{\text{loc}}}$ the overall costs, including the costs for the recursive calls, can then be bounded by

$$
\begin{aligned}
&\mathcal{O}\big(\, n_{\text{es}}\, N \log^\alpha N + n_{\text{es}}^2\, N \,\big) + \sum_{l=1}^{l_{\text{rec}}} \widetilde{m}^l\, \mathcal{O}\bigg(\Big(\frac{N}{\widetilde{m}^l}\Big)^{\beta_{\text{dom}}} \frac{N}{\widetilde{m}^l} \log^\alpha\Big(\frac{N}{\widetilde{m}^l}\Big) \;+\; \Big(\frac{N}{\widetilde{m}^l}\Big)^{2\beta_{\text{dom}}} \frac{N}{\widetilde{m}^l} \bigg) \\
&= \mathcal{O}\big(\, n_{\text{es}}\, N \log^\alpha N + n_{\text{es}}^2\, N \,\big) + \sum_{l=1}^{l_{\text{rec}}} \mathcal{O}\bigg(\Big(\frac{N}{\widetilde{m}^l}\Big)^{\beta_{\text{dom}}} N \log^\alpha N \;+\; \Big(\frac{N}{\widetilde{m}^l}\Big)^{2\beta_{\text{dom}}} N \bigg) \\
&\leq \mathcal{O}\big(\, n_{\text{es}}\, N \log^\alpha N + n_{\text{es}}^2\, N \,\big) + \sum_{l=1}^{l_{\text{rec}}} \mathcal{O}\bigg(\frac{n_{\text{es}}}{\widetilde{m}^{l\beta_{\text{dom}}}} N \log^\alpha N \;+\; \frac{n_{\text{es}}^2}{\widetilde{m}^{2l\beta_{\text{dom}}}} N \bigg) \\
&= \mathcal{O}\big(\, n_{\text{es}}\, N \log^\alpha N \,\big) \sum_{l=0}^{l_{\text{rec}}} \frac{1}{C^l} \;+\; \mathcal{O}\big(\, n_{\text{es}}^2\, N \,\big) \sum_{l=0}^{l_{\text{rec}}} \frac{1}{C^{2l}} \qquad \text{where } C := 2^{l_{\text{loc}}\,\beta_{\text{dom}}} \\
&= \mathcal{O}\big(\, n_{\text{es}}\, N \log^\alpha N + n_{\text{es}}^2\, N \,\big).
\end{aligned}
$$

We can sum up that the complexity of recursive $\mathcal{H}$-AMLS is bounded[1] by $\mathcal{O}(n_{\text{es}} N \log^\alpha N + n_{\text{es}}^2 N)$ and theoretically dominated by tasks ($\mathbb{T}6$) and ($\mathbb{T}8$). The operations involved in task ($\mathbb{T}6$) and task ($\mathbb{T}8$) are the $\mathcal{H}$-matrix times vector multiplication, which accumulates to a total of $\mathcal{O}(n_{\text{es}} N \log^\alpha N)$, and the usual scalar product accumulating to at most $\mathcal{O}(n_{\text{es}}^2 N)$ multiplications or additions. Both of these operations have extremely small constants involved and are therefore for problem sizes up to $N = 6,000,000$ not the bottleneck. Instead, most of the computational time is spent in the transformation steps ($\mathbb{T}2$) and ($\mathbb{T}3$), both of them in $\mathcal{O}(N \log^\alpha N)$ which is asymptotically in $o(n_{\text{es}} N)$. In the numerical examples we can observe that the costs for ($\mathbb{T}6$) and ($\mathbb{T}8$) are slowly increasing relative to the total cost, and that the total complexity stays in $\mathcal{O}(n_{\text{es}} N)$ for very large-scale problems.

7.3. Accuracy of the Eigenpair Approximation

The downside of faster computations and reduced memory requirements in $\mathcal{H}$-AMLS — achieved by a coarsening of the $\mathcal{H}$-matrix accuracy ε in task ($\mathbb{T}2$+$\mathbb{T}3$) and a reduction of the number of selected eigenvectors k_i in task ($\mathbb{T}4$) — is a possible loss in quality of the eigenpair approximations.

Keeping in mind the initial problem, the Rayleigh quotients $\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}$ in (7.4) are used to approximate the n_{es} smallest eigenvalues λ_j of the continuous problem (4.2). The corresponding approximation error is bounded by

$$
\underbrace{|\,\lambda_j - \widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}\,|}_{\substack{\text{error of the} \\ \mathcal{H}\text{-AMLS method}}} \;\leq\; \underbrace{|\,\lambda_j - \lambda_j^{(h)}\,|}_{\substack{\text{error caused by} \\ \text{the discretisation}}} \;+\; \underbrace{|\,\lambda_j^{(h)} - \widehat{\lambda}_j\,|}_{\substack{\text{error caused by the} \\ \text{modal truncation}}} \;+\; \underbrace{|\,\widehat{\lambda}_j - \widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}\,|}_{\substack{\text{error caused by the} \\ \mathcal{H}\text{-matrix approximation} \\ \text{(and the recursive approach)}}}
$$

[1] Note that the upper bound $\mathcal{O}(n_{\text{es}} N \log^\alpha N + n_{\text{es}}^2 N)$ for the computational costs of recursive $\mathcal{H}$-AMLS computing the $n_{\text{es}} \sim N^\beta$ smallest eigenpairs is obtained for all $\beta \in (0,1)$, i.e., also in the case when $\beta > 1/3$.

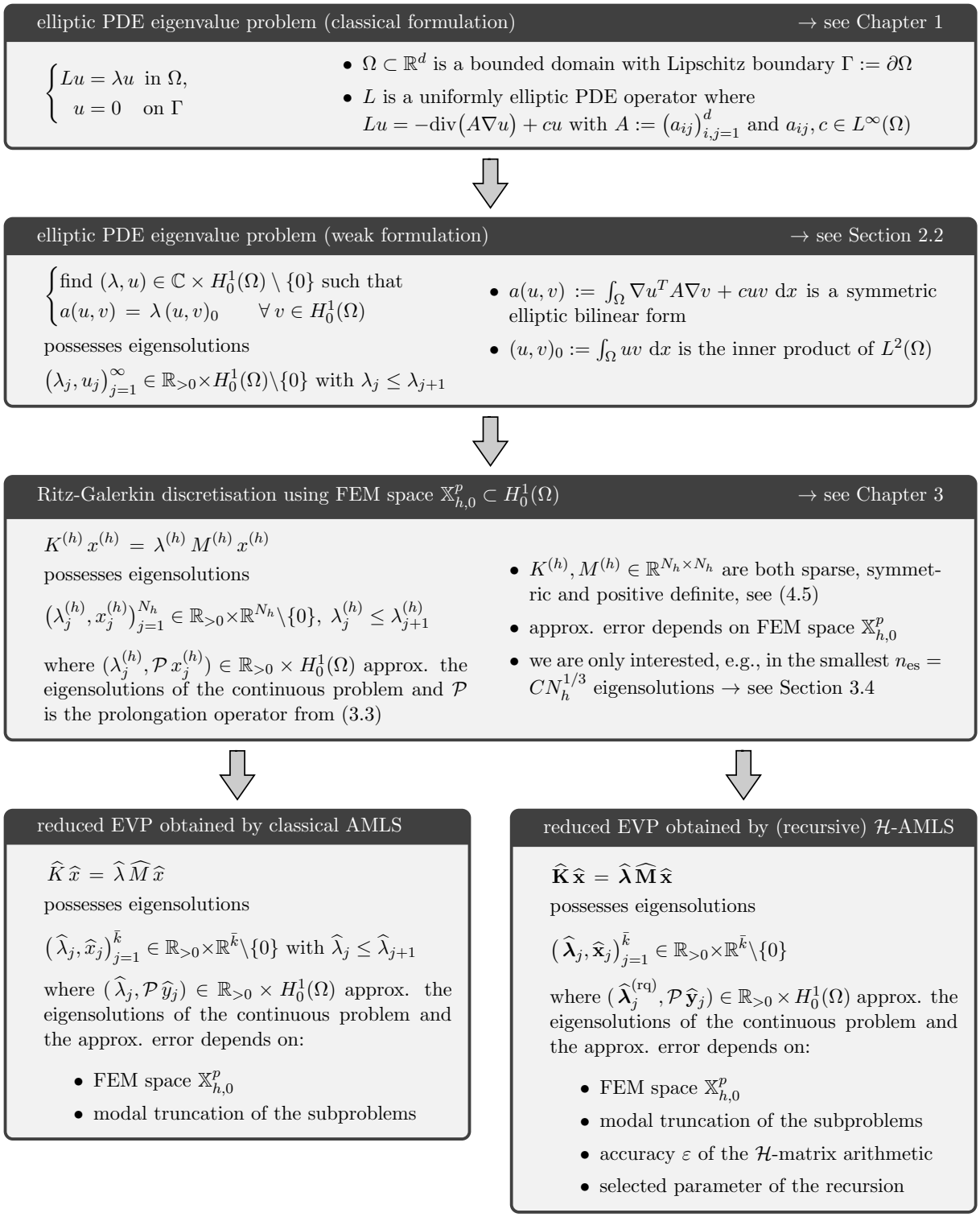


Figure 7.1.: Overview of the different eigenvalue problems and the interconnection between these problems.

where $\lambda_j^{(h)}$ is the eigenvalue of the discrete problem (4.4) and $\widehat{\lambda}_j$ is the eigenvalue of the reduced problem (5.26) from classical AMLS. The upper index of $\lambda_j^{(h)}$ is indicating the mesh width of the underlying finite element space V_h. In Figure 7.1 an overview of the different eigenvalue problems is given and the interconnection between them is summarised. The approximation error of the non-recursive $\mathcal{H}$-AMLS method is associated with the finite element discretisation, the modal truncation, and the $\mathcal{H}$-matrix approximation. The error caused by the modal truncation is influenced by the number of selected eigenvectors k_i in (7.1), and the error caused by the use of the approximative $\mathcal{H}$-matrix arithmetic is influenced by the chosen accuracy ε in (6.12). In the recursive version of $\mathcal{H}$-AMLS an additional error occurs when in task ($\mathbb{T}4$) the $\mathcal{H}$-AMLS method is applied recursively for the solution of large-sized subdomain problems. The approximation quality of the approximative eigensolution (7.5), and with it the approximation error of the recursive $\mathcal{H}$-AMLS method, is influenced by the chosen parameters k_i and ε in the recursive calls of $\mathcal{H}$-AMLS.

In contrast to the $\mathcal{H}$-AMLS method, the approximation error of classical approaches, like the block-SIL algorithm (cf. [36]), is only associated with the finite element discretisation because (almost) exact eigenvalues $\lambda_j^{(h)}$ of the discrete problem (4.4) are computed. The corresponding discretisation errors are used as reference values for the $\mathcal{H}$-AMLS method. To compete with a classical approach, the error caused by the modal truncation and the error caused by the approximative $\mathcal{H}$-matrix arithmetic have to be small enough that the error of the $\mathcal{H}$-AMLS method is of the same order as the discretisation error

$$\underbrace{|\,\lambda_j - \widehat{\boldsymbol{\lambda}}_j^{(\mathrm{rq})}\,|}_{\text{error of } \mathcal{H}\text{-AMLS}} \quad \approx \quad \underbrace{|\,\lambda_j - \lambda_j^{(h)}\,|}_{\text{discretisation error}}. \tag{7.7}$$

Dividing (7.7) by $|\lambda_j|$ one obtains the equivalent statement expressed in form of relative errors

$$\widehat{\boldsymbol{\delta}}_j := \underbrace{\frac{|\,\lambda_j - \widehat{\boldsymbol{\lambda}}_j^{(\mathrm{rq})}\,|}{\lambda_j}}_{\substack{\text{relative error of} \\ \mathcal{H}\text{-AMLS}}} \quad \approx \quad \underbrace{\frac{|\,\lambda_j - \lambda_j^{(h)}\,|}{\lambda_j}}_{\substack{\text{relative error} \\ \text{of discretisation}}} =: \widehat{\delta}_j^{(h)}. \tag{7.8}$$

Ultimately, the overarching aim arises to choose the parameters k_i and ε in such a way that the approximation error of $\mathcal{H}$-AMLS fulfils (7.8) while the computational costs and storage requirements of $\mathcal{H}$-AMLS are reduced as much as possible.

7.4. Improving the $\mathcal{H}$-AMLS Approximations with Subspace Iteration

The $\mathcal{H}$-AMLS method described in Section 7.1 is a projection method without any iterative improvement where the accuracy of the computed eigenpair approximations depends on the a priori chosen mode selection strategy and the accuracy of the $\mathcal{H}$-matrix arithmetic. However, even in the case that the accuracy of the computed eigenpair approximations is not satisfactory for some reason, the performed $\mathcal{H}$-AMLS computation was not useless since the computed eigenvector approximations are at least well suited as an initial subspace of a subsequent subspace iteration. Using the subspace iteration the accuracy of the $\mathcal{H}$-AMLS approximations can

Algorithm 3 Basic Subspace Iteration with Rayleigh-Ritz Projection

procedure SUBSPACEITERATION(K, M)
  initialise iteration matrix $Q^{(0)} \in \mathbb{R}^{N \times n_{\text{es}}}$ with full rank;
  ▷ *perform iteration where $n_{\text{iter}} \in \mathbb{N}$ is some given value*
  for $i = 1, \ldots, n_{\text{iter}}$ **do**
    M-orthonormalize $Q^{(i-1)}$;
    compute $A := MQ^{(i-1)} \in \mathbb{R}^{N \times n_{\text{es}}}$;
    solve $KQ^{(i)} = A$ for $Q^{(i)} \in \mathbb{R}^{N \times n_{\text{es}}}$;
  end for
  ▷ *perform Rayleigh-Ritz projection where $Q := Q^{(n_{\text{iter}})}$*
  compute reduced stiffness matrix $K_Q := Q^T A$; ▷ *note that $A = KQ$*
  compute reduced mass matrix $M_Q := Q^T M Q$;
  solve the reduced EVP $K_Q S_Q = M_Q S_Q D_Q$ with $S_Q^T M_Q S_Q = \mathrm{Id} \in \mathbb{R}^{n_{\text{es}} \times n_{\text{es}}}$;
  transform eigenvectors $S := Q S_Q$;
  return (D_Q, S); ▷ *return the computed eigensolutions*
end procedure

be improved up to the discretisation error if wanted, i.e., that (almost) exact eigenpairs of the problem (K, M) are computed. In the following, the subspace iteration is briefly recalled and it is explained why particularly this method is a good choice for the iterative improvement of the $\mathcal{H}$-AMLS approximations.

The subspace iteration [8, 10, 63] — which is sometimes also called orthogonal or simultaneous iteration — generalises the concept of the power iteration to multiple iteration vectors. The subspace iteration, as described in Algorithm 3, computes eigenpair approximations of the problem (K, M) associated to the smallest n_{es} eigenpairs, and provided that the initial subspace of the iteration is not M-orthogonal to one of the sought eigenvectors the convergence rate for the approximation of the j-th eigenpair (see, e.g., [10, 63]) is given by

$$\lambda_j / \lambda_{n_{\text{es}}+1} \qquad \text{for } j = 1, \ldots, n_{\text{es}}. \tag{7.9}$$

The Rayleigh-Ritz projection[2] in Algorithm 3 is needed to extract the corresponding eigenvector approximations from the iteration subspace which is spanned by the columns of the matrix $Q^{(i)}$.

To make out of Algorithm 3 an efficient and practically applicable eigensolver several improvements are needed (see, e.g., [8, 11] for details) such as: Implementation of shift-invert techniques to accelerate the convergence; implementation of a convergence check where already converged iteration-vectors are locked (i.e., converged vectors are extracted from the subsequent iteration process); performing the orthonormalization as infrequently as possible and only for those vectors for which it is necessary; and operating on a subspace whose dimension is a little bit larger than the number of sought eigenpairs (cf. [11]).

The subspace iteration is still widely used in practice because of its robustness and efficiency [11], but typically the Lanczos method [8] is more efficient especially when many eigenpairs are

[2] The Rayleigh-Ritz projection in Algorithm 3 includes the computation and the solution of the reduced eigenvalue problem (K_Q, M_Q), where the diagonal matrix $D_Q \in \mathbb{R}^{n_{\text{es}} \times n_{\text{es}}}$ contains the eigenvalues of (K_Q, M_Q) and $S_Q \in \mathbb{R}^{N \times n_{\text{es}}}$ column-wise the corresponding eigenvectors. The matrix D_Q together with the transformed eigenvector matrix $S := QS_Q$ finally provide the sought eigenpair approximations of the problem (K, M).

sought. However, in combination with $\mathcal{H}$-AMLS the subspace iteration provides important advantages: As mentioned in the beginning the eigenvector approximations of $\mathcal{H}$-AMLS (which are already quite close to the exact ones) are well suited for the construction of the initial subspace of the subspace iteration. The subspace iteration typically leads to a fast solution if the initial subspace is already close to the sought eigenspace. Correspondingly, it is reasonable that the accuracy of the $\mathcal{H}$-AMLS eigenpair approximations is improved sufficiently by the subspace iteration within a few iterations.[3] Furthermore, the factorisation $K^{\mathcal{H}} \approx L^{\mathcal{H}} \widetilde{K}^{\mathcal{H}} (L^{\mathcal{H}})^T$, computed in task ($\mathbb{T}2$) of the $\mathcal{H}$-AMLS method, can be reused as a preconditioner for the solution of the linear system in line 7 of Algorithm 3. For these two reasons the subspace iteration is a good choice for the iterative improvement of the $\mathcal{H}$-AMLS approximations. The efficiency of this approach has already been demonstrated in [68] in a purely algebraic setting. There a basic version of the subspace iteration (without any acceleration techniques) has been used to improve the eigenpair approximations of the classical AMLS method. It is shown in [68] for several problems that the accuracy of the eigenvalue and eigenvector approximations of AMLS can be significantly improved already within one or two iteration steps. The same results can be expected when $\mathcal{H}$-AMLS is combined with the subspace iteration. In particular, when $\mathcal{H}$-AMLS is applied with a subsequent subspace iteration the $\mathcal{H}$-AMLS method becomes less dependent from the a priori chosen mode selection strategy and the chosen accuracy of the $\mathcal{H}$-arithmetic, since the accuracy of the $\mathcal{H}$-AMLS approximations can be improved in any case up to the discretisation error if wanted. This should allow $\mathcal{H}$-AMLS to become applicable to a wider range of problems, e.g., to problems where a high accuracy of the computed eigenpair approximations has to be guaranteed.

The efficiency of the subspace iteration for the iterative improvement of the $\mathcal{H}$-AMLS approximations (the accuracy can be significantly improved within one iteration step, cf. [68]) motivates to introduce a variation of the in Section 7.1 presented $\mathcal{H}$-AMLS method: Instead of applying task ($\mathbb{T}9$) it is proposed to apply Algorithm 4 right after finishing task ($\mathbb{T}8$). Let $\mathbf{S} \in \mathbb{R}^{N \times n_{\text{es}}}$ be the matrix containing column-wise the $\mathcal{H}$-AMLS eigenvector approximations $\widehat{\mathbf{y}}_j$ which have been computed in task ($\mathbb{T}8$), then Algorithm 4 performs one (approximative) iteration step of the subspace iteration using $\mathbf{S}$ as initial subspace. Note that, when in task ($\mathbb{T}7$) a partial eigensolution of the $\mathcal{H}$-reduced eigenvalue problem of the form

$$\widehat{\mathbf{K}}\,\widehat{\mathbf{S}} \;=\; \widehat{\mathbf{M}}\,\widehat{\mathbf{S}}\,\widehat{\mathbf{D}} \qquad \text{with} \quad \widehat{\mathbf{S}}^T\,\widehat{\mathbf{M}}\,\widehat{\mathbf{S}} \;=\; \mathrm{Id}$$

is computed where the diagonal matrix $\widehat{\mathbf{D}} \in \mathbb{R}^{n_{\text{es}} \times n_{\text{es}}}$ contains the eigenvalues $(\widehat{\boldsymbol{\lambda}}_j)_{j=1}^{n_{\text{es}}}$ and $\widehat{\mathbf{S}} \in \mathbb{R}^{\bar{k} \times n_{\text{es}}}$ column-wise the associated eigenvectors $(\widehat{\mathbf{x}}_j)_{j=1}^{n_{\text{es}}}$, then the matrix $\mathbf{S}$ is approximatively M-orthonormal. Furthermore, note that the computation of $K^{-1}(M\mathbf{S})$ in Algorithm 4 is performed only approximatively, and correspondingly the error of the improved eigenvector approximations is still associated with the error induced by the approximative $\mathcal{H}$-arithmetic. The evaluation of $K^{-1}(MS)$ can be performed also exactly by using the factorisation of $K^{\mathcal{H}}$ as a preconditioner, however, this would further increase the computational costs.

Note that, when Algorithm 4 is applied in the $\mathcal{H}$-AMLS method right after task ($\mathbb{T}8$) then the Rayleigh quotients of the improved eigenvectors do not have to be computed, since already

[3] In particular, the convergence for the eigenpairs at the lower end of the spectrum should be faster because of the convergence rate (7.9).

Algorithm 4 Implementation of Task (𝕋SI)

procedure ImproveEigensolution(K, M, $L^{\mathcal{H}}$, $\widetilde{K}^{\mathcal{H}}$, $\mathbf{S}$)
  ▷ *perform one (approximative) iteration step of the subspace iteration*
  compute $A_1 := M\mathbf{S} \in \mathbb{R}^{N \times n_{\text{es}}}$; ▷ *note that K and M are sparse*
  solve $L^{\mathcal{H}} A_2 = A_1$ for A_2 using fast $\mathcal{H}$-arithmetic; ▷ *$L^{\mathcal{H}}$ is a lower triangular matrix*
  compute $A_3 := (\widetilde{K}^{\mathcal{H}})^{-1} A_2$ using fast $\mathcal{H}$-arithmetic; ▷ *$\widetilde{K}^{\mathcal{H}}$ is a block diagonal matrix*
  solve $(L^{\mathcal{H}})^T A_4 = A_3$ for A_4 using fast $\mathcal{H}$-arithmetic;
  ▷ *perform the Rayleigh-Ritz projection where $Q := A_4$*
  compute reduced stiffness matrix $K_Q := Q^T K Q$;
  compute reduced mass matrix $M_Q := Q^T M Q$;
  solve the reduced EVP $K_Q S_Q = M_Q S_Q D_Q$ with $S_Q^T M_Q S_Q = \mathrm{Id} \in \mathbb{R}^{n_{\text{es}} \times n_{\text{es}}}$;
  transform eigenvectors $S_{\text{new}} := Q S_Q$;
  return (D_Q, S_{new}); ▷ *return improved eigensolution*
end procedure

the equality

$$\widehat{\boldsymbol{\lambda}}_j^{(\text{SI})} = \frac{(\widehat{\mathbf{y}}_j^{(\text{SI})})^T K \, \widehat{\mathbf{y}}_j^{(\text{SI})}}{(\widehat{\mathbf{y}}_j^{(\text{SI})})^T M \, \widehat{\mathbf{y}}_j^{(\text{SI})}} \qquad \text{for } j = 1, \ldots, n_{\text{es}}$$

is fulfilled, where $\widehat{\boldsymbol{\lambda}}_j^{(\text{SI})}$ denotes the j-th diagonal entry of the matrix D_Q from Algorithm 4 and $\widehat{\mathbf{y}}_j^{(\text{SI})}$ is the j-th column vector of the matrix S_{new}. In particular, the computed eigenvector approximations $\widehat{\mathbf{y}}_j^{(\text{SI})}$ are M-orthonormal. In the following Algorithm 4 is referred to as task (𝕋SI), and the $\mathcal{H}$-AMLS method is simply called *$\mathcal{H}$-AMLS with (𝕋SI)-improvement* when task (𝕋SI) is performed instead of task (𝕋9).

The computational costs of task (𝕋SI) are as follows:

- The computation of $M\mathbf{S}$ is performed in $\mathcal{O}(n_{\text{es}} N)$ using the sparsity of M.
- The computation of the matrices $A_2, A_4 \in \mathbb{R}^{N \times n_{\text{es}}}$ in Algorithm 4 [forward and backward substitution in $L^{\mathcal{H}}$ and $(L^{\mathcal{H}})^T$] can be performed column-wise in $\mathcal{O}(n_{\text{es}} N \log^{\alpha} N)$ using the fast $\mathcal{H}$-matrix arithmetic.
- The multiplication $(K^{\mathcal{H}})^{-1} A_2$ is performed in $\mathcal{O}(n_{\text{es}} N \log^{\alpha} N)$ exploiting the fast $\mathcal{H}$-arithmetic for the computation of the inverse $(K^{\mathcal{H}})^{-1}$ and for the computation of n_{es} $\mathcal{H}$-matrix times vector multiplications.
- The multiplications $Q^T(KQ)$ and $Q^T(MQ)$ involve $2n_{\text{es}}$ sparse matrix times vector multiplications in $\mathcal{O}(N)$ plus $2n_{\text{es}}^2$ scalar products of length N which sum up to costs of the order $\mathcal{O}(n_{\text{es}}^2 N)$.
- Since we aim at $n_{\text{es}} \in \mathcal{O}(N^{1/3})$ the solution of eigenvalue problem (K_Q, M_Q) in Algorithm 4 can be performed in $\mathcal{O}(N)$ by a dense linear algebra solver with cubic complexity.
- The multiplication QS_Q involves Nn_{es} scalar products of length n_{es} leading to costs of $\mathcal{O}(n_{\text{es}}^2 N)$.

We can sum up that the complexity of task ($\mathbb{T}$SI) is dominated by the $\mathcal{H}$-matrix times vector operations, which accumulates to a total of $\mathcal{O}(n_{\text{es}} N \log^{\alpha} N)$, and the usual scalar product accumulating to at most $\mathcal{O}(n_{\text{es}}^2 N)$ multiplications or additions. As described in Section 7.2, both of these operations have constants involved that are much smaller than the constants involved in the $\mathcal{H}$-matrix operations in (6.12). Therefore, in $\mathcal{H}$-AMLS task ($\mathbb{T}$SI) is for problem sizes up to $N = 6,000,000$ not the bottleneck, instead most of the time is spent in task ($\mathbb{T}$2) and ($\mathbb{T}$3), both of them in $\mathcal{O}(N \log^{\alpha} N)$ which is asymptotically in $o(n_{\text{es}} N)$.

Last but not least another feature is listed which possibly allows the application of $\mathcal{H}$-AMLS to a wider range of problems:

Remark 7.2 (Black Box $\mathcal{H}$-AMLS) *To perform the $\mathcal{H}$-AMLS method the $\mathcal{H}$-matrix representations of K and M are needed. These $\mathcal{H}$-matrix representations are based on a block cluster tree $\mathcal{T}_{I\times I}$ and an admissibility condition that describe a partitioning of K and M into submatrices that are stored in full matrix or $\mathrm{R}(k)$-matrix representation (cf. Section 6.1). To construct the needed block cluster tree geometry information of the underlying finite element space is needed in order to determine a geometric partitioning of the index set I, and to determine diameters and distances of clusters for the admissibility condition. The $\mathcal{H}$-matrix arithmetic itself, which is used in the $\mathcal{H}$-AMLS method, requires only the $\mathcal{H}$-matrix format induced by $\mathcal{T}_{I\times I}$ and does not need any geometry information. However, in some applications geometry information might not be available, and only the already assembled stiffness matrix K and mass matrix M can be provided. In this case, the black box approach presented in [34] can be used to construct the necessary block cluster tree $\mathcal{T}_{I\times I}$ and an algebraic admissibility condition. This black box approach is based on the matrix graph of the sparse matrices K and M, and uses the connectivity information of the indices $i \in I$ to derive, e.g., a partitioning of the index set I. Using this approach geometry data associated with the indices is no longer required, and hence the $\mathcal{H}$-AMLS method can be applied as well in a purely algebraic setting.*

8. Implementation of $\mathcal{H}$-AMLS

The $\mathcal{H}$-AMLS method has been implemented in C++ using the $\mathcal{H}$-matrix software library *HLIBpro* v2.3. The HLIBpro [50, 53] is a C++ library which provides a complete implementation of the available $\mathcal{H}$-matrix arithmetic, and which contains a wide range of routines for $\mathcal{H}$-matrix construction, index set clustering, low rank approximation, etc. Moreover, the HLIBpro has been parallelised for shared and distributed memory machines [51, 52, 54]. Beside that, a LAPACK/BLAS [3, 56] library has been used for the implementation of $\mathcal{H}$-AMLS since a direct eigensolver routine is needed, and since a corresponding library is required by the HLIBpro itself (the HLIBpro uses LAPACK/BLAS for all low-level linear algebra functions).

This chapter describes the implementation of the $\mathcal{H}$-AMLS method. In particular, it is shown how the computations involved in $\mathcal{H}$-AMLS are performed in detail and how basic parameters of $\mathcal{H}$-AMLS have been chosen which lead to the numerical results presented in Chapter 9. Furthermore, most of the tasks of $\mathcal{H}$-AMLS are very well parallelisable. To benefit from the multiple cores of today's workstations and compute servers the $\mathcal{H}$-AMLS method has been parallelised for shared memory systems. How the different tasks of $\mathcal{H}$-AMLS have been parallelised is discussed here as well. But before this topic is started the following Lemma is presented which enables to change the block diagonalisation in task ($\mathbb{T}2$) of the $\mathcal{H}$-AMLS method:

Lemma 8.1 *Consider the transformation matrix L arising in task ($\mathbb{T}2$) of the classical AMLS method when the matrix K is block diagonalised. Let $B := \mathrm{diag}[B_1, \ldots, B_m]$ be a block diagonal matrix consisting of arbitrary regular matrices $B_i \in \mathbb{R}^{N_i \times N_i}$ for $i = 1, \ldots, m$. If in the AMLS method the transformation is performed with matrix $L_B := LB$ instead of L then the computed eigenvector approximations $(\widehat{\lambda}_j, \widehat{y}_j)$ remain unchanged.*

Proof: If in classical AMLS the transformation matrix L is replaced by L_B then in task ($\mathbb{T}2$) and ($\mathbb{T}3$) of AMLS the transformed matrices $\widetilde{K}_B := (L_B)^{-1} K (L_B)^{-T}$ and $\widetilde{M}_B := (L_B)^{-1} M (L_B)^{-T}$ are computed instead of $\widetilde{K}$ and $\widetilde{M}$, and it holds

$$\widetilde{K}_B = B^{-1} \widetilde{K} B^{-T} \quad \text{and} \quad \widetilde{M}_B = B^{-1} \widetilde{M} B^{-T}$$

with $B^{-1} = \mathrm{diag}[B_1^{-1}, \ldots, B_m^{-1}]$. Denote $(\widetilde{K}_B)_{ii}$ and $(\widetilde{M}_B)_{ii}$ for $i = 1, \ldots, m$ the submatrices of $\widetilde{K}_B$ and $\widetilde{M}_B$ in block row and block column i. Then in task ($\mathbb{T}4$) of AMLS the partial eigensolutions of the subproblems $\left((\widetilde{K}_B)_{ii}, (\widetilde{M}_B)_{ii} \right)$ are computed where it holds $(\widetilde{K}_B)_{ii} = B_i^{-1} \widetilde{K}_{ii} B_i^{-T}$ and $(\widetilde{M}_B)_{ii} = B_i^{-1} \widetilde{M}_{ii} B_i^{-T}$. Note that the eigenvalue problems $\left((\widetilde{K}_B)_{ii}, (\widetilde{M}_B)_{ii} \right)$ and $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ are equivalent, i.e., the eigenvalues of both problems coincide and the eigenvectors are transformed. More precisely, if in task ($\mathbb{T}4$) a mode selection strategy is applied which only depends on the size of the subproblem eigenvalues (see, e.g., Remark 5.7) then the partial eigensolution of $\left((\widetilde{K}_B)_{ii}, (\widetilde{M}_B)_{ii} \right)$ is given by $(\widetilde{D}_i, B_i^T \widetilde{S}_i)$ where $(\widetilde{D}_i, \widetilde{S}_i)$ is the partial eigensolution of $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$. Correspondingly, in task ($\mathbb{T}5$) of AMLS the subspace spanned by the columns of the matrix $Z_B := B^T Z$ is used for the reduction, and in task ($\mathbb{T}6$) the reduced matrices $\widehat{K}_B := Z_B^T \widetilde{K}_B Z_B$

Algorithm 5 Implementation of Non-Recursive and Recursive $\mathcal{H}$-AMLS

▷ *parameters that are used in $\mathcal{H}$-AMLS and their corresponding default values*
parameter {
  do_recursive_HAMLS:=false; ▷ *switch between recursive and non-recursive version*
  apply_task_(TSI):=false; ▷ *switch to activate improvement task (TSI)*
  $n_{\min} := 40$, $\eta := 50$; ▷ *parameters controlling shape of block cluster tree $\mathcal{T}_{I\times I}$*
  $n_{\min}^{\mathrm{AMLS}} := 1000$; ▷ *parameter controlling the size of subdomain problems*
  ε; ▷ *relative accuracy of the used $\mathcal{H}$-arithmetic → see Chapter 9*
  β_{int}, β_{dom}, C_{int}, C_{dom}, $C_{\mathrm{dom}}^{\approx}$; ▷ *control number of selected eigenvectors → see Chapter 9*
  n_{es}; ▷ *number of sought eigenpairs*
};

procedure ApplyHAMLS(K, M, $(\xi_i)_{i=1}^N$, parameter)
  ▷ *task (T1): construct the $\mathcal{H}$-matrix representations of K and M → see Section 8.1*
  $I := \{1, \ldots, N\}$;
  $\mathcal{T}_I :=$ ConstructClusterTree(I, $(\xi_i)_{i\in I}$, $n_{\min}$); ▷ *see Algorithm 6*
  $\mathcal{T}_{I\times I} :=$ ConstructBlockClusterTree($\mathcal{T}_I$, $n_{\min}$, η); ▷ *see Algorithm 2*
  reorder rows/columns of K and M to obtain $K^{\mathcal{H}}, M^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$;

  ▷ *task (T2+T3): transform eigenvalue problem ($K^{\mathcal{H}}$, $M^{\mathcal{H}}$) → see Section 8.2, Algorithm 7*
  $(\widetilde{K}^{\mathcal{H}}, \widetilde{M}^{\mathcal{H}}, L^{\mathcal{H}}) :=$ TransformEVP($K^{\mathcal{H}}$, $M^{\mathcal{H}}$, $\mathcal{T}_{I\times I}$, ε);

  ▷ *initialise the auxiliary data → see Section 8.3 and Algorithm 8*
  $\mathcal{T}_{\mathrm{AMLS}} :=$ CreateSubstructureTree($\mathcal{T}_I$, $n_{\min}^{\mathrm{AMLS}}$);
  $m := \#\mathcal{L}(\mathcal{T}_{\mathrm{AMLS}})$; ▷ *$m$ is the number of subproblems*
  create the bijection $\psi : \{1, \ldots, m\} \to \mathcal{L}(\mathcal{T}_{\mathrm{AMLS}})$; ▷ *see (8.5) for the definition of ψ*

  ▷ *task (T4): compute partial eigensolution of subproblems → see Section 8.4, Algorithm 9*
  $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=1}^m :=$ CompPartialEigensol($\widetilde{K}^{\mathcal{H}}$, $\widetilde{M}^{\mathcal{H}}$, C_{dom}, C_{int}, β_{dom}, β_{int});

  ▷ *apply the condensation of subproblems → see Section 8.7 and Algorithm 16*
  ▷ *note that $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)$ and $\mathcal{T}_{\mathrm{AMLS}}$ can change during the condensation process*
  if do_recursive_HAMLS **then**
    ApplyCondensation($\widetilde{K}^{\mathcal{H}}$, $\widetilde{M}^{\mathcal{H}}$, $\mathcal{T}_{\mathrm{AMLS}}$, $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=1}^m$, $C_{\mathrm{dom}}^{\approx}$, β_{dom});
  end if

  ▷ *task (T6): compute the reduced matrices $\widehat{\mathbf{K}}$ and $\widehat{\mathbf{M}}$ → see Section 8.5 and Algorithm 11*
  $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}}) :=$ CompReducedMatrices($\widetilde{K}^{\mathcal{H}}$, $\widetilde{M}^{\mathcal{H}}$, $\mathcal{T}_{\mathrm{AMLS}}$, $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=1}^m$);

  ▷ *task (T7): compute partial eigensol. of the reduced eigenvalue problem → see Section 8.5*
  solve $\widehat{\mathbf{K}}\,\widehat{\mathbf{S}} = \widehat{\mathbf{M}}\,\widehat{\mathbf{S}}\,\widehat{\mathbf{D}}$ with $\widehat{\mathbf{S}}^T\,\widehat{\mathbf{M}}\,\widehat{\mathbf{S}} = \mathrm{Id}$ where $\widehat{\mathbf{D}} \in \mathbb{R}^{n_{\mathrm{es}}\times n_{\mathrm{es}}}$, $\widehat{\mathbf{S}} \in \mathbb{R}^{\bar{k}\times n_{\mathrm{es}}}$;

  ▷ *task (T8+T9+TSI): transform and improve eigensolutions → see Section 8.6, Alg. 12*
  $(\mathbf{D}, \mathbf{S}) :=$ TransformEigensol(K, M, $L^{\mathcal{H}}$, $\widetilde{K}^{\mathcal{H}}$, $(\widetilde{\mathbf{S}}_i)_{i=1}^m$, $\widehat{\mathbf{S}}$, apply_task_TSI);

  return $(\mathbf{D}, \mathbf{S})$;
end procedure

and $\widehat{M}_B := Z_B^T \widetilde{M}_B Z_B$ are computed. Since

$$\widehat{K}_B = Z^T B B^{-1} \widetilde{K} B^{-T} B^T Z = \widehat{K} \qquad \text{and} \qquad \widehat{M}_B = Z^T B B^{-1} \widetilde{M} B^{-T} B^T Z = \widehat{M}$$

it follows that the reduced eigenvalue problem which is obtained by using the transformation matrix L_B coincides with the reduced problem obtained by using transformation matrix L, and hence in task ($\mathbb{T}7$) the same eigenpairs $(\widehat{\lambda}_j, \widehat{x}_j)_{j=1}^{n_{\text{es}}}$ are computed. Furthermore, when in task ($\mathbb{T}8$) the eigenvectors $\widehat{x}_j$ are transformed we obtain

$$(L_B)^{-T} Z_B \, \widehat{x}_j = (LB)^{-T} B^T Z \, \widehat{x}_j = L^{-T} Z \, \widehat{x}_j = \widehat{y}_j.$$

Altogether, it follows that the same eigenpair approximations are computed when in AMLS the transformation matrix L_B is used instead of L. ■

According to Lemma 8.1 the AMLS method can be applied with a modified transformation matrix without changing the computed eigenpair approximations $(\widehat{\lambda}_j, \widehat{y}_j)_{j=1}^{n_{\text{es}}}$. For example, instead of performing in task ($\mathbb{T}2$) the block LDL$^{\text{T}}$-factorisation $K = L\widetilde{K}L^T$ [see, e.g., (5.22)] also the *complete* LDL$^{\text{T}}$-factorisation can be performed, i.e., that the factorisation $K = L_D D L_D^T$ is computed where D is a diagonal matrix and L_D a lower triangular matrix with unit diagonal. Since the matrix L_D can be written as $L_D = LB$ with a suitable regular block diagonal matrix B the result of the AMLS approximation remains unchanged according to Lemma 8.1. By a similar argumentation as done in the proof of Lemma 8.1 it can be shown that also in the recursive version of AMLS the computed eigenpair approximations remain unchanged when instead of the block diagonalisation a complete diagonalisation is performed.

Performing a complete LDL$^{\text{T}}$-factorisation in task ($\mathbb{T}2$) of the $\mathcal{H}$-AMLS method has the following advantages: The (complete) LDL$^{\text{T}}$-factorisation of a symmetric $\mathcal{H}$-matrix $K^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$ is already implemented in the HLIBpro library [50] with a computational complexity of $\mathcal{O}(N \log^{\alpha} N)$. Correspondingly, there is no need to implement an explicit block LDL$^{\text{T}}$-factorisation and instead the well proven and tested HLIBpro routine can be used in task ($\mathbb{T}2$) for the transformation of the stiffness matrix. Furthermore, this HLIBpro routine is already parallelised very efficiently for shared memory systems [54]. Another advantage of performing a complete LDL$^{\text{T}}$-factorisation in task ($\mathbb{T}2$) is that this eases the implementation of the recursive $\mathcal{H}$-AMLS method. Keep in mind that when in task ($\mathbb{T}2$) a block LDL$^{\text{T}}$-factorisation is performed that then the submatrices on the block diagonal of the transformed matrices $\widetilde{K}$ and $\widetilde{M}$ remain unchanged if they are associated with a subdomain, i.e., it holds $\widetilde{K}_{ii}^{\mathcal{H}} = K_{ii}^{\mathcal{H}}$ and $\widetilde{M}_{ii}^{\mathcal{H}} = M_{ii}^{\mathcal{H}}$. If in task ($\mathbb{T}4$) the partial eigensolution of subproblem $(K_{ii}^{\mathcal{H}}, M_{ii}^{\mathcal{H}})$ should be computed recursively by $\mathcal{H}$-AMLS then again the matrix $K_{ii}^{\mathcal{H}}$ has to be block diagonalised and $M_{ii}^{\mathcal{H}}$ has to be transformed correspondingly. If in task ($\mathbb{T}2$) instead a complete LDL$^{\text{T}}$-factorisation is performed, in connection with the appropriate matrix partitioning of $K^{\mathcal{H}}$ and $M^{\mathcal{H}}$ (cf. Section 8.1), then the submatrices $\widetilde{K}_{ii}^{\mathcal{H}}$ are already diagonalised and no further transformation is needed when $\mathcal{H}$-AMLS is applied recursively. This means when a complete LDL$^{\text{T}}$-factorisation is performed then the problem $(K^{\mathcal{H}}, M^{\mathcal{H}})$ is transformed globally. In this light, the recursive application of $\mathcal{H}$-AMLS can be seen as a condensation process (see Section 8.7 for details) where the spectral information of several subproblems is condensed using $\mathcal{H}$-AMLS into the (approximative) spectral information of a single subproblem of larger size.

The $\mathcal{H}$-AMLS method has been implemented using the approach discussed above, i.e., where in task ($\mathbb{T}2$) a complete $\mathrm{LDL}^{\mathrm{T}}$-factorisation is performed and the eigenvalue problem $(K^{\mathcal{H}}, M^{\mathcal{H}})$ is transformed globally. The implementation of $\mathcal{H}$-AMLS is summarised in Algorithm 5. For given finite element matrices $K, M \in \mathbb{R}^{N\times N}$ from (5.20) and associated geometric representatives $(\xi_i)_{i=1}^N$ from (6.7) Algorithm 5 computes the approximative partial eigensolution

$$K\,\mathbf{S} \;\approx\; M\,\mathbf{S}\,\mathbf{D} \qquad \text{with} \quad \mathbf{S}^T M\,\mathbf{S} \;\approx\; \mathrm{Id} \tag{8.1}$$

where $\mathbf{D} \in \mathbb{R}^{n_{\mathrm{es}}\times n_{\mathrm{es}}}$ is a diagonal matrix containing approximations of the n_{es} smallest eigenvalues and $\mathbf{S} \in \mathbb{R}^{N\times n_{\mathrm{es}}}$ the matrix containing column-wise approximations of the corresponding eigenvectors. Algorithm 5 includes the implementation of the non-recursive and recursive version of $\mathcal{H}$-AMLS, and as well the version with ($\mathbb{T}$SI)-improvement.[1] In the following Algorithm 5 is described in detail. The description starts with the implementation of non-recursive $\mathcal{H}$-AMLS (Section 8.1–8.6), and based on this, it is shown how recursive $\mathcal{H}$-AMLS has been implemented (Section 8.7). Finally, in Section 8.8 it is described how $\mathcal{H}$-AMLS has been parallelised for shared memory systems.

8.1. Task ($\mathbb{T}$1): Construction of the $\mathcal{H}$-matrices

Algorithm 6 Construction of the Cluster Tree

```
1: procedure ConstructClusterTree( t, (ξ_i)_{i∈t}, n_min )
2:     if #t ≤ n_min then
3:         S(t) := ∅;
4:     else if t is a domain-cluster then
5:         geometric dissection of t into domain-clusters t_1, t_2 and interface-cluster t_3;
6:         S(t) := {t_1, t_2, t_3};
7:     else if t is an interface-cluster then
8:         ▷ level(t) is defined as the length of the path between t ∈ T_I and the root of T_I
9:         if level(t) ≡ 0 mod d then
10:            S(t) := {t};
11:        else
12:            geometric bisection of t into interface-clusters t_1 and t_2;
13:            S(t) := {t_1, t_2};
14:        end if
15:    end if
16:    for all s ∈ S(t) do
17:        ConstructClusterTree( s, (ξ_i)_{i∈s}, n_min )
18:    end for
19:    return cluster t;
20: end procedure
```

In order to compute in (6.12) the transformed eigenvalue problem $(\widetilde{K}^{\mathcal{H}}, \widetilde{M}^{\mathcal{H}})$ by the fast $\mathcal{H}$-matrix arithmetic, first of all, the matrices K and M have to be represented in the matrix format $\mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$. For this purpose a cluster tree $\mathcal{T}_I$ over the index set $I = \{1, \ldots, N\}$ has to

[1] Depending on whether task ($\mathbb{T}$9) or task ($\mathbb{T}$SI) has been applied in Algorithm 5 the diagonal matrix $\mathbf{D}$ contains the eigenvalue approximations $\widehat{\boldsymbol{\lambda}}_j^{(\mathrm{rq})}$ or $\widehat{\boldsymbol{\lambda}}_j^{(\mathrm{SI})}$, and the matrix $\mathbf{S}$ column-wise the eigenvector approximations $\widehat{\mathbf{y}}_j$ or $\widehat{\mathbf{y}}_j^{(\mathrm{SI})}$. Furthermore, it holds $\mathbf{S}^T M\mathbf{S} = \mathrm{Id}$ in (8.1) when task ($\mathbb{T}$SI) has been applied.

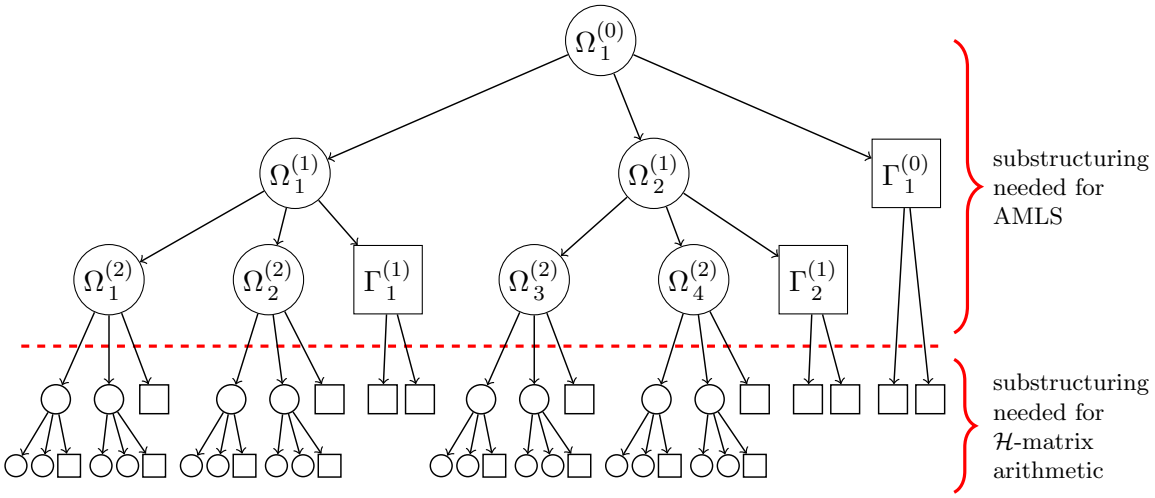


Figure 8.1.: Schematic example of the substructuring of the domain Ω applied in $\mathcal{H}$-AMLS: A two-level geometric nested dissection (necessary for AMLS, cf. Figure 5.3) is applied followed by an additional two-level geometric nested dissection of the subdomains and a one-level geometric bisection of the interfaces (necessary for $\mathcal{H}$-matrix approximation).

be constructed. The substructuring of the domain Ω, which is associated to the construction of $\mathcal{T}_I$, has already been discussed in Section 6.2. However, the described substructuring scheme is not optimal for the computation of (6.12), especially when in task (T2) a complete $\mathrm{LDL}^{\mathrm{T}}$-factorisation is performed. Instead of applying a recursive geometric bisection of the index sets associated to subdomains it is much more efficient regarding the computational costs of (6.12) to apply as well a geometric nested dissection, cf. [35]. To be more precise, let $t \subset I$ be a cluster associated to a subdomain, then t is geometrically subdivided into two so-called *domain-clusters* t_1 and t_2 which are separated by a so-called *interface-cluster* t_3 such that

$$t = \dot{\bigcup_{i\in\{1,2,3\}}} t_i \qquad \text{with} \qquad \Omega_{t_1} \cap \Omega_{t_2} = \emptyset, \tag{8.2}$$

where Ω_{t_i} is the support [see (6.1)] of the cluster t_i. The implementation of such a subdivision is typically based on a initial geometric bisection (cf. Algorithm 1) and followed by the construction of an explicit separator t_3, cf. [34, 35]. This substructuring scheme is applied recursively to the domain-cluster t_1 and t_2 until the size of the clusters is small enough. In contrast to this, interface-cluster are recursively subdivided by a geometric bisection. The resulting partitioning of the index set I is organised in the cluster tree $\mathcal{T}_I$. The construction of $\mathcal{T}_I$ is summarised in Algorithm 6 which has to be applied to the index set I and the associated geometric representatives $(\xi_i)_{i\in I}$ (the representatives ξ_i from (6.7) are needed for the geometric subdivision), and where the parameter $n_{\min}$ is controlling the minimal size of the clusters. In the numerical examples presented later in this work the parameter $n_{\min}$ has been set to 40 which is a standard value for $n_{\min}$ (cf. [50]) and which lead to a good computational performance. Note that in Algorithm 6 the subdivision of the interface-clusters is delayed every d-th step in order to avoid that interfaces are substructured faster in smaller parts than subdomains, cf. [35]. The domain substructuring of Ω associated to the construction of $\mathcal{T}_I$ is illustrated in Figure 8.1.

In the next step the block cluster tree $\mathcal{T}_{I\times I}$ associated to $\mathcal{T}_I$ has to be constructed. This is

done by applying Algorithm 2 to the product index set $I \times I$ using the parameters $n_{\min}$ and η, and the following adjusted admissibility condition

$$\text{block cluster } s \times t \text{ is admissible } :\Longleftrightarrow \begin{cases} \text{if condition (6.2) is fulfilled } \underline{\text{or}} \\ \text{if } s \text{ and } t \text{ are domain-clusters with } s \neq t \end{cases}. \tag{8.3}$$

The adjusted admissibility condition takes into account that for a domain-cluster t which is subdivided as in (8.2) typically the distance between Ω_{t_1} and Ω_{t_2} is very small in contrast to the diameters of Ω_{t_1} and Ω_{t_2}, and that correspondingly the block cluster $t_1 \times t_2$ is often not admissible according to condition (6.2). However, since the submatrices of K and M associated to the block cluster $t_1 \times t_2$ are zero and, most importantly, remain zero when (6.12) is computed (cf. [35]), it follows that block cluster $t_1 \times t_2$ should be considered as well as admissible since the corresponding submatrices can be represented as R(k)-matrices of rank zero. As mentioned in Section 6.1 the parameter $\eta > 0$ controlling the number of admissible subblocks $s \times t$ is typically set to $\eta = 1$ (see, e.g., [33]). However, in numerical tests better results have been obtained according to the computational time using larger η and correspondingly having larger but fewer admissible subblocks. In [40] better results have been obtained as well when even subblocks $s \times t$ with $s \neq t$ were accepted as admissible. In numerical tests $\eta := 50$ has been a good choice and this value has been used in the rest of the work.

Finally, the rows and columns of K and M are reordered according to the partitioning of I (which is described by the cluster tree $\mathcal{T}_I$), so that K and M are partitioned into blocks according to $\mathcal{L}(\mathcal{T}_{I\times I})$. Since for block cluster $s \times t \in \mathcal{L}(\mathcal{T}_{I\times I})$ fulfilling admissibility condition (8.3) the associated supports Ω_s and Ω_t are geometrically separated it follows that the submatrices $K_{|s\times t}$ and $M_{|s\times t}$ are equal to zero (cf. Lemma 3.6) and can be represented exactly by R(k)-matrices with rank zero. Hence, no approximation is necessary to represent K and M in the matrix format $\mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$. The (exact) $\mathcal{H}$-matrix representations of K and M are denoted by $K^{\mathcal{H}}$ and $M^{\mathcal{H}}$.

The modified substructuring scheme described in Algorithm 6 has two advantages: First, the computation of (6.12) becomes faster (see, e.g., [35]). Second, the matrices $K^{\mathcal{H}}$ and $M^{\mathcal{H}}$ are partitioned in such a way that in the $\mathcal{H}$-AMLS method the submatrices $K^{\mathcal{H}}_{ii}$ and $M^{\mathcal{H}}_{ii}$ associated to a subdomain are already partitioned according to a nested dissection. This becomes advantageous when in the recursive approach the subdomain eigenvalue problem $(K^{\mathcal{H}}_{ii}, M^{\mathcal{H}}_{ii})$ is solved recursively by $\mathcal{H}$-AMLS and no further partitioning of $K^{\mathcal{H}}_{ii}$ and $M^{\mathcal{H}}_{ii}$ is needed.

8.2. Task ($\mathbb{T}$2+$\mathbb{T}$3): Transformation of the Eigenvalue Problem

The computation of the transformed eigenvalue problem $(\widetilde{K}^{\mathcal{H}}, \widetilde{M}^{\mathcal{H}})$ is described in Algorithm 7. First a *nearly complete* LDL$^{\mathrm{T}}$-factorisation $K^{\mathcal{H}} \approx L^{\mathcal{H}}\widetilde{K}^{\mathcal{H}}(L^{\mathcal{H}})^T$ is computed using the corresponding HLIBpro routine (cf. [50]) with approximation accuracy ε where $\widetilde{K}^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$ is symmetric and $L^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$ is a lower triangular, unit diagonal matrix. Nearly complete LDL$^{\mathrm{T}}$-factorisation means that $K^{\mathcal{H}}$ is diagonalised only up to the block diagonal matrices which are associated to leaves of the block cluster tree $\mathcal{T}_{I\times I}$, i.e., the matrix $\widetilde{K}^{\mathcal{H}}$ is zero except for the dense block diagonal matrices $\widetilde{K}^{\mathcal{H}}_{|t\times t}$ which are associated to some $t \times t \in \mathcal{L}(\mathcal{T}_{I\times I})$. According to [50] the nearly complete LDL$^{\mathrm{T}}$-factorisation of an $\mathcal{H}$-matrix is more stable than the complete one. Because of Lemma 8.1 the transformation in task ($\mathbb{T}$2) can be modified in such a way.

Algorithm 7 Computation of the Transformed Eigenvalue Problem

procedure TRANSFORMEVP($K^{\mathcal{H}}$, $M^{\mathcal{H}}$, $\mathcal{T}_{I\times I}$, ε)
  ▷ *diagonalise the stiffness matrix $K^{\mathcal{H}}$ using the $\mathcal{H}$-arithmetic with approximation accuracy ε*
  initialise lower triangular matrix $L^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$;
  initialise symmetric matrix $\widetilde{K}^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$;
  compute (nearly) complete LDL$^{\mathrm{T}}$-decomposition $K^{\mathcal{H}} \approx L^{\mathcal{H}}\widetilde{K}^{\mathcal{H}}(L^{\mathcal{H}})^T$;
  ▷ *transform the mass matrix $M^{\mathcal{H}}$ using the $\mathcal{H}$-arithmetic with approximation accuracy ε*
  initialise matrix $A^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$;
  solve $L^{\mathcal{H}}A^{\mathcal{H}} = M^{\mathcal{H}}$ for $A^{\mathcal{H}}$;
  initialise symmetric matrix $\widetilde{M}^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$;
  solve $\widetilde{M}^{\mathcal{H}}(L^{\mathcal{H}})^T = A^{\mathcal{H}}$ for $\widetilde{M}^{\mathcal{H}}$; ▷ *compute only the lower triangular part of $\widetilde{M}^{\mathcal{H}}$*
  return $(\widetilde{K}^{\mathcal{H}}, \widetilde{M}^{\mathcal{H}}, L^{\mathcal{H}})$;
end procedure

After factorising $K^{\mathcal{H}}$ the transformed mass matrix $\widetilde{M}^{\mathcal{H}}$ is computed. This is done in two steps: First the matrix $A^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$ is computed by solving $L^{\mathcal{H}}A^{\mathcal{H}} = M^{\mathcal{H}}$ for $A^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$ where the lower triangular structure of $L^{\mathcal{H}}$ is exploited. The computation is performed using the corresponding HLIBpro routine with approximation accuracy ε. By solving $\widetilde{M}^{\mathcal{H}}(L^{\mathcal{H}})^T = A^{\mathcal{H}}$ for $\widetilde{M}^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$ finally the transformed mass matrix is obtained. In order to save computational time, explicitly only the lower block triangular part of the symmetric matrix $\widetilde{M}^{\mathcal{H}}$ is computed.

8.3. Auxiliary Data

For the implementation of the remaining $\mathcal{H}$-AMLS tasks the substructures of the domain Ω have to be identified (respectively, the corresponding subsets of I) which induce the m subproblems of the $\mathcal{H}$-AMLS method and the $m \times m$ block partitioning [cf., e.g., (5.21) or (5.29)] of the transformed matrices $\widetilde{K}^{\mathcal{H}}$ and $\widetilde{M}^{\mathcal{H}}$. For this purpose a truncated version of the cluster tree $\mathcal{T}_I$ is introduced which is called in the following *AMLS substructure tree* (or short *AMLS tree*) and which is referred to as $\mathcal{T}_{\mathrm{AMLS}}$. The AMLS tree is obtained by applying Algorithm 8 to $\mathcal{T}_I$ using the parameter $n_{\min}^{\mathrm{AMLS}} \in \mathbb{N}$ with $n_{\min}^{\mathrm{AMLS}} \geq n_{\min}$: In the first step of Algorithm 8 the tree $\mathcal{T}_{\mathrm{AMLS}}$ is initialised as a copy of $\mathcal{T}_I$. Thereafter $\mathcal{T}_{\mathrm{AMLS}}$ is traversed, starting from the root running recursively through the sons, until an interface-cluster is reached or a domain-cluster which is smaller or equal to $n_{\min}^{\mathrm{AMLS}}$. In both cases the sons of these clusters are truncated, i.e., for the corresponding $t \in \mathcal{T}_{\mathrm{AMLS}}$ the set of sons is set to $S(t) := \emptyset$. The parameter $n_{\min}^{\mathrm{AMLS}} \in \mathbb{N}$ controls the minimal size of the domain-clusters in $\mathcal{T}_{\mathrm{AMLS}}$, and it is set to the largest size of an eigenvalue problem which can still be solved easily by a direct solver. Note that all interface-cluster in the resulting AMLS tree $\mathcal{T}_{\mathrm{AMLS}}$ are leaves. In numerical tests $n_{\min}^{\mathrm{AMLS}} := 1000$ has been a good choice and this value is used in the rest of the work. The difference between the AMLS tree $\mathcal{T}_{\mathrm{AMLS}}$ and the cluster tree $\mathcal{T}_I$ can be illustrated in Figure 8.1: The upper part of the described domain substructuring in Figure 8.1 is associated with $\mathcal{T}_{\mathrm{AMLS}}$, whereas both the upper and lower part is associated with $\mathcal{T}_I$.

The leaves of the AMLS tree describe a disjoint partitioning of the index set I which results

Algorithm 8 Creation of the Substructure Tree

```
1: procedure CreateSubstructureTree( T_I, n_min^AMLS )
2:     initialise T_AMLS as a copy of T_I;
3:     U := {t} with t := root(T_AMLS);                ▷ t is the root of the tree T_AMLS
4:     while U ≠ ∅ do
5:         select arbitrary t ∈ U and set U := U \ {t};
6:         if t ∉ L(T_AMLS) then
7:             if t is a domain-cluster and #t > n_min^AMLS then
8:                 U := U ∪ S(t);          ▷ append sons of large domain-clusters to the cluster set U
9:             else
10:                S(t) := ∅;              ▷ delete sons of small domain-clusters and interface-clusters
11:            end if
12:        end if
13:    end while
14:    return T_AMLS;
15: end procedure
```

in a block partitioning of the product index set $I \times I$ of the form

$$I \times I \;=\; \dot{\bigcup_{\substack{s\in\mathcal{L}(\mathcal{T}_{\mathrm{AMLS}}),\\ t\in\mathcal{L}(\mathcal{T}_{\mathrm{AMLS}})}}} s \times t, \tag{8.4}$$

see Figure 8.2(b) for illustration. The partitioning of $I \times I$ in (8.4) corresponds to the $m \times m$ block partitioning of the transformed matrices $\widetilde{K}^{\mathcal{H}}$ and $\widetilde{M}^{\mathcal{H}}$ which has been described in Section 7.1 where $m = \#\mathcal{L}(\mathcal{T}_{\mathrm{AMLS}})$. More precisely, there exists a unique bijection

$$\psi : \{1, \ldots, m\} \to \mathcal{L}(\mathcal{T}_{\mathrm{AMLS}}) \quad \text{with} \quad i \mapsto \psi(i) \subset I \tag{8.5}$$

corresponding to the $m \times m$ block partitioning of $\widetilde{K}^{\mathcal{H}}$ and $\widetilde{M}^{\mathcal{H}}$ such that for $i, j = 1, \ldots, m$ it holds

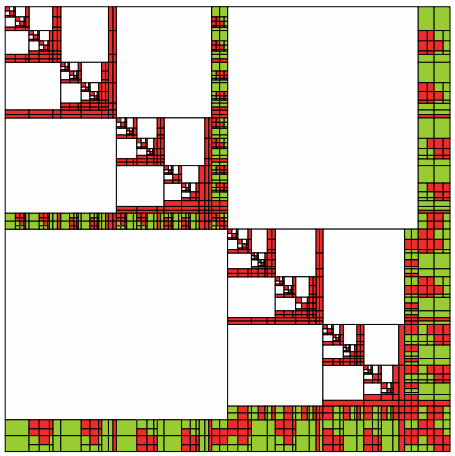
$$\widetilde{K}^{\mathcal{H}}_{ij} := (\widetilde{K}^{\mathcal{H}})_{|s\times t} \quad \text{and} \quad \widetilde{M}^{\mathcal{H}}_{ij} := (\widetilde{M}^{\mathcal{H}})_{|s\times t} \qquad \text{with} \quad s := \psi(i), t := \psi(j).$$

This bijection allows to assign to each subproblem index $i \in \{1, \ldots, m\}$ the corresponding index set $\psi(i) \in \mathcal{L}(\mathcal{T}_{\mathrm{AMLS}})$ and vice versa.

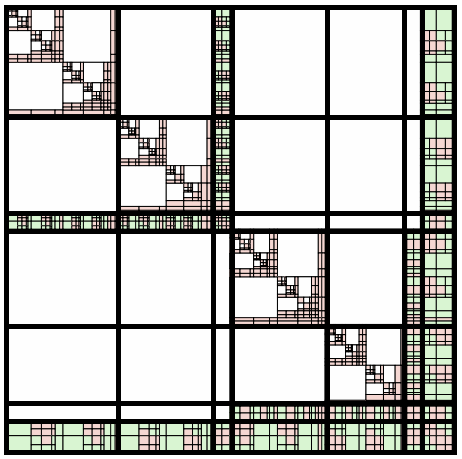
The block structure of the $\mathcal{H}$-matrices $\widetilde{K}^{\mathcal{H}}, \widetilde{M}^{\mathcal{H}} \in \mathcal{H}(\mathcal{T}_{I\times I}, \varepsilon)$, however, is described by the leaves of the block cluster tree $\mathcal{T}_{I\times I}$ via

$$I \times I \;=\; \dot{\bigcup_{s\times t\in\mathcal{L}(\mathcal{T}_{I\times I})}} s \times t,$$

and typically does not coincide with the $m\times m$ block structure described in (8.4), as it can be seen in Figure 8.2. To implement the remaining tasks of $\mathcal{H}$-AMLS the submatrices $\widetilde{K}^{\mathcal{H}}_{ij}$,$\widetilde{M}^{\mathcal{H}}_{ij}$ have to be extracted from $\widetilde{K}^{\mathcal{H}}$ and $\widetilde{M}^{\mathcal{H}}$. Note that the $\mathcal{H}$-matrices are implemented in a recursive block structure format which is guided by the block cluster tree $\mathcal{T}_{I\times I}$, cf. (6.9). Correspondingly, in order to obtain, for example, the submatrix $\widetilde{M}^{\mathcal{H}}_{ij}$ associated to the clusters $s := \psi(i)$ and $t := \psi(j)$, the block structure of $(\widetilde{M}^{\mathcal{H}})_{|I\times I}$ has to be recursively truncated to the block cluster $s\times t$, and when finally a leaf matrix $(\widetilde{M}^{\mathcal{H}})_{|u\times v}$ is reached with $u\times v \in \mathcal{L}(\mathcal{T}_{I\times I})$ and $u\times v \cap s\times t \neq \emptyset$



(a) The used $\mathcal{H}$-matrix format $\mathcal{H}(\mathcal{T}_{I\times I},\varepsilon)$ is independent of the constructed substructure tree $\mathcal{T}_{\mathrm{AMLS}}$.



(b) $m \times m$ block partitioning used in $\mathcal{H}$-AMLS to access the submatrices $\widetilde{K}_{ii}^{\mathcal{H}}$ and $\widetilde{M}_{ij}^{\mathcal{H}}$. In this example the substructure tree $\mathcal{T}_{\mathrm{AMLS}}$ has two levels resulting in $m = 7$ subproblems.

Figure 8.2.: Improved $\mathcal{H}$-matrix format $\mathcal{H}(\mathcal{T}_{I\times I},\varepsilon)$ used in $\mathcal{H}$-AMLS (cf. Section 8.1) for the finite element discretisation of an elliptic PDE eigenvalue problem on $\Omega = (0,1)^3$ with $\#I = 2744$ degrees of freedom. Red blocks represent full matrices, green blocks R(k)-matrices and white blocks submatrices equal to zero which don't cause computational costs during the computations performed in task (T2) and (T3).

the recursion is stopped and the corresponding full matrix or R(k)-matrix representation of $(\widetilde{M}^{\mathcal{H}})_{|u\times v}$ is truncated to the block cluster $\bar{u}\times\bar{v}$ where $\bar{u} := u \cap s$ and $\bar{v} := v \cap t$. Note that for an R(k)-matrix[2] $R = AB^T \in \mathbb{R}^{u\times v}$ with $A \in \mathbb{R}^{u\times k}$ and $B \in \mathbb{R}^{v\times k}$ the restriction of R to the block cluster $\bar{u}\times\bar{v}$ is given by the R(k)-matrix $\bar{R} = \bar{A}\bar{B}^T \in \mathbb{R}^{\bar{u}\times\bar{v}}$ with $\bar{A} \in \mathbb{R}^{\bar{u}\times k}$ and $B \in \mathbb{R}^{\bar{v}\times k}$.

8.4. Task (T4): Computation of the Partial Eigensolutions

In task (T4) the partial eigensolutions $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)$ of the subproblems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ have to be computed for $i = 1,\dots,m$. $\widetilde{\mathbf{D}}_i \in \mathbb{R}^{k_i\times k_i}$ is the diagonal matrix containing the $k_i \le N_i$ smallest eigenvalues of $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ and $\widetilde{\mathbf{S}}_i \in \mathbb{R}^{N_i\times k_i}$ the matrix containing column-wise the corresponding eigenvectors (cf. Section 7.1). In Remark 5.7 different mode selection strategies have been discussed. It is proposed to compute all eigenvectors of the discrete problems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ which still lead to reasonable approximations of the corresponding continuous eigenfunctions. More precisely, for subdomain eigenvalue problems it is proposed to compute only eigenvectors belonging to the smallest $k_i := \lceil C_{\mathrm{dom}}(N_i)^{\beta_{\mathrm{dom}}}\rceil$ eigenvalues are and for interface problems only the eigenvectors belonging to the smallest $k_i := \lceil C_{\mathrm{int}}(N_i)^{\beta_{\mathrm{int}}}\rceil$ eigenvalues. The concrete choice of the constants $C_{\mathrm{dom}}, C_{\mathrm{int}} > 0$ and $\beta_{\mathrm{dom}}, \beta_{\mathrm{int}} \in (0,1)$ is discussed in detail in Section 9.1.1 where numerical results are presented. This mode selection strategy depends solely on the size and the type of the subproblem.

[2]The notation $\mathbb{R}^{u\times v}$ indicates that R is a matrix of size $\#u \times \#v$ whose row indices are associated with the cluster $u \subset I$ and whose column indices are associated with $v \subset I$. Correspondingly, $A \in \mathbb{R}^{u\times k}$ and $B \in \mathbb{R}^{v\times k}$ are matrices of size $\#u \times k$ and $\#v \times k$ with $k \in \mathbb{N}$, where the row indices are associated with the clusters u and v.

Algorithm 9 Computation of the Partial Eigensolutions

procedure CompPartialEigensol($\widetilde{K}^{\mathcal{H}}$, $\widetilde{M}^{\mathcal{H}}$, C_{dom}, C_{int}, β_{dom}, β_{int})
  for $i = 1, \ldots, m$ **do**
    extract submatrices $\widetilde{K}_{ii}^{\mathcal{H}} := (\widetilde{K}^{\mathcal{H}})_{|t\times t}$ and $\widetilde{M}_{ii}^{\mathcal{H}} := (\widetilde{M}^{\mathcal{H}})_{|t\times t}$ with $t := \psi(i)$;
    if t is a domain-cluster **then**
      ▷ *compute partial eigensolution of a subproblem associated to a subdomain*
      initialise full matrices $\widetilde{\mathbf{D}}_i \in \mathbb{R}^{k_i\times k_i}$, $\widetilde{\mathbf{S}}_i \in \mathbb{R}^{N_i\times k_i}$ with $k_i := \min\{\lceil C_{\mathrm{dom}}(N_i)^{\beta_{\mathrm{dom}}}\rceil, N_i\}$;
      solve $\widetilde{K}_{ii}^{\mathcal{H}}\,\widetilde{\mathbf{S}}_i = \widetilde{M}_{ii}^{\mathcal{H}}\,\widetilde{\mathbf{S}}_i\,\widetilde{\mathbf{D}}_i$ using the dense LAPACK eigensolver `dsygvx`;
    else
      ▷ *compute partial eigensolution of a subproblem associated to an interface*
      initialise full matrices $\widetilde{\mathbf{D}}_i \in \mathbb{R}^{k_i\times k_i}$, $\widetilde{\mathbf{S}}_i \in \mathbb{R}^{N_i\times k_i}$ with $k_i := \min\{\lceil C_{\mathrm{int}}(N_i)^{\beta_{\mathrm{int}}}\rceil, N_i\}$;
      if $\#t \le n_{\min}^{\mathrm{AMLS}}$ **then**
        solve $\widetilde{K}_{ii}^{\mathcal{H}}\,\widetilde{\mathbf{S}}_i = \widetilde{M}_{ii}^{\mathcal{H}}\,\widetilde{\mathbf{S}}_i\,\widetilde{\mathbf{D}}_i$ using the dense LAPACK eigensolver `dsygvx`;
      else
        solve $\widetilde{K}_{ii}^{\mathcal{H}}\,\widetilde{\mathbf{S}}_i = \widetilde{M}_{ii}^{\mathcal{H}}\,\widetilde{\mathbf{S}}_i\,\widetilde{\mathbf{D}}_i$ using the eigensolver $\mathcal{H}$-SIL;
      end if
    end if
  end for
  return $(\mathbf{D}_i, \mathbf{S}_i)_{i=1}^m$;
end procedure

Remark 8.2 *Of course also other mode selection strategies can be implemented, for example, a strategy motivated by the results of Corollary 3.17 on the maximal size $\lambda_{\max}^{\mathrm{EF}}$ of well approximable eigenfunctions. This means, depending on whether the eigenvalue problem $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ is associated to a subdomain or an interface all eigenvectors of subproblem $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ are computed whose eigenvalues are smaller than the truncation bound ω_{dom} (for subdomains) and ω_{int} (for interfaces) where*

$$\omega_{\mathrm{dom}} := \tilde{C}_{\mathrm{dom}} N_h^{\tilde{\beta}_{\mathrm{dom}}} \qquad \text{and} \qquad \omega_{\mathrm{int}} := \tilde{C}_{\mathrm{int}} N_h^{\tilde{\beta}_{\mathrm{int}}}$$

with suitable constants $\tilde{C}_{\mathrm{dom}}, \tilde{C}_{\mathrm{int}}, \tilde{\beta}_{\mathrm{dom}}, \tilde{\beta}_{\mathrm{int}} > 0$ depending on the polynomial degree of the underlying finite element space V_h and the spatial dimension d. Using this mode selection strategy the number k_i of chosen eigenvectors is not known in advance which, however, is highly desirable when the recursive version of $\mathcal{H}$-AMLS is implemented (see Section 8.7 for details) and the size k_u has to be determined in order to check condensation condition (8.15).

The implementation of task (𝕋4) is described in Algorithm 9. Depending on the type of the subproblem a different eigensolver is used. If the eigenvalue problem $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ is associated to a subdomain (i.e., $t := \psi(i)$ is a domain-cluster) then the size of the problem is relatively small (according to Section 8.3 it holds $N_i = \#t \le n_{\min}^{\mathrm{AMLS}}$) and it can be handled easily by a direct eigensolver. For this purpose the dense eigensolver routine `dsygvx` of LAPACK [3] is used. This eigensolver is used as well when the problem $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ is associated to an interface of small size, i.e., when it holds $N_i \le n_{\min}^{\mathrm{AMLS}}$. However, if the interface problem is getting large the solution by the dense LAPACK solver becomes too expensive. In the classical AMLS method the matrices $\widetilde{K}_{ii}$ and $\widetilde{M}_{ii}$ associated to an interface are typically dense, however, in the $\mathcal{H}$-AMLS method the matrices $\widetilde{K}_{ii}^{\mathcal{H}}$ and $\widetilde{M}_{ii}^{\mathcal{H}}$ are represented in the $\mathcal{H}$-matrix format. This data-sparse $\mathcal{H}$-matrix structure can be exploited by an iterative eigensolver. In the following two eigensolvers are presented which have been implemented, and which can be used for the solution of large

interface eigenvalue problems.

Eigensolver $\mathcal{H}$-ARPACK

Let $A^{\mathcal{H}}$ and $B^{\mathcal{H}}$ be symmetric $\mathcal{H}$-matrices and $A^{\mathcal{H}}$ regular. We consider the eigenvalue problem

$$A^{\mathcal{H}} x \;=\; \lambda B^{\mathcal{H}} x \tag{8.6}$$

where the eigensolutions associated to the smallest eigenvalues are sought. To solve this problem the FORTRAN 77 library *ARPACK* [57] can be used which provides routines for the solution of large scale eigenvalue problems. The implementation of the corresponding eigensolver is as follows: To solve problem (8.6) the eigensolutions of the transformed problem

$$(A^{\mathcal{H}})^{-1}\, B^{\mathcal{H}}\, x \;=\; \frac{1}{\lambda}\, x \tag{8.7}$$

are computed which are associated to the largest eigenvalues $1/\lambda$. The solution of (8.7) is performed using the ARPACK routines `dsaupd` and `dseupd` that implement the so-called *Implicitly Restarted Lanczos Method* which is a combination of the Lanczos process with the implicitly shifted QR technique, see [57] for details. ARPACK provides a reverse communication interface (cf. [57]) that allows the usage of the fast $\mathcal{H}$-arithmetic for the involved matrix-vector multiplications. Note that the $\mathcal{H}$-matrix-vector multiplication is performed exactly. Furthermore, the matrix $(A^{\mathcal{H}})^{-1}B^{\mathcal{H}}$ in (8.7) is not evaluated explicitly, instead an approximative $\mathrm{LDL}^{\mathrm{T}}$-factorisation $A^{\mathcal{H}} \approx L_A^{\mathcal{H}} D (L_A^{\mathcal{H}})^T$ is computed by the fast $\mathcal{H}$-arithmetic which is used as an preconditioner (cf. [34, 35, 49]) for the iterative solution of the linear system involved in (8.7). The accuracy of the $\mathcal{H}$-arithmetic used for the computation of this factorisation is chosen in such a way that it holds $\|\mathrm{Id} - (L_A^{\mathcal{H}} D (L_A^{\mathcal{H}})^T)^{-1} A^{\mathcal{H}}\|_2 \leq 10^{-2}$, and correspondingly, the preconditioner $(L_A^{\mathcal{H}} D (L_A^{\mathcal{H}})^T)^{-1}$ guarantees in a linear iteration method a convergence with a convergence rate of at least 10^{-2}. The tolerance parameter `tol` used for the stopping criterion (cf. [57]) of the ARPACK routine `dsaupd` has been set to 10^{-8}. Altogether, this leads to an eigensolver which computes eigenpair approximations $(\tilde{\lambda}, \tilde{x})$ with a relative residual error $\|A^{\mathcal{H}}\tilde{x} - \tilde{\lambda} B^{\mathcal{H}}\tilde{x}\|_2 / \|A^{\mathcal{H}}\tilde{x}\|_2$ typically in the order of single machine precision. In the following this eigensolver is referred to as *$\mathcal{H}$-ARPACK*.

Unfortunately, the ARPACK library is not *thread-safe* (see Section 8.8), in particular, it is not possible to solve several eigenvalue problems concurrently by the eigensolver $\mathcal{H}$-ARPACK. Correspondingly, another thread-safe eigensolver has to be implemented in order to parallelise task (𝕋4) efficiently. But due to the high accuracy of the $\mathcal{H}$-ARPACK eigenpair approximations, this eigensolver is used as reference solver when in Chapter 9 numerical results are presented and the approximation quality of $\mathcal{H}$-AMLS is compared with these of a classical approach.[3] For the solution of large interface eigenvalue problems (in both the parallel and the sequential implementation of $\mathcal{H}$-AMLS), however, the following eigensolver is used:

Eigensolver $\mathcal{H}$-SIL

Consider the eigenvalue problem (8.6) with the symmetric matrices $A^{\mathcal{H}} := \widetilde{K}_{ii}^{\mathcal{H}}$ and $B^{\mathcal{H}} := \widetilde{M}_{ii}^{\mathcal{H}}$, where $A^{\mathcal{H}}$ is positive definite. In order to solve this problem the eigensolutions of the transformed

[3] $\mathcal{H}$-ARPACK is applied to the sparse problem (K, M) in the analogical way as to the problem (8.7). In order to compute a preconditioner for K the corresponding $\mathcal{H}$-matrix representation $K^{\mathcal{H}}$ is factorised.

Algorithm 10 Check if $\widehat{\mathbf{M}}_{ij}$ is a Zero Matrix

```
1: procedure SubmatrixIsZero( i, j, T_AMLS )
2:     get clusters s := ψ(i) and t := ψ(j);
3:     ▷ F(t) ∈ T_AMLS is defined as the father of t ∈ T_AMLS, i.e., it holds t ∈ S(F(t))
4:     if t is an interface-cluster then
5:         t := F(t);
6:     end if
7:     if s is an interface-cluster then
8:         s := F(s);
9:     end if
10:    ▷ if F(s) ∩ F(t) = ∅ then M̂_ij is a zero matrix, otherwise not
11:    return ( F(s) ∩ F(t) ≡ ∅ );
12: end procedure
```

problem

$$L_A^{-1} B^{\mathcal{H}} L_A^{-T}\, y \;=\; \frac{1}{\lambda}\, y \qquad \text{with} \quad A^{\mathcal{H}} = L_A L_A^T \quad \text{and} \quad y := L_A^T x \tag{8.8}$$

are computed which are associated to the largest eigenvalues $1/\lambda$, where $A^{\mathcal{H}} = L_A L_A^T$ in (8.8) is the exact Cholesky factorisation of $A^{\mathcal{H}}$. Note that $A^{\mathcal{H}} = \widetilde{K}_{ii}^{\mathcal{H}}$ is nearly diagonal (cf. Section 8.2) since in task ($\mathbb{T}2$) a nearly complete LDL$^{\mathrm{T}}$-decomposition of $K^{\mathcal{H}}$ is performed, and correspondingly it is quite inexpensive to compute the (numerically) exact Cholesky factorisation of $A^{\mathcal{H}}$. For the solution of problem (8.8) the Lanczos method (without shift) has been implemented where the matrix $L_A^{-1} B^{\mathcal{H}} L_A^{-T}$ is not evaluated explicitly, instead the vector $y_{\text{new}} := L_A^{-1} B^{\mathcal{H}} L_A^{-T} y$ is computed by solving the triangular system $L_A^T y' = y$ for y', computing $y'' := B^{\mathcal{H}} y'$ using the fast $\mathcal{H}$-arithmetic, and solving $L_A y_{\text{new}} = y''$ for y_{new}. Note that the matrix L_A is nearly diagonal, and correspondingly the computation of y' and y_{new} is inexpensive. In the following this eigensolver is referred to as *$\mathcal{H}$-SIL*.

For the solution of (8.8) only a basic version of the Lanczos method has been implemented (see, e.g., [8]) with a fixed number of maximal iterations. Benchmarks have shown that in some examples the accuracy of the eigenpair approximations computed by $\mathcal{H}$-SIL is slightly worse than the one of $\mathcal{H}$-ARPACK. However, in task ($\mathbb{T}4$) it is not necessary to compute (numerically) exact eigensolutions of the discrete problems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$. Instead the aim is to provide enough spectral information from each subproblem $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ in order to derive from the reduced problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$ sufficiently good eigenpair approximations $(\,\widehat{\boldsymbol{\lambda}}_j^{(\mathrm{rq})}, \widehat{\mathbf{y}}_j)_{j=1}^{n_{\mathrm{es}}}$ of the original problem (K, M) [cf. Remark 5.7]. Hence, it is sufficient to compute only approximative eigenpairs of $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$, and to compensate a possibly lower approximation quality of the computed partial eigensolution by slightly increasing the number of computed eigenpairs.

8.5. Task ($\mathbb{T}6$+$\mathbb{T}7$): Computation and Solution of the Reduced EVP

In task ($\mathbb{T}6$) the reduced matrices $\widehat{\mathbf{K}} := \mathbf{Z}^T \widetilde{K}^{\mathcal{H}} \mathbf{Z}$ and $\widehat{\mathbf{M}} := \mathbf{Z}^T \widetilde{M}^{\mathcal{H}} \mathbf{Z}$ have to be computed. The computation of these matrices is described in Algorithm 11 where it is noted that $\widehat{\mathbf{K}}, \widehat{\mathbf{M}} \in \mathbb{R}^{\bar{k} \times \bar{k}}$

Algorithm 11 Computation of the Reduced Matrices

procedure CompReducedMatrices($\widetilde{K}^{\mathcal{H}}$, $\widetilde{M}^{\mathcal{H}}$, $\mathcal{T}_{\text{AMLS}}$, $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=1}^m$)
  initialise full matrices $\widehat{\mathbf{K}}, \widehat{\mathbf{M}} \in \mathbb{R}^{\bar{k}\times\bar{k}}$ with matrix entries equal to zero and $\bar{k} = \sum_{i=1}^m k_i$;
  initialise $m \times m$ block structure $\widehat{\mathbf{K}} = (\widehat{\mathbf{K}}_{ij})_{i,j=1}^m$ and $\widehat{\mathbf{M}} = (\widehat{\mathbf{M}}_{ij})_{i,j=1}^m$;
  for $i = 1, \ldots, m$ **do**
    extract submatrices $\widetilde{K}_{ii}^{\mathcal{H}} := (\widetilde{K}^{\mathcal{H}})_{|t\times t}$ with $t := \psi(i)$;
    compute $\widehat{\mathbf{K}}_{ii} := \widetilde{\mathbf{S}}_i^T \widetilde{K}_{ii}^{\mathcal{H}} \widetilde{\mathbf{S}}_i \in \mathbb{R}^{k_i\times k_i}$ using the $\mathcal{H}$-arithmetic;
    for $j = 1, \ldots, i$ **do**
      extract submatrix $\widetilde{M}_{ij}^{\mathcal{H}} := (\widetilde{M}^{\mathcal{H}})_{|s\times t}$ with $s := \psi(i)$, $t := \psi(j)$;
      if $\neg$ SubmatrixIsZero(i, j, $\mathcal{T}_{\text{AMLS}}$) **then** ▷ *see Algorithm 10*
        compute $\widehat{\mathbf{M}}_{ij} := \widetilde{\mathbf{S}}_i^T \widetilde{M}_{ij}^{\mathcal{H}} \widetilde{\mathbf{S}}_j \in \mathbb{R}^{k_i\times k_j}$ using the $\mathcal{H}$-arithmetic;
      end if
    end for
  end for
  return $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$;
end procedure

possess the following $m \times m$ block structure

$$\widehat{\mathbf{K}} = \operatorname{diag}[\widehat{\mathbf{K}}_{11}, \ldots, \widehat{\mathbf{K}}_{mm}] \quad \text{with} \quad \widehat{\mathbf{K}}_{ii} = \widetilde{\mathbf{S}}_i^T \widetilde{K}_{ii}^{\mathcal{H}} \widetilde{\mathbf{S}}_i \in \mathbb{R}^{k_i\times k_i} \quad \text{for } i = 1, \ldots, m,$$
$$\widehat{\mathbf{M}} = (\widehat{\mathbf{M}}_{ij})_{i,j=1}^m \quad \text{with} \quad \widehat{\mathbf{M}}_{ij} = \widetilde{\mathbf{S}}_i^T \widetilde{M}_{ij}^{\mathcal{H}} \widetilde{\mathbf{S}}_j \in \mathbb{R}^{k_i\times k_j} \quad \text{for } i, j = 1, \ldots, m.$$

Because of the symmetry of the reduced matrices only the lower triangular part of $\widehat{\mathbf{M}}$ has to be computed. Furthermore, some $\widehat{\mathbf{M}}_{ij}$ are zero matrices (i.e., all entries of these matrices are equal to zero). Note that $\widehat{\mathbf{M}}_{ij}$ is a zero matrix if and only if $\widetilde{M}_{ij}^{\mathcal{H}}$ is a zero matrix, and that a large part of the block-sparsity structure of $M^{\mathcal{H}}$ is retained in $\widetilde{M}^{\mathcal{H}}$ [see, e.g., (5.23) and (5.30)]. To see if $\widetilde{M}_{ij}^{\mathcal{H}}$ and $\widehat{\mathbf{M}}_{ij}$ are zero matrices the substructure tree $\mathcal{T}_{\text{AMLS}}$ has to be observed. For a cluster $t \in \mathcal{T}_{\text{AMLS}}$ with $t \neq \text{root}(\mathcal{T}_{\text{AMLS}})$ we define the unique cluster $F(t) := u \in \mathcal{T}_{\text{AMLS}}$ fulfilling $t \in S(u)$ as the *father* of t. For the clusters $s = \psi(i)$ and $t = \psi(j)$ associated to $\widehat{\mathbf{M}}_{ij}$ it can be observed that $\widehat{\mathbf{M}}_{ij}$ is a zero matrix (for illustration see upper part of Figure 8.1):

- if s and t are domain-cluster with $i \neq j$,
- if s is a domain-cluster and t an interface-cluster with $s \cap F(t) = \emptyset$,
- if s is an interface-cluster and t a subdomain-cluster with $F(s) \cap t = \emptyset$,
- if s and t are both interface-clusters with $F(s) \cap F(t) = \emptyset$.

This result is summarised in Algorithm 10. In the case that $\widehat{\mathbf{M}}_{ij}$ has to be computed (the same holds for $\widehat{\mathbf{K}}_{ii}$) then in the first step the matrix $A := \widetilde{M}_{ij}^{\mathcal{H}} \widetilde{\mathbf{S}}_j \in \mathbb{R}^{N_i\times k_j}$ is computed (numerically exact) where the data-sparse structure of the $\mathcal{H}$-matrix $\widetilde{M}_{ij}^{\mathcal{H}}$ is exploited, and in the second step $\widetilde{M}_{ij}^{\mathcal{H}} = \widetilde{\mathbf{S}}_i^T A$ is computed as the product of two matrices in full matrix representation. Note that $\widehat{\mathbf{K}}_{ii} = \widetilde{\mathbf{D}}_i \in \mathbb{R}^{k_i\times k_i}$ and $\widehat{\mathbf{M}}_{ii} = \mathrm{Id} \in \mathbb{R}^{k_i\times k_i}$ when a (numerically) exact partial eigensolution $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)$ has been computed with normalisation $\widetilde{\mathbf{S}}_i^T \widetilde{M}_{ii}^{\mathcal{H}} \widetilde{\mathbf{S}}_i = \mathrm{Id}$, and that in this case the computation of $\widehat{\mathbf{K}}_{ii}$ and $\widehat{\mathbf{M}}_{ii}$ has not to be performed. However, in benchmarks could be observed that the used eigensolvers `dsygvx` and $\mathcal{H}$-SIL computed in some cases eigensolutions where the

corresponding relative residual error[4] has been larger than machine precision. Hence, to be on the safe side the matrices $\widehat{\mathbf{K}}_{ii}$ and $\widehat{\mathbf{M}}_{ii}$ are computed in Algorithm 11 as well.

In the next step of $\mathcal{H}$-AMLS, in task ($\mathbb{T}7$), the partial eigensolution

$$\widehat{\mathbf{K}}\,\widehat{\mathbf{S}} \;=\; \widehat{\mathbf{M}}\,\widehat{\mathbf{S}}\,\widehat{\mathbf{D}} \qquad \text{with} \quad \widehat{\mathbf{S}}^T\,\widehat{\mathbf{M}}\,\widehat{\mathbf{S}} \;=\; \mathrm{Id} \tag{8.9}$$

of the $\mathcal{H}$-reduced eigenvalue problem is computed, where the diagonal matrix $\widehat{\mathbf{D}} \in \mathbb{R}^{n_{\mathrm{es}} \times n_{\mathrm{es}}}$ contains the eigenvalues $(\widehat{\boldsymbol{\lambda}}_j)_{j=1}^{n_{\mathrm{es}}}$ and $\widehat{\mathbf{S}} \in \mathbb{R}^{\bar{k} \times n_{\mathrm{es}}}$ column-wise the associated eigenvectors $(\widehat{\mathbf{x}}_j)_{j=1}^{n_{\mathrm{es}}}$. Although the reduced problem is partially structured[5] the computation is performed by the dense LAPACK eigensolver `dsygvx`[6]. Since it is ultimately aimed that the size $\bar{k}$ of the $\mathcal{H}$-reduced eigenvalue problem is bounded by $\mathcal{O}(n_{\mathrm{es}})$ when $\mathcal{H}$-AMLS is applied recursively (cf. Section 8.7), the problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$ will be small enough to be solved easily by the dense solver `dsygvx`.

8.6. Task ($\mathbb{T}8$+$\mathbb{T}9$+$\mathbb{T}\mathrm{SI}$): Transformation of the Eigensolutions

In task ($\mathbb{T}8$) of $\mathcal{H}$-AMLS the computed eigenvectors $(\widehat{\mathbf{x}}_j)_{j=1}^{n_{\mathrm{es}}}$ of the $\mathcal{H}$-reduced problem have to be transformed via

$$\widehat{\mathbf{y}}_j := (L^{\mathcal{H}})^{-T}\,\mathbf{Z}\,\widehat{\mathbf{x}}_j \qquad \text{with} \quad \mathbf{Z} := \mathrm{diag}[\widetilde{\mathbf{S}}_1, \ldots, \widetilde{\mathbf{S}}_m] \in \mathbb{R}^{N \times \bar{k}}, \tag{8.10}$$

The computation of $\widehat{\mathbf{y}}_j$ in (8.10) can be performed vector-wise for $j = 1, \ldots, n_{\mathrm{es}}$. However, in terms of runtime it is typically much more efficient to represent $(\widehat{\mathbf{x}}_j)_{j=1}^{n_{\mathrm{es}}}$ by the full matrix $\widehat{\mathbf{S}}$ from (8.9) and to perform the computation in (8.10) matrix-wise, since for most computer systems with cache memories (cf. Section 8.8) data locality can be exploited much better in the matrix-wise approach than in the vector-wise approach.

To compute (8.10) matrix-wise in the first step the matrix $\widetilde{\mathbf{S}} := \mathbf{Z}\,\widehat{\mathbf{S}}$ is computed via

$$\underbrace{\begin{bmatrix} \widetilde{\mathbf{S}}^{(1)} \\ \vdots \\ \widetilde{\mathbf{S}}^{(m)} \end{bmatrix}}_{=\,\widetilde{\mathbf{S}} \in \mathbb{R}^{N \times n_{\mathrm{es}}}} = \underbrace{\begin{bmatrix} \widetilde{\mathbf{S}}_1 & & \\ & \ddots & \\ & & \widetilde{\mathbf{S}}_m \end{bmatrix}}_{=\,\mathbf{Z} \in \mathbb{R}^{N \times \bar{k}}} \underbrace{\begin{bmatrix} \widehat{\mathbf{S}}^{(1)} \\ \vdots \\ \widehat{\mathbf{S}}^{(m)} \end{bmatrix}}_{=\,\widehat{\mathbf{S}} \in \mathbb{R}^{\bar{k} \times n_{\mathrm{es}}}} \quad \text{with} \quad \widetilde{\mathbf{S}}^{(i)} := \widetilde{\mathbf{S}}_i\,\widehat{\mathbf{S}}^{(i)} \quad \text{for } i = 1, \ldots, m \tag{8.11}$$

where the block diagonal structure of $\mathbf{Z}$ is exploited, i.e., for $\widetilde{\mathbf{S}}_i \in \mathbb{R}^{N_i \times k_i}$ the matrices $\widetilde{\mathbf{S}}$ and $\widehat{\mathbf{S}}$ are decomposed correspondingly into blocks with $\widetilde{\mathbf{S}}^{(i)} \in \mathbb{R}^{N_i \times n_{\mathrm{es}}}$, $\widehat{\mathbf{S}}^{(i)} \in \mathbb{R}^{k_i \times n_{\mathrm{es}}}$. Note that $(\widehat{\mathbf{D}}, \widetilde{\mathbf{S}})$ is an approximative eigensolution of eigenvalue problem $(\widetilde{K}^{\mathcal{H}}, \widetilde{M}^{\mathcal{H}})$ with $\widehat{\mathbf{D}}$ from (8.9) and where $\widetilde{\mathbf{S}}^T \widetilde{M}^{\mathcal{H}} \widetilde{\mathbf{S}} = \mathrm{Id}$. In the next step the transformed eigenvector matrix $\mathbf{S}_{\mathrm{temp}} := (L^{\mathcal{H}})^{-T}\widetilde{\mathbf{S}}$ is computed by solving the triangular system $(L^{\mathcal{H}})^T \mathbf{S}_{\mathrm{temp}} = \widetilde{\mathbf{S}}$ for $\mathbf{S}_{\mathrm{temp}}$ via the fast $\mathcal{H}$-arithmetic. In particular, the computation of $\mathbf{S}_{\mathrm{temp}} = (L^{\mathcal{H}})^{-T}\widetilde{S}$ is performed numerically exact. Finally, the

[4] We define the *relative residual error* of an eigenpair approximation (λ, x) of problem (K, M) by $\|Kx - \lambda Mx\|_2 / \|Kx\|_2$.

[5] $\widehat{\mathbf{K}}$ is a block diagonal matrix and $\widehat{\mathbf{M}}$ has a block-sparsity structure similar to the structure of $\widetilde{M}^{\mathcal{H}}$ (cf. Figure 8.2). See Section 8.5 for details.

[6] Since eigensolver `dsygvx` (cf. [3]) requests only the lower triangular matrix part of the symmetric problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$ it is sufficient in Algorithm 11 to compute and store only the lower block triangular part of $\widehat{\mathbf{M}}$.

Algorithm 12 Transformation and Improvement of Eigensolutions

procedure TRANSFORMEIGENSOL(K, M, $L^{\mathcal{H}}$, $\widetilde{K}^{\mathcal{H}}$, $(\widetilde{\mathbf{S}}_i)_{i=1}^m$, $\widehat{\mathbf{S}}$, apply_task_𝕋SI)
  ▷ *compute the transformed eigenvectors* $\widetilde{\mathbf{S}} := \mathbf{Z}\,\widehat{\mathbf{S}}$
  initialise full matrix $\widetilde{\mathbf{S}} \in \mathbb{R}^{N \times n_{\text{es}}}$;
  partition $\widetilde{\mathbf{S}}$ and $\widehat{\mathbf{S}}$ in m block rows as it is done in (8.11);
  for $i = 1, \ldots, m$ **do**
    compute $\widetilde{\mathbf{S}}^{(i)} = \mathbf{S}_i\,\widehat{\mathbf{S}}^{(i)}$;
  end for
  ▷ *compute* $\mathbf{S}_{\text{temp}} := (L^{\mathcal{H}})^{-T}\widetilde{\mathbf{S}} \in \mathbb{R}^{N \times n_{\text{es}}}$
  solve the triangular system $(L^{\mathcal{H}})^T\mathbf{S}_{\text{temp}} = \widetilde{\mathbf{S}}$ for $\mathbf{S}_{\text{temp}}$ using fast (and exact) $\mathcal{H}$-arithmetic;
  ▷ *restore the original index ordering of the approximated eigenvectors*
  apply inverse of the in task (𝕋1) performed index permutation π to column vectors of $\mathbf{S}_{\text{temp}}$;
  set $\mathbf{S} := \mathbf{S}_{\text{temp}}$;

  if apply_task_𝕋SI **then**
    ▷ *improve eigenvalue and eigenvector approximation by task (𝕋SI)*
    $(\mathbf{D}, \mathbf{S})$:= IMPROVEEIGENSOLUTION(K, M, $L^{\mathcal{H}}$, $\widetilde{K}^{\mathcal{H}}$, $\mathbf{S}$); ▷ *see Algorithm 4*
  else
    ▷ *compute Rayleigh quotients exploiting the sparse structure of* K, M
    initialise full matrix $\mathbf{D} \in \mathbb{R}^{n_{\text{es}} \times n_{\text{es}}}$ with entries equal to zero;
    for $j = 1, \ldots, n_{\text{es}}$ **do**
      $\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})} := \widehat{\mathbf{y}}_j^T K\,\widehat{\mathbf{y}}_j / \widehat{\mathbf{y}}_j^T M\,\widehat{\mathbf{y}}_j$ where $\widehat{\mathbf{y}}_j$ denotes the j-th column of $\mathbf{S}$;
      set the j-th diagonal entry of $\mathbf{D}$ equal to $\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}$;
    end for
  end if

  return $(\mathbf{D}, \mathbf{S})$;
end procedure

matrix $\mathbf{S}$ is obtained by applying the inverse of the in task (𝕋1) performed index permutation π (cf. Section 7.1) to the column vectors of the matrix $\mathbf{S}_{\text{temp}}$, and hence the permutated matrix $\mathbf{S} := \mathbf{S}_{\text{temp}}$ is containing column-wise the eigenvector approximations $\widehat{\mathbf{y}}_j$ of the initial eigenvalue problem (K, M) with the original index ordering.

Benchmarks have shown that the matrix-wise approach for the computation of (8.10) is significantly faster than the vector-wise approach, whereas the vector-wise approach scaled better on shared memory systems (see Section 8.8) with the number of cores than the matrix-wise approach. Nonetheless, also with many cores (benchmarks have been performed with up to 32 cores) the parallel implementation of the matrix-wise approach (cf. Section 8.8.6) still has been significantly faster than the parallel implementation of the vector-wise approach.

Depending on the chosen parameter setting in the next step of Algorithm 12 the Rayleigh quotients $\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}$ are computed (which involves matrix-vector multiplications with the sparse matrices K and M), or instead the in Section 7.4 described improvement task (𝕋SI) is applied. Finally, Algorithm 12 returns the eigenpair approximations $(\mathbf{D}, \mathbf{S})$ stated in (8.1) where depending on whether task (𝕋9) or task (𝕋SI) has been applied the diagonal matrix $\mathbf{D} \in \mathbb{R}^{n_{\text{es}} \times n_{\text{es}}}$ contains the eigenvalue approximations $\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}$ or $\widehat{\boldsymbol{\lambda}}_j^{(\text{SI})}$, and the matrix $\mathbf{S} \in \mathbb{R}^{N \times n_{\text{es}}}$ column-wise the eigenvector approximations $\widehat{\mathbf{y}}_j$ or $\widehat{\mathbf{y}}_j^{(\text{SI})}$. The approximation quality of the computed eigen-

pair approximations has been discussed in Section 7.3, and it depends on the chosen accuracy ε of the $\mathcal{H}$-arithmetic used in task ($\mathbb{T}2$) and ($\mathbb{T}3$), and the chosen mode selection strategy in task ($\mathbb{T}4$). How the accuracy and the mode selection strategy should be chosen is discussed in Chapter 9 where numerical results are presented.

8.7. Implementation of the recursive $\mathcal{H}$-AMLS method

In the previous sections the implementation of non-recursive $\mathcal{H}$-AMLS version has been described. According to Section 8.3 the domain Ω (respectively the index set I) has been substructured in so many levels until eigenvalue problems $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ associated to subdomains are getting small enough (according to Alg. 8 it holds $N_i \leq n_{\min}^{\mathrm{AMLS}}$) to be solved easily by a direct solver. However, the downside of this is that the number of subproblems is getting large when N increases: Applying the domain substructuring described in Section 8.1, Algorithm 8 leads to an AMLS substructure tree with m subproblems where $m = m(N)$ is typically of the order

$$m(N) \;\in\; \mathcal{O}\Big(\tfrac{N}{n_{\min}^{\mathrm{AMLS}}}\Big) \;=\; \mathcal{O}(N) \qquad \text{as } N \to \infty. \tag{8.12}$$

But as the number of subproblem increases with N also the size $\bar{k} = \sum_{i=1}^{m(N)} k_i$ of the reduced eigenvalue problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$ increases. Although the reduced problem is partially structured[7], eventually the total complexity of $\mathcal{H}$-AMLS is dominated by the solution of the reduced eigenvalue problem. Beside that also the computation of the reduced matrix $\widehat{\mathbf{M}}$ is getting expensive when $\bar{k}$ is getting large. To resolve this problem the $\mathcal{H}$-AMLS method is applied recursively (cf. Section 5.2.3 and Section 7.1) which enables both: small subdomain problems that are easy to solve, and a reduced problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$ whose size $\bar{k}$ is bounded by the order of $\mathcal{O}(n_{\mathrm{es}})$. To apply the recursive version of $\mathcal{H}$-AMLS the so-called *condensation process* has to be performed which is applied right after task ($\mathbb{T}4$) in Algorithm 5 (in non-recursive $\mathcal{H}$-AMLS this process is omitted). In the condensation process, which is described in the following, the spectral information of many small subproblems is condensed using $\mathcal{H}$-AMLS into the spectral information of few *superordinated* subproblems of larger size.

8.7.1. Applying $\mathcal{H}$-AMLS to a Superordinated Subproblem

Let $u \in \mathcal{T}_{\mathrm{AMLS}}$ be a cluster with $u \notin \mathcal{L}(\mathcal{T}_{\mathrm{AMLS}})$. Then there exist unique subproblem indices $\mathrm{i\_first}(u), \mathrm{i\_last}(u) \in \{1, \ldots, m\}$ with $\mathrm{i\_first}(u) < \mathrm{i\_last}(u)$ such that

$$u \;=\; \dot{\bigcup_{\mathrm{i\_first}(u) \leq i \leq \mathrm{i\_last}(u)}} \psi(i) \tag{8.13}$$

In the non-recursive version of the $\mathcal{H}$-AMLS method the spectral information of the subproblems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$ has been used, together with the spectral information of the other subproblems, to form in task ($\mathbb{T}6$) the reduced eigenvalue problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$. The spectral information of $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$ is approximatively described by the partial eigensolutions $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$,

[7] $\widehat{\mathbf{K}}$ is a block diagonal matrix and $\widehat{\mathbf{M}}$ has a block-sparsity structure similar to the structure of $\widetilde{M}^{\mathcal{H}}$ (cf. Figure 8.2). See Section 8.5 for details.

Algorithm 13 Apply $\mathcal{H}$-AMLS to the subproblem associated to the cluster $u \in \mathcal{T}_{\text{AMLS}}$

procedure ApplyHAMLSSub($\widetilde{K}^{\mathcal{H}}$, $\widetilde{M}^{\mathcal{H}}$, $\mathcal{T}_{\text{AMLS}}$, $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=1}^m$, u, $C_{\text{dom}}^{\approx}$, β_{dom})
  ▷ *comp. reduced matrices* $\widehat{\mathbf{K}}_u, \widehat{\mathbf{M}}_u \in \mathbb{R}^{\bar{k}_u \times \bar{k}_u}$ *associated to the cluster* u → *see Algorithm 14*
  $(\widehat{\mathbf{K}}_u, \widehat{\mathbf{M}}_u)$ = CompReducedMatricesSub($\widetilde{K}^{\mathcal{H}}$, $\widetilde{M}^{\mathcal{H}}$, $\mathcal{T}_{\text{AMLS}}$, $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=1}^m$, u);

  ▷ *determine number of needed eigenvectors*
  $k_u := \min\{ \lceil C_{\text{dom}}^{\approx}(N_u)^{\beta_{\text{dom}}} \rceil, N_u \}$ with $N_u := \sum_{\text{i\_first}(u)}^{\text{i\_last}(u)} N_i$;

  ▷ *compute the partial eigensolution of the reduced eigenvalue problem*
  initialise full matrices $\widehat{\mathbf{D}}_u \in \mathbb{R}^{k_u \times k_u}$ and $\widehat{\mathbf{S}}_u \in \mathbb{R}^{\bar{k}_u \times k_u}$;
  solve $\widehat{\mathbf{K}}_u \, \widehat{\mathbf{S}}_u = \widehat{\mathbf{M}}_u \, \widehat{\mathbf{S}}_u \, \widehat{\mathbf{D}}_u$ using the dense LAPACK eigensolver `dsygvx`;

  ▷ *transform the eigenvectors of the reduced eigenvalue problem* → *see Algorithm 15*
  $\widetilde{\mathbf{S}}_u$:=TransformEigenvectorsSub($(\widetilde{\mathbf{S}}_i)_{i=1}^m$, $\widehat{\mathbf{S}}_u$, u);

  ▷ *return (approximative) partial eigensolution* $(\widetilde{\mathbf{D}}_u, \widetilde{\mathbf{S}}_u)$ *of* $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ *where* $\widetilde{\mathbf{D}}_u := \widehat{\mathbf{D}}_u$
  return $(\widetilde{\mathbf{D}}_u, \widetilde{\mathbf{S}}_u)$;
end procedure

and in total $\bar{k}_u := \sum_{i=\text{i\_first}(u)}^{\text{i\_last}(u)} k_i$ eigenvectors are selected from these subproblems for the formation of the reduced problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$. But instead, it is also possible to consider in the further proceeding of $\mathcal{H}$-AMLS the *superordinated* eigenvalue problem $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ associated to the cluster u where

$$\widetilde{K}_u^{\mathcal{H}} := (\widetilde{K}^{\mathcal{H}})_{|u \times u} \in \mathbb{R}^{N_u \times N_u}, \quad \widetilde{M}_u^{\mathcal{H}} := (\widetilde{M}^{\mathcal{H}})_{|u \times u} \in \mathbb{R}^{N_u \times N_u} \text{ and } N_u := \sum_{i=\text{i\_first}(u)}^{\text{i\_last}(u)} N_i.$$

Instead of using the spectral information of the subproblems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})_{i=\text{i\_first}(u)}^{\text{i\_last}(u)}$ the spectral information of the superordinated problem can be used to form in task (𝕋6) the reduced problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$. For this purpose a partial eigensolution $(\widetilde{\mathbf{D}}_u, \widetilde{\mathbf{S}}_u)$ of $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ has to be computed. This eigensolution can be computed very efficiently by the $\mathcal{H}$-AMLS method, especially when N_u is large. Here the question arises how many eigensolutions of $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ have to be computed. According to the mode selection strategy described in Section 8.4 only those eigenvectors are computed which still lead to reasonable approximations of the corresponding continuous eigenfunctions. Since $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ is an eigenvalue problem associated to a subdomain (note that u is always a domain-cluster) these are the eigenvectors associated to the smallest $\lceil C_{\text{dom}}(N_u)^{\beta_{\text{dom}}} \rceil \in \mathbb{N}$ eigenvalues (cf. Section 8.4). However, the $\mathcal{H}$-AMLS method computes in general only approximative eigenvectors of a discrete problem and not (numerically) exact ones such as, for example, the dense LAPACK solver `dsygvx` does in task (𝕋4) in the most cases. Correspondingly, it is reasonable to compute slightly more eigensolutions for $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ when $\mathcal{H}$-AMLS is used, in order to compensate a possibly lower approximation quality and to provide enough spectral information of $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ for the formation of the reduced problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$ in task (𝕋6). This means, when $\mathcal{H}$-AMLS is used for the solution of $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ then it is suggested to compute the eigenvectors associated to the smallest k_u eigenvalues where

$$k_u := \lceil C_{\text{dom}}^{\approx}(N_u)^{\beta_{\text{dom}}} \rceil \in \mathbb{N} \quad \text{with some constant } C_{\text{dom}}^{\approx} \geq C_{\text{dom}}. \tag{8.14}$$

Algorithm 14 Computation of the reduced matrices associated to cluster $u \in \mathcal{T}_{\text{AMLS}}$

procedure CompReducedMatricesSub($\widetilde{K}^{\mathcal{H}}$, $\widetilde{M}^{\mathcal{H}}$, $\mathcal{T}_{\text{AMLS}}$, $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=1}^m$, u)
  compute $\bar{k}_u := \sum_{i=\text{i\_first}(u)}^{\text{i\_last}(u)} k_i$ and $m_u := \text{i\_last}(u) - \text{i\_first}(u) + 1$;
  initialise full matrices $\widehat{\mathbf{K}}_u, \widehat{\mathbf{M}}_u \in \mathbb{R}^{\bar{k}_u \times \bar{k}_u}$ with matrix entries equal to zero;
  initialise $m_u \times m_u$ block structure $\widehat{\mathbf{K}}_u = (\widehat{\mathbf{K}}_{ij})_{i,j=\text{i\_first}(u)}^{\text{i\_last}(u)}$ and $\widehat{\mathbf{M}}_u = (\widehat{\mathbf{M}}_{ij})_{i,j=\text{i\_first}(u)}^{\text{i\_last}(u)}$;
  for $i = \text{i\_first}(u), \ldots, \text{i\_last}(u)$ **do**
    extract submatrices $\widetilde{K}_{ii}^{\mathcal{H}} := (\widetilde{K}^{\mathcal{H}})_{|t\times t}$ with $t := \psi(i)$;
    compute $\widehat{\mathbf{K}}_{ii} := \widetilde{\mathbf{S}}_i^T \widetilde{K}_{ii}^{\mathcal{H}} \widetilde{\mathbf{S}}_i \in \mathbb{R}^{k_i \times k_i}$ using the $\mathcal{H}$-arithmetic;
    for $j = 1, \ldots, \text{i\_last}(u)$ **do**
      extract submatrix $\widetilde{M}_{ij}^{\mathcal{H}} := (\widetilde{M}^{\mathcal{H}})_{|s\times t}$ with $s := \psi(i)$, $t := \psi(j)$;
      if $\neg$ SubmatrixIsZero(i, j, $\mathcal{T}_{\text{AMLS}}$) **then** ▷ *see Algorithm 10*
        compute $\widehat{\mathbf{M}}_{ij} := \widetilde{\mathbf{S}}_i^T \widetilde{M}_{ij}^{\mathcal{H}} \widetilde{\mathbf{S}}_j \in \mathbb{R}^{k_i \times k_j}$ using the $\mathcal{H}$-arithmetic;
      end if
    end for
  end for
  return $(\widehat{\mathbf{K}}_u, \widehat{\mathbf{M}}_u)$;
end procedure

Algorithm 15 Transforming eigenvectors of the reduced EVP associated to cluster $u \in \mathcal{T}_{\text{AMLS}}$

procedure TransformEigenvectorsSub($(\widetilde{\mathbf{S}}_i)_{i=1}^m$, $\widehat{\mathbf{S}}_u$, u)
  ▷ *compute* $\widetilde{\mathbf{S}}_u := \mathbf{Z}_u \widehat{\mathbf{S}}_u$ *where* $\mathbf{Z}_u := \operatorname{diag}\big[\widetilde{\mathbf{S}}_{\text{i\_first}(u)}, \ldots, \widetilde{\mathbf{S}}_{\text{i\_last}(u)}\big] \in \mathbb{R}^{N_u \times \bar{k}_u}$
  initialise full matrix $\widetilde{\mathbf{S}}_u \in \mathbb{R}^{N_u \times k_u}$ with $N_u := \sum_{\text{i\_first}(u)}^{\text{i\_last}(u)} N_i$;
  partition $\widetilde{\mathbf{S}}_u$ and $\widehat{\mathbf{S}}_u$ in m_u block rows in a similar way as it is done in (8.11);
  for $i = \text{i\_first}(u), \ldots, \text{i\_last}(u)$ **do**
    compute $\widetilde{\mathbf{S}}_u^{(i)} = \widetilde{\mathbf{S}}_i \widehat{\mathbf{S}}_u^{(i)}$ where $\widetilde{\mathbf{S}}_u^{(i)} \in \mathbb{R}^{N_i \times k_u}$, $\widetilde{\mathbf{S}}_i \in \mathbb{R}^{N_i \times k_i}$ and $\widehat{\mathbf{S}}_u^{(i)} \in \mathbb{R}^{k_i \times k_u}$;
  end for
  return $\mathbf{S}_u$;
end procedure

The implementation of $\mathcal{H}$-AMLS applied to eigenvalue problem $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ is described in Algorithm 13: Tasks ($\mathbb{T}1$)–($\mathbb{T}4$) of $\mathcal{H}$-AMLS are omitted since $\widetilde{K}_u^{\mathcal{H}}$ has already block diagonal structure and the partial eigensolutions of $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})_{i=\text{i\_first}(u)}^{\text{i\_last}(u)}$ have been computed as well. The first task to be performed is the computation of the reduced matrices

$$\widehat{\mathbf{K}}_u := \mathbf{Z}_u^T \widetilde{K}_u^{\mathcal{H}} \mathbf{Z}_u \in \mathbb{R}^{\bar{k}_u \times \bar{k}_u} \quad \text{and} \quad \widehat{\mathbf{M}}_u := \mathbf{Z}_u^T \widetilde{K}_u^{\mathcal{H}} \mathbf{Z}_u \in \mathbb{R}^{\bar{k}_u \times \bar{k}_u}$$

with $\mathbf{Z}_u := \operatorname{diag}\big[\widetilde{\mathbf{S}}_{\text{i\_first}(u)}, \ldots, \widetilde{\mathbf{S}}_{\text{i\_last}(u)}\big] \in \mathbb{R}^{N_u \times \bar{k}_u}$, which posses the $m_u \times m_u$ block structure

$$\widehat{\mathbf{K}}_u = (\widehat{\mathbf{K}}_{ij})_{i,j=\text{i\_first}(u)}^{\text{i\_last}(u)} \quad \text{and} \quad \widehat{\mathbf{M}}_u = (\widehat{\mathbf{M}}_{ij})_{i,j=\text{i\_first}(u)}^{\text{i\_last}(u)}$$

where $m_u := \text{i\_last}(u) - \text{i\_first}(u)$. The computation of the reduced matrices is described in Algorithm 14 and is done in the analogical way as the computation of the global reduced matrices $\widehat{\mathbf{K}}$ and $\widehat{\mathbf{M}}$ (cf. Section 8.5). In the next step the partial eigensolution $(\widehat{\mathbf{D}}_u, \widehat{\mathbf{S}}_u)$ of the reduced problem $(\widehat{\mathbf{K}}_u, \widehat{\mathbf{M}}_u)$ is computed

$$\widehat{\mathbf{K}}_u \widehat{\mathbf{S}}_u = \widehat{\mathbf{M}}_u \widehat{\mathbf{S}}_u \widehat{\mathbf{D}}_u \qquad \text{with} \quad \widehat{\mathbf{S}}_u^T \widehat{\mathbf{M}}_u \widehat{\mathbf{S}}_u = \mathrm{Id},$$

where the diagonal matrix $\widehat{\mathbf{D}}_u \in \mathbb{R}^{k_u \times k_u}$ contains the $k_u \leq \bar{k}_u$ smallest eigenvalues and $\widehat{\mathbf{S}}_u \in \mathbb{R}^{\bar{k}_u \times k_u}$ column-wise the associated eigenvectors, and where k_u is chosen as in (8.14). As for the global reduced eigenvalue problem (cf. Section 8.5) the computation of the partial eigensolution $(\widehat{\mathbf{D}}_u, \widehat{\mathbf{S}}_u)$ is done by the dense LAPACK eigensolver `dsygvx`. In the last step of Algorithm 13 the eigenvectors of the reduced problem are transformed by computing $\widetilde{\mathbf{S}}_u := \mathbf{Z}_u \widehat{\mathbf{S}}_u$ (see Algorithm 15 for implementation), and the approximative partial eigensolution

$$\widetilde{K}_u^{\mathcal{H}} \, \widetilde{\mathbf{S}}_u \;\approx\; \widetilde{M}_u^{\mathcal{H}} \, \widetilde{\mathbf{S}}_u \, \widetilde{\mathbf{D}}_u \qquad \text{with} \quad \widetilde{\mathbf{S}}_u^T \, \widetilde{M}_u^{\mathcal{H}} \, \widetilde{\mathbf{S}}_u \;=\; \mathrm{Id}$$

is obtained where $\widetilde{\mathbf{D}}_u := \widehat{\mathbf{D}}_u$.

In summary, it could be seen how the spectral information $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$ of the subproblems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$ has been condensed via $\mathcal{H}$-AMLS into the spectral information of a single subproblem of larger size. In particular, in the following it is said that the *subproblems* $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$ *have been condensed* if in the further process of $\mathcal{H}$-AMLS the data $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$ and $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$ is replaced by the data $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ and $(\widetilde{\mathbf{D}}_u, \widetilde{\mathbf{S}}_u)$.

The question arises when subproblems should be condensed. The spectral information of $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$ and of $(\widetilde{\mathbf{D}}_u, \widetilde{\mathbf{S}}_u)$ is considered as equivalent concerning their contribution to the global reduced eigenvalue problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$. Overall, it is aimed to bound the size $\bar{k}$ of the global reduced problem by $\mathcal{O}(n_{\mathrm{es}})$. Correspondingly, in the case that $k_u < \bar{k}_u$ it is advantageous to condense the subproblems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$ in order to bound $\bar{k}$. On the other side the condensation of subproblems results in additional computational costs. Correspondingly, the subproblems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$ should only be condensed when, for example, it holds

$$\bar{k}_u \;>\; 2\, k_u. \tag{8.15}$$

This means that if *condensation condition* (8.15) is fulfilled it is considered as more efficient to condense the subproblems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})_{i=\mathrm{i\_first}(u)}^{\mathrm{i\_last}(u)}$ and to use the spectral information of problem $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ instead.

8.7.2. Implementation of the Condensation Process

The complete condensation process which is performed in the recursive version of $\mathcal{H}$-AMLS is described in Algorithm 16: In the first step of this process subproblems are identified which should be condensed. This is done by applying Algorithm 17 to the AMLS substructure tree $\mathcal{T}_{\mathrm{AMLS}}$. Let $\mathrm{depth}(\mathcal{T}_{\mathrm{AMLS}}) \in \mathbb{N}_0$ be the length of the longest path in $\mathcal{T}_{\mathrm{AMLS}}$, and $\mathrm{level}(t) \in \mathbb{N}_0$ the length of the path between the cluster $t \in \mathcal{T}_{\mathrm{AMLS}}$ and the root of $\mathcal{T}_{\mathrm{AMLS}}$. Applying Algorithm 17 the AMLS tree is traversed level-wise, starting from level $l = \mathrm{depth}(\mathcal{T}_{\mathrm{AMLS}}) - 1$ up to level $l = 1$, where every three levels clusters $u \in \mathcal{T}_{\mathrm{AMLS}}$ are tested if the cluster and the corresponding subproblems fulfil condensation condition (8.15). If this is the case the cluster u is appended to the set U. As soon as on a concrete level $l' \in \mathbb{N}$ clusters[8] are found whose associated subproblems should be condensed, then Algorithm 17 finishes and returns the set U back to Algorithm 16. Searching for clusters on levels $l < l'$ of the AMLS tree is first continued when all subproblems associated to the clusters $u \in U$ have been condensed. The corresponding

[8] Note that on one level there can be several clusters u fulfilling the condensation condition (8.15).

Algorithm 16 Implementation of the Condensation Process

procedure ApplyCondensation($\widetilde{K}^{\mathcal{H}}$, $\widetilde{M}^{\mathcal{H}}$, $\mathcal{T}_{\text{AMLS}}$, $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=1}^{m}$, $C_{\text{dom}}^{\approx}$, β_{dom})
  while true **do**
    ▷ *in the set U clusters are collected that represent subproblems which should be condensed*
    U :=FindSubproblemsToCondense($\mathcal{T}_{\text{AMLS}}$, $(k_i)_{i=1}^{m}$, $C_{\text{dom}}^{\approx}$, β_{dom}); ▷ *see Alg. 17*
    if $U \equiv \emptyset$ **then**
      return; ▷ *end condensation process when no further subproblems can be condensed*
    end if
    ▷ *compute for $u \in U$ (approx.) partial eigensol. $(\widetilde{\mathbf{D}}_u, \widetilde{\mathbf{S}}_u)$ of $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ → see Alg. 13*
    for all $u \in U$ **do**
      $(\widetilde{\mathbf{D}}_u, \widetilde{\mathbf{S}}_u)$:= ApplyHAMLSSub($\widetilde{K}^{\mathcal{H}}$, $\widetilde{M}^{\mathcal{H}}$, $\mathcal{T}_{\text{AMLS}}$, $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=1}^{m}$, u, $C_{\text{dom}}^{\approx}$, β_{dom});
      for $i = \text{i\_first}(u), \ldots, \text{i\_last}(u)$ **do**
        delete $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)$; ▷ *eigensol.s which are not needed anymore can be deleted*
      end for
    end for
    ▷ *introduce cluster-notation for the partial eigensol.s of old subprobl.s which are still in use*
    for all $u \in \left\{ t \in \mathcal{L}(\mathcal{T}_{\text{AMLS}}) \,:\, t \not\subseteq J \text{ with } J := \bigcup_{v \in U} v \right\}$ **do**
      $(\widetilde{\mathbf{D}}_u, \widetilde{\mathbf{S}}_u) := (\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)$ with $i := \psi^{-1}(u)$;
    end for
    ▷ *truncate the substructure tree $\mathcal{T}_{\text{AMLS}}$ by deleting the sons of the new used subproblems*
    for all $u \in U$ **do**
      $S(u) := \emptyset$;
    end for
    ▷ *update the auxiliary data according to the truncated substructure tree $\mathcal{T}_{\text{AMLS}}$*
    $m := \#\mathcal{L}(\mathcal{T}_{\text{AMLS}})$; ▷ *update the number of subproblems*
    initialise the bijection $\psi : \{1, \ldots, m\} \to \mathcal{L}(\mathcal{T}_{\text{AMLS}})$; ▷ *see (8.5) for definition of ψ*
    ▷ *initialise index-notation for the partial eigensol.s of the subprobl.s which are now in use*
    for $i = 1, \ldots, m$ **do**
      $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i) := (\widetilde{\mathbf{D}}_u, \widetilde{\mathbf{S}}_u)$ with $u := \psi(i)$;
    end for
  end while
end procedure

condensations are performed by Algorithm 13. Note that for all $u \in U$ the partial eigensolutions $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)_{i=\text{i\_first}(u)}^{\text{i\_last}(u)}$ are not needed anymore and that they can be deleted. Since old subproblems are replaced by new ones the auxiliary data of $\mathcal{H}$-AMLS has to be updated to the new situation. However, typically not all subproblems are condensed into superordinated subproblems, i.e., some old subproblems are still in use. For the partial eigensolutions of these old but still used subproblems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ the cluster-notation $(\widetilde{\mathbf{D}}_t, \widetilde{\mathbf{S}}_t) := (\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)$ is introduced where $t := \psi(i)$. The index-notation $(\widetilde{\mathbf{D}}_i, \widetilde{\mathbf{S}}_i)$ which has been used so far depends on the shape of the AMLS tree and becomes invalid when $\mathcal{T}_{\text{AMLS}}$ is updated. To update $\mathcal{T}_{\text{AMLS}}$ simply the sons of all $u \in U$ are deleted, i.e., it is set $S(u) := \emptyset$. Once the AMLS tree has been truncated the number of used subproblems m and the bijection ψ from (8.5) is updated, and finally, the index-notation for the partial eigensolutions of the old and the new created subproblems is introduced. After the auxiliary data has been updated the condensation process in Algorithm 16 is restarted from the beginning with the truncated tree until no further subproblems can be found which can be condensed. The choice that every three level subproblems are condensed can be replaced by other approaches, but ultimately this approach emulates a recursive $\mathcal{H}$-AMLS method with a

three-level substructuring.

Algorithm 17 Find subproblems which should be condensed

procedure FINDSUBPROBLEMSTOCONDENSE($\mathcal{T}_{\text{AMLS}}$, $(k_i)_{i=1}^m$, $C_{\text{dom}}^{\approx}$, β_{dom})
  ▷ *collect in the set U clusters representing subproblems which should be condensed*
  $U := \emptyset$;
  for $l = \text{depth}(\mathcal{T}_{\text{AMLS}}) - 1, \ldots, 1$ **do** ▷ $\text{depth}(\mathcal{T}_{\text{AMLS}})$:=*length of longest path in* $\mathcal{T}_{\text{AMLS}}$
    $\mathcal{T}_{\text{AMLS}}^{(l)} := \{u \in \mathcal{T}_{\text{AMLS}} : \text{level}(u) = l\}$;
    for all $u \in \mathcal{T}_{\text{AMLS}}^{(l)} \setminus \mathcal{L}(\mathcal{T}_{\text{AMLS}})$ **do**
      ▷ *$\bar{k}_u$ eigenvectors are selected when subproblems associated to u are not condensed*
      $\bar{k}_u := \sum_{i=\text{i\_first}(u)}^{\text{i\_last}(u)} k_i$;
      ▷ *k_u eigenvectors are selected when subproblems associated to u are condensed*
      $k_u := \min\{\, \lceil C_{\text{dom}}^{\approx}(N_u)^{\beta_{\text{dom}}} \rceil,\, N_u \,\}$ with $N_u := \sum_{\text{i\_first}(u)}^{\text{i\_last}(u)} N_i$;
      ▷ *if $\bar{k}_u$ is too large then the subproblems associated to cluster u should be condensed*
      if $l \equiv 0 \bmod 3$ **and** $\bar{k}_u > 2\,k_u$ **then**
        $U := U \cup \{u\}$;
      end if
    end for
    ▷ *stop searching if on current level subproblems have been found which should be condensed*
    if $\#U > 0$ **then**
      return U;
    end if
  end for
  return U;
end procedure

8.8. Parallelisation of the $\mathcal{H}$-AMLS method

In this section it is described how the $\mathcal{H}$-AMLS method has been parallelised for shared memory systems. Shared memory systems are computer systems with multiple CPU cores and a shared main memory which is directly accessible by all CPU cores. The communication between the different cores is solely realised by reading and writing data to the shared memory. In contrast to shared memory systems stand distributed memory systems, i.e., computer systems where each CPU has its own private memory, and where cores, respectively computational tasks, have to communicate explicitly via a message-passing system. However, this work is solely focused on parallelising $\mathcal{H}$-AMLS on shared memory systems. This topic is started by describing shared memory systems in more detail and by discussing basic issues of parallel computing.

A shared memory system with a memory architecture where each position in the memory is accessed in equal time by all CPU cores is called *uniform memory access architecture* (short *UMA-architecture*). However, the memory of shared memory systems is often constructed in some hierarchical form (especially for systems with many cores) where each core has — depending on its position in the system — local and non-local shared memory. If the access time to some parts of the memory differs for some cores then the system architecture is called *non-uniform memory access architecture* (short *NUMA-architecture*). Correspondingly, the parallel performance of a program running on a NUMA-architecture may depend as well on the distribution

of the data on the shared memory.

Another important issue of shared memory systems is that the memory can become easily a bottleneck when too many cores access data at the same time. The problem is the limited memory bandwidth of the system, and the contention between the cores over a shared memory bus that can be used only by a single core at once. Correspondingly, the cores are forced to wait until needed data is transferred to or from the memory. This problem is also referred to as the so-called *von Neumann bottleneck* or the *memory wall*. One attempt to mitigate this bottleneck is to provide *caches* between the cores and the main memory. Caches are very fast, but small, hardware memories on the CPUs which hold recently accessed data. If data is reused by a core and still present in the local cache, the core can access this data much faster than when it has to get the data from the main memory. However, CPU caches are not part of the shared main memory and, in particular, they are not considered for the classification of UMA- or NUMA-architectures.

Nowadays, shared memory systems (with UMA- and NUMA-architectures) are realised on nearly all workstations and compute servers, where the number of cores is typically in the range of 2 up to 128 or even more, and where the cores are equipped with several levels of caches.

On a shared memory system a program is executed by the operating system as a process[9]. In order to utilise multiple cores of the shared memory system within a process, *threads* are typically used. Threads are execution paths within a single process which can execute concurrently and which share all resources of the process such as the address space and the memory. Each thread is assigned one-to-one to a core by the operating system. To access threads in C++ programming a special interface is needed which is provided, for example, by the *Threading Building Blocks* [46] or by *OpenMP* [27].

When implementing a multi-threaded program one has to take care when multiple threads access (read/write) the same data. If the outcome of the program depends on the undetermined timing of computational tasks a so-called *race condition* is present. To avoid these unwanted race conditions the execution of data dependent tasks has to be scheduled. If no race condition is present the program is called *thread-safe*.

The parallel performance of a program is often measured by the following quantities:

Definition 8.3 (Speedup and Efficiency) *For a given algorithm let $t(p)$ be the runtime of the parallel implementation using $p \in \mathbb{N} \setminus \{1\}$ cores, and $t(1)$ the runtime of the best sequential implementation. Then the parallel speedup $S(p)$ and the parallel efficiency $E(p)$ of the parallel program are defined by*

$$S(p) := \frac{t(1)}{t(p)} \qquad \text{and} \qquad E(p) := \frac{S(p)}{p} = \frac{t(1)}{t(p)\,p}.$$

The speedup of a parallel program is optimal if it holds $S(p) = p$ for $p \in \mathbb{N}$. In practice, however, an optimal speedup is often not achieved due to a variety of overheads which are associated with the parallelisation: For example, overhead imposed by the use of parallel software, overhead caused by spawning and finishing of threads, overhead due to synchronisation of data dependent tasks (cores can become idle if for example input data for a tasks is not yet computed), and overhead due to load imbalances (e.g., because of different workloads per thread some cores

[9] A *process* is an instance of a running program.

may finish before others and become idle), etc. Furthermore, the speedup of a program can be limited by the von Neumann bottleneck. If for example the cores are required to perform minimal processing on large amounts of data then the memory can become a bottleneck, and cores become idle as well since they have to wait for data. Beside this, parts of the program may be unparallelisable so that they can be processed only by a single tasks: Denote $c_{\text{seq}} \in [0, 1]$ the fraction of the overall computational work in an algorithm which can be handled only sequentially. Furthermore, it is assumed that the overall computational work is fixed for all $p \in \mathbb{N}$. Then the time needed for the parallel execution of the corresponding program is bounded from below by

$$t(p) \;\geq\; c_{\text{seq}}\, t(1) + \frac{(1 - c_{\text{seq}})\, t(1)}{p},$$

and correspondingly the speedup of the program is bounded from above by

$$S(p) \;\leq\; \frac{1}{c_{\text{seq}} + (1 - c_{\text{seq}})/p} \;=:\; S_{\max}(p). \tag{8.16}$$

Statement (8.16) is known as *Amdahl's Law* [2], and it follows that the maximal speedup of the program is limited by its sequential part (we have $S(p) \leq S_{\max}(p) \leq 1/c_{\text{seq}}$ for all $p \in \mathbb{N}$) and that the parallel efficiency $E(p)$ of the program tends to zero for $p \to \infty$ (since $\lim_{p\to\infty} S_{\max}(p) = 1/c_{\text{seq}}$). But note that Amdahl's Law is only valid when the computational work is fixed for all $p \in \mathbb{N}$. However, in practice typically with increasing computer resources (i.e., more available cores) larger problems are solved, where often the parallelisable part of the overall computational work grows much faster than the sequential part. From this point of view, Amdahl's Law may be considered as too pessimistic concerning a realistic evaluation of the parallel performance of a program.

The Threading Building Blocks (short TBB) mentioned above have been used for the parallelisation of the HLIBpro v2.3 on shared memory systems, cf. [50]. TBB [46, 62] is a C++ template library which offers standard parallel constructs, such as special commands for parallelising loops. However, the main principle of the TBB parallelisation is the decomposition of the computational work into many small tasks that can be performed concurrently. To define these tasks and possible data dependencies certain TBB data structures and TBB commands have to be used. When the parallel program is executed the tasks are automatically mapped by the TBB library to worker threads where the scheduling of the threads is managed independently by TBB. In particular, the programmer has neither access to the worker threads nor access to their scheduling or management. The abstract task-based concept of the TBB parallelisation allows the programmer to implement scalable and portable code without knowing details of the used hardware and without knowing how the thread management is realised by the used operating system. To obtain highly scalable code, the programmer only has to implement many tasks which can be performed concurrently.

The TBB library (version 4.2) has been used as well to parallelise the most cost intensive computational parts of the $\mathcal{H}$-AMLS method. For the LAPACK/BLAS linear algebra routines which are required by the HLIBpro the *Intel Math Kernel Library* v10.3 (short *MKL*) has been used. MKL [45] is a parallel library for shared memory systems, in particular, the library is thread-safe[10]. Although the MKL linear algebra routines can be executed by multiple threads,

[10]Note that the standard LAPACK/BLAS library provided by Netlib (`www.netlib.org/lapack/`) is not thread-safe.

multi-threading has been deactivated for these routines — by setting the number of threads for the MKL routines globally to 1, cf. [45] — in order to avoid the overuse of multiple threads in the HLIBpro routines and the $\mathcal{H}$-AMLS method. The only exception from this is the parallelisation of task ($\mathbb{T}7$) where a multi-threaded MKL routine has been used. Details how task ($\mathbb{T}7$) and all other tasks of $\mathcal{H}$-AMLS have been parallelised are presented in the following sections. Details on the resulting parallel performance are given in Section 9.4 where numerical experiments are presented.

8.8.1. Parallelisation of Task ($\mathbb{T}1$)

The HLIBpro provides multi-threaded routines (cf. [50, 52]) for the construction of the cluster tree $\mathcal{T}_I$ (but not for the block cluster tree $\mathcal{T}_{I\times I}$) and the construction of the $\mathcal{H}$-matrix representations. However, benchmarks have shown (in Section 9.4 the used computer system is described) that the runtime of the multi-threaded computation did not improve against the sequential one, the computation became even slower. Beside the parallel overhead associated with the multi-threaded computation, especially, the limited memory bandwidth of the system inhibits that task ($\mathbb{T}1$) benefits from the multi-threaded computation. The work to be performed in task ($\mathbb{T}1$) is mainly characterised by initialising data and allocating memory, and is not computationally intensive. Even the concurrent construction of the $\mathcal{H}$-matrix representations $K^{\mathcal{H}}$ and $M^{\mathcal{H}}$ using two threads did not accelerate the computation. Henceforth, task ($\mathbb{T}1$) has been implemented in such a way that by default always a single thread is used for the corresponding computation.

8.8.2. Parallelisation of Task ($\mathbb{T}2$+$\mathbb{T}3$)

The computation of $K^{\mathcal{H}} \approx L^{\mathcal{H}}\widetilde{K}^{\mathcal{H}}(L^{\mathcal{H}})^T$ and $\widetilde{M}^{\mathcal{H}} \approx (L^{\mathcal{H}})^{-1}M^{\mathcal{H}}(L^{\mathcal{H}})^{-T}$ is performed using the corresponding multi-threaded HLIBpro routines (cf. [50, 52]). The $\mathrm{LDL}^{\mathrm{T}}$-factorisation in the $\mathcal{H}$-matrix format, which has to be performed for $K^{\mathcal{H}}$, is based on a block algorithm similar to the algorithm used for the block $\mathrm{LDL}^{\mathrm{T}}$-factorisation of a dense matrix, and which is applied recursively to the block structure of the $\mathcal{H}$-matrix [cf. (6.9)]. Beside the standard parallel implementation of this recursive algorithm, where the involved matrix operations are performed in parallel (taking data dependencies into account) on the local level of the recursion, the HLIBpro v2.3 offers as well a parallel implementation where all individual computational tasks are considered on a global scope of the $\mathrm{LDL}^{\mathrm{T}}$-factorisation (cf. [54]). The task-based implementation with global scope shows a much improved parallel scaling behaviour compared to the standard parallel implementation with the local scope (cf. [54]). Correspondingly, when $\mathcal{H}$-AMLS is applied with more than one thread the task-based implementation of the $\mathrm{LDL}^{\mathrm{T}}$-factorisation is used. However, due to additional overhead the runtime of the task-based implementation is slightly larger when only a single thread is used than the runtime of the standard recursive implementation. For this reason, the standard recursive implementation of the $\mathrm{LDL}^{\mathrm{T}}$-factorisation is used when $\mathcal{H}$-AMLS is applied with a single thread.

8.8.3. Parallelisation of Task ($\mathbb{T}4$)

In the parallel implementation of task ($\mathbb{T}4$) the partial eigensolutions of the subproblems $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ are computed in parallel for $i = 1, \ldots, m$. In context of the used TBB library in *parallel* means here and in the following that each subproblem is labelled as an own computational task which

is then scheduled independently by TBB to worker threads when the program is running. To further improve the parallel performance of task ($\mathbb{T}4$), the matrix-vector operations, which are involved in the eigensolver $\mathcal{H}$-SIL (cf. Section 8.4), are performed in parallel using the corresponding multi-threaded HLIBpro routines. All this results in a very high parallel granularity of task ($\mathbb{T}4$) which is becoming higher with increasing problem size.

Note that the computational work associated to each subdomain problem $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ is nearly the same, however, for interface eigenvalue problems the work is varying. In general, the size N_i of an interface problem $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ and the number k_i of sought eigenvectors are different varying for different i. This can potentially lead to load imbalances when the subproblems are solved in parallel, and the computational work assigned to the several threads is different. In the case of load imbalances the TBB library applies the so-called concept of *task-stealing* where computational tasks are transferred to other threads to avoid that threads become idle. But since the parallel scalability of the iterative eigensolver $\mathcal{H}$-SIL is limited, it may still happen that in the parallel execution of task ($\mathbb{T}4$) the solution of the largest interface problem is running while all other subproblems have been finished.

8.8.4. Parallelisation of Task ($\mathbb{T}6$)

The computation of the reduced matrices $\widehat{\mathbf{K}}$ and $\widehat{\mathbf{M}}$ is described in Algorithm 11. Note that all submatrices $\widehat{\mathbf{K}}_{ij}$ and $\widehat{\mathbf{M}}_{ij}$ can be computed concurrently. In order to avoid in the parallel implementation of Algorithm 11 load imbalances and overhead due to unnecessary task managing, in the first step all index pairs (i, j) are collected where a submatrix $\widehat{\mathbf{K}}_{ij}$ or $\widehat{\mathbf{M}}_{ij}$ has to be computed. For this purpose, we define the set $W \subset \{1, \ldots, m\}^2 \times \{0, 1\}$ by

$$W := \Big\{(i, j, 0) \,:\, 1 \le i \le m,\, 1 \le j \le i,\, \mathrm{dist}(\Omega_{\psi(i)}, \Omega_{\psi(j)}) = 0\Big\} \cup \Big\{(i, i, 1) \,:\, 1 \le i \le m\Big\}$$

where the triple $w \in W$ with $w = (i, j, 0)$ indicates that submatrix $\widehat{\mathbf{M}}_{ij}$ has to be computed, and $w = (i, i, 1)$ that $\widehat{\mathbf{K}}_{ii}$ has to be computed. In the next step all submatrices associated to the $w \in W$ are computed in parallel where, as noted before, the TBB library manages independently the scheduling of the parallel tasks to the worker threads. In general, however, the computational effort associated to each $w \in W$ is not equal. Furthermore, in the recursive version of $\mathcal{H}$-AMLS (i.e., when the condensation process is performed) the number of subproblems m is bounded and typically only few, but relatively large, submatrices $\widehat{\mathbf{K}}_{ii}$ and $\widehat{\mathbf{M}}_{ij}$ have to be computed. Hence, in order to avoid load imbalances and to provide a high parallel granularity the computation of each submatrix is performed in parallel as well.

The parallel computation of $\widehat{\mathbf{M}}_{ij} = \mathbf{S}_i^T \widetilde{M}_{ij}^{\mathcal{H}} \mathbf{S}_j$ (respectively, of $\widehat{\mathbf{K}}_{ii} = \mathbf{S}_i^T \widetilde{K}_{ii}^{\mathcal{H}} \mathbf{S}_i$) has been implemented as follows: In the first step the matrix $\mathbf{C} := A^{\mathcal{H}}\mathbf{B}$ has to be computed where $A^{\mathcal{H}} := \widetilde{M}_{ij}^{\mathcal{H}} \in \mathbb{R}^{N_i \times N_j}$ is in $\mathcal{H}$-matrix representation, and $\mathbf{B} := \mathbf{S}_j \in \mathbb{R}^{N_j \times k_j}$ and $\mathbf{C} \in \mathbb{R}^{N_i \times k_j}$ are in full matrix representation. In the case that the size of $\mathbf{B}$ and $\mathbf{C}$ is large enough both matrices are subdivided in $c \in \mathbb{N}$ block columns of (nearly) equal size, and the matrix product $\mathbf{C} = A^{\mathcal{H}}\mathbf{B}$ is computed block-wise in parallel. More precisely, let $C_{\mathrm{col}} \in \mathbb{N}$ be a predefined column width and $C_{\min} \in \mathbb{N}$ some predefined threshold. If the conditions

$$k_j > C_{\mathrm{col}} \quad \text{and} \quad N_i > C_{\min} \tag{8.17}$$

are fulfilled the matrices $\mathbf{C}$ and $\mathbf{B}$ are decomposed into $c := \lceil k_i / C_{\mathrm{col}} \rceil$ block columns of (nearly) equal size so that

$$\underbrace{\Big[\mathbf{C}_1, \ldots, \mathbf{C}_c\Big]}_{= \mathbf{C} \in \mathbb{R}^{N_i \times k_j}} = \underbrace{A^{\mathcal{H}}}_{\in \mathbb{R}^{N_i \times N_j}} \underbrace{\Big[\mathbf{B}_1, \ldots, \mathbf{B}_c\Big]}_{= \mathbf{B} \in \mathbb{R}^{N_j \times k_j}}, \tag{8.18}$$

and the block columns $\mathbf{C}_l := A^{\mathcal{H}} \mathbf{B}_l$ are computed in parallel for $l = 1, \ldots, c$. If condition (8.17) is not fulfilled the block decomposition (8.18) is not performed.

Furthermore, depending on the size and structure of the involved matrices, the computation of $\mathbf{C}' := A^{\mathcal{H}} \mathbf{B}'$ with $\mathbf{C}' := \mathbf{C}_l$ and $\mathbf{B}' := \mathbf{B}_l$ [respectively, $\mathbf{C}' := \mathbf{C}$ and $\mathbf{B}' := \mathbf{B}$ if (8.17) is not fulfilled] is performed in parallel as well. This is done the following way: If the $\mathcal{H}$-matrix $A^{\mathcal{H}}$ is associated to a leaf in the block cluster tree $\mathcal{T}_{I \times I}$ then $A^{\mathcal{H}}$ is a full matrix or an R(k)-matrix [cf. (6.9)], and the product $\mathbf{C}' = A^{\mathcal{H}} \mathbf{B}'$ is computed sequentially using the corresponding HLIBpro routine. However, if $A^{\mathcal{H}}$ is not associated to a leaf of $\mathcal{T}_{I \times I}$, then the $\mathcal{H}$-matrix $A^{\mathcal{H}}$ possess a block structure of the form (6.9). Assume in this case that $A^{\mathcal{H}}$ is decomposed into $r \in \mathbb{N}$ block rows and $c \in \mathbb{N}$ block columns, then the matrices $\mathbf{B}'$ and $\mathbf{C}'$ are decomposed correspondingly such that

$$\underbrace{\begin{bmatrix} \mathbf{C}'_1 \\ \vdots \\ \mathbf{C}'_r \end{bmatrix}}_{= \mathbf{C}' \in \mathbb{R}^{N_i \times k'}} = \underbrace{\begin{bmatrix} A^{\mathcal{H}}_{11} & \ldots & A^{\mathcal{H}}_{1c} \\ \vdots & \ddots & \vdots \\ A^{\mathcal{H}}_{r1} & \ldots & A^{\mathcal{H}}_{rc} \end{bmatrix}}_{= A^{\mathcal{H}} \in \mathbb{R}^{N_i \times N_j}} \underbrace{\begin{bmatrix} \mathbf{B}'_1 \\ \vdots \\ \mathbf{B}'_c \end{bmatrix}}_{= \mathbf{B}' \in \mathbb{R}^{N_j \times k'}} \quad \text{with } \mathbf{C}'_l := \sum_{l'=1}^{c} A^{\mathcal{H}}_{ll'} \mathbf{B}'_{l'} \text{ for } l = 1, \ldots, r. \tag{8.19}$$

In the case that the conditions

$$r > 1 \qquad \text{and} \qquad \max\{N_i, N_j\}\, k' > C_{\min} \tag{8.20}$$

are fulfilled then the block rows $\mathbf{C}'_l$ in (8.19) are computed in parallel for $l = 1, \ldots, r$; and if (8.20) is not fulfilled the block columns are computed sequentially.[11] The described approach for the parallel computation of $A^{\mathcal{H}} \mathbf{B}'$ is applied recursively in (8.19) for the computation of the matrix products $A^{\mathcal{H}}_{ll'} \mathbf{B}'_{l'}$. In summary, one can say that for the parallel computation of $\mathbf{C} := \widetilde{M}^{\mathcal{H}}_{ij} \widetilde{\mathbf{S}}_j$ the matrices $\widetilde{\mathbf{S}}_j$ and $\mathbf{C}$ are decomposed (depending on the size and the structure) first into block columns and then recursively into block rows.

After the computation of $\mathbf{C}$ the matrix $\widehat{\mathbf{M}}_{ij} = \widetilde{\mathbf{S}}_i^T \mathbf{C}$ has to be computed, where $\widetilde{\mathbf{S}}_i \in \mathbb{R}^{N_i \times k_i}$ and $\mathbf{C} \in \mathbb{R}^{N_i \times k_j}$ are both in full matrix representation and where typically $\max\{k_i, k_j\} \ll N_i$. If the conditions

$$k_i > C_{\mathrm{col}} \quad \text{and} \quad k_j > C_{\mathrm{col}} \quad \text{and} \quad N_i > C_{\min} \tag{8.21}$$

are fulfilled the matrix $\mathbf{E} := \widehat{\mathbf{M}}_{ij} \in \mathbb{R}^{k_i \times k_j}$ is decomposed into a $r \times c$ block structure with submatrices of (nearly) same size where $r := \lceil k_i / C_{\mathrm{col}} \rceil$ and $c := \lceil k_j / C_{\mathrm{col}} \rceil$. Then the matrix

[11] The HLIBpro actually uses the same approach for the parallel computation of the matrix product $A^{\mathcal{H}} \mathbf{B}'$, however, the block rows $\mathbf{C}'_l$ in (8.19) are computed in parallel only when $\min\{N_i, k'\} > C$ with some given constant $C > 0$. However, since in our setting k' is typically relative small and $N_i \gg k'$ condition (8.20) is better suited as a criterion for the parallel computation of $A^{\mathcal{H}} \mathbf{B}'$.

$\mathbf{D} := \widetilde{\mathbf{S}}_i^T$ is decomposed correspondingly into r block rows and $\mathbf{C}$ into c block columns so that the matrix product $\mathbf{E} = \mathbf{DC}$ can be written as

$$\underbrace{\begin{bmatrix} \mathbf{E}_{11} & \dots & \mathbf{E}_{1c} \\ \vdots & \ddots & \vdots \\ \mathbf{E}_{r1} & \dots & \mathbf{E}_{rc} \end{bmatrix}}_{= \mathbf{E} \in \mathbb{R}^{k_i \times k_j}} = \underbrace{\begin{bmatrix} \mathbf{D}_1 \\ \vdots \\ \mathbf{D}_r \end{bmatrix}}_{= \mathbf{D} \in \mathbb{R}^{k_i \times N_i}} \underbrace{\begin{bmatrix} \mathbf{C}_1, \dots, \mathbf{C}_c \end{bmatrix}}_{= \mathbf{C} \in \mathbb{R}^{N_i \times k_j}}, \tag{8.22}$$

and the submatrices $\mathbf{E}_{ll'} := \mathbf{D}_l \mathbf{C}_{l'}$ are computed in parallel for $1 \leq l \leq r$ and $1 \leq l' \leq c$. If condition (8.21) is not fulfilled decomposition (8.22) is not performed and the matrix $\mathbf{E} = \mathbf{DC}$ is computed sequentially.

In summary, due to the parallel computation of the submatrices $\widehat{\mathbf{M}}_{ij}$ and $\widehat{\mathbf{K}}_{ii}$ for $w \in W$, and due to the parallel computation of each submatrix itself, for task ($\mathbb{T}6$) a very high parallel granularity can be provided which is becoming higher with increasing problem size. In numerical tests $C_{\text{col}} := 150$ and $C_{\min} := 15,000$ have been a good choice and this value are used in the rest of the work.

8.8.5. Parallelisation of Task ($\mathbb{T}7$)

The reduced eigenvalue problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$ is solved using the dense LAPACK eigensolver `dsygvx` which is provided by the MKL library. As already mentioned, throughout the $\mathcal{H}$-AMLS method all MKL linear algebra routines are executed with a single thread in order to avoid the overuse of multiple threads in the parallel HLIBpro routines. Only in the case that the size $\bar{k}$ of the $\mathcal{H}$-reduced eigenvalue problem is larger than a predefined threshold $C_{\text{MKL}} \in \mathbb{N}$ the eigensolver `dsygvx` routine is applied with the maximum number of available threads. In numerical tests the threshold $C_{\text{MKL}} := 4000$ has been a good choice (i.e., for problems larger than this size the multi-threaded execution of `dsygvx` was noticeably faster than the single-threaded one) and has been used in the rest of this work.

8.8.6. Parallelisation of Task ($\mathbb{T}8$+$\mathbb{T}9$)

The implementation of task ($\mathbb{T}8$) is described in Algorithm 12. In the parallel version of $\mathcal{H}$-AMLS the computation of the matrices $\widetilde{\mathbf{S}}^{(i)} \in \mathbb{R}^{N_i \times n_{\text{es}}}$ is performed in parallel for $i = 1, \dots, m$. Furthermore, in order to provide a high parallel granularity for large sized problems, the computation of each matrix product $\widetilde{\mathbf{S}}^{(i)} = \widetilde{\mathbf{S}}_i \widehat{\mathbf{S}}^{(i)}$ is performed in parallel as well. This is done in the following way: Let $C_{\text{row}} \in \mathbb{N}$ be a predefined threshold. In the case that $N_i > C_{\text{row}}$ the matrices $\mathbf{F} := \widetilde{\mathbf{S}}_i \in \mathbb{R}^{N_i \times k_i}$ and $\mathbf{G} := \widetilde{\mathbf{S}}^{(i)} \in \mathbb{R}^{N_i \times n_{\text{es}}}$ are decomposed in $r := \lceil N_i / C_{\text{row}} \rceil$ block rows of (nearly) equal size such that

$$\underbrace{\begin{bmatrix} \mathbf{G}_1 \\ \vdots \\ \mathbf{G}_r \end{bmatrix}}_{= \mathbf{G} \in \mathbb{R}^{N_i \times n_{\text{es}}}} = \underbrace{\begin{bmatrix} \mathbf{F}_1 \\ \vdots \\ \mathbf{F}_r \end{bmatrix}}_{= \mathbf{F} \in \mathbb{R}^{N_i \times k_i}} \underbrace{\widehat{\mathbf{S}}^{(i)}}_{\in \mathbb{R}^{k_i \times n_{\text{es}}}} \tag{8.23}$$

and the submatrices $\mathbf{G}_l = \mathbf{F}_l \widehat{\mathbf{S}}^{(i)}$ are computed in parallel for $l = 1, \ldots, r$. If $N_i \leq C_{\text{row}}$ decomposition (8.23) is not performed and $\widetilde{\mathbf{S}}^{(i)} = \widetilde{\mathbf{S}}_i \widehat{\mathbf{S}}^{(i)}$ is computed sequentially. In numerical tests $C_{\text{row}} := 4000$ has been a good choice and has been used in all benchmarks presented in this work.

After the parallel computation of $\widetilde{\mathbf{S}}$ the matrix $\mathbf{S}_{\text{temp}} = (L^{\mathcal{H}})^{-T} \widetilde{\mathbf{S}}$ has to be computed (cf. Alg. 12). For this purpose the matrices $\mathbf{S}_{\text{temp}}$ and $\widetilde{\mathbf{S}}$ are subdivided into $c \in \mathbb{N}$ block columns of (nearly) equal size such that

$$\underbrace{\left[\mathbf{S}_{\text{temp}}^{(1)}, \ldots, \mathbf{S}_{\text{temp}}^{(c)}\right]}_{= \mathbf{S}_{\text{temp}} \in \mathbb{R}^{N \times n_{\text{es}}}} = \underbrace{(L^{\mathcal{H}})^{-T}}_{\in \mathbb{R}^{N \times N}} \underbrace{\left[\widetilde{\mathbf{S}}^{(1)}, \ldots, \widetilde{\mathbf{S}}^{(c)}\right]}_{= \widetilde{\mathbf{S}} \in \mathbb{R}^{N \times n_{\text{es}}}}. \tag{8.24}$$

The number of block columns c is defined by

$$c := \min\left\{p, \left\lfloor n_{\text{es}} / C'_{\text{col}} \right\rfloor\right\} \tag{8.25}$$

where $p \in \mathbb{N}$ denotes the number of used threads and $C'_{\text{col}} \in \mathbb{N}$ is some predefined minimal column width. Using block decomposition (8.24) the matrix $\mathbf{S}_{\text{temp}}$ is computed in parallel by solving the triangular system $(L^{\mathcal{H}})^T \mathbf{S}_{\text{temp}}^{(l)} = \widetilde{\mathbf{S}}^{(l)}$ for $\mathbf{S}_{\text{temp}}^{(l)}$ using the corresponding multi-threaded HLIBpro routine and computing $\mathbf{S}_{\text{temp}}^{(l)}$ in parallel for $l = 1, \ldots, c$. The number of block columns c is defined in such a way that, if the problem is large enough, to each thread the same work load is assigned. However, if the problem is not large enough the column width of the matrices $\widetilde{\mathbf{S}}^{(l)}$ and $\mathbf{S}_{\text{temp}}^{(l)}$ is bounded from below by C'_{col} in order to avoid efficiency losses. Of course the computation of $\mathbf{S}_{\text{temp}}$ can be performed as well vector-wise (i.e, where $c = n_{\text{es}}$), however, the matrix-wise approach is typically much more efficient in terms of runtime than the vector-wise approach since the matrix-wise approach can exploit data locality much better (cf. [50]). Although the vector-wise approach scales better with the number of cores, numerical tests have shown that even with 32 used CPU cores the parallel computation of $\mathbf{S}_{\text{temp}}$ with the matrix-wise approach has still been significantly faster than with the vector-wise approach. In numerical tests $C'_{\text{col}} := 20$ has been a good choice for the minimal column width and has been used in all benchmarks presented in this work.

In task ($\mathbb{T}9$) the Rayleigh quotients $\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}$ have to be computed (cf. Alg. 12). In the parallel version of $\mathcal{H}$-AMLS the computation of $\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}$ is done in parallel for $j = 1, \ldots, n_{\text{es}}$.

8.8.7. Parallelisation of Task ($\mathbb{T}$SI)

The implementation of task ($\mathbb{T}$SI) is described in Algorithm 4. For the parallelisation of this task the same approaches have been used as for the parallelisation of the previous tasks:

- The sparse matrix multiplication $A_1 := MS$ (cf. Algorithm 4) is performed in parallel by computing each column of A_1 in parallel. The computation of $A_2 := (L^{\mathcal{H}})^{-1} A_1$ and $A_4 := (L^{\mathcal{H}})^{-T} A_3$ is performed in parallel using the same approach as for the parallel computation of $\mathbf{S}_{\text{temp}} = (L^{\mathcal{H}})^{-T} \widetilde{\mathbf{S}}$ in task ($\mathbb{T}8$). Furthermore, the computation of $A_3 := (\widetilde{K}^{\mathcal{H}})^{-1} A_2$ is performed in parallel by the corresponding multi-threaded HLIBpro routine.

- The matrices K_Q and M_Q from Algorithm 4 are computed concurrently. Furthermore, the sparse matrix multiplications KQ and MQ are computed column-wise in parallel (same

approach as for $A_1 = MS$). The computation of $Q^T(KQ)$ and $Q^T(MQ)$ is performed in parallel using the approach described in (8.22).

- The reduced eigenvalue problem (K_Q, M_Q) is solved by the same parallel approach which has been described for task ($\mathbb{T}7$).
- The computation of $S_{\text{new}} = QS_Q$ is performed in parallel using the approach described in (8.23).

8.8.8. Parallelisation of the Condensation Process

The condensation process, which is applied in the recursive version of $\mathcal{H}$-AMLS, is described in Algorithm 16. In the first step of Algorithm 16 clusters $u \in \mathcal{T}_{\text{AMLS}}$ are collected in the set U which represent subproblems that should be condensed. Since the condensations associated to different clusters $u \in U$ can be handled independently, these condensations are computed in parallel in the parallel version of $\mathcal{H}$-AMLS. Furthermore, each condensation for itself is performed in parallel which is done in the following way: The condensation of the subproblems that are associated to a cluster $u \in U$ is performed by applying the $\mathcal{H}$-AMLS method to the problem $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$. See Section 8.7 for details and Algorithm 13 for the corresponding implementation. In the first step of Algorithm 13 the reduced eigenvalue problem $(\widehat{\mathbf{K}}_u, \widehat{\mathbf{M}}_u)$ is computed. For the parallel computation of $(\widehat{\mathbf{K}}_u, \widehat{\mathbf{M}}_u)$ the same approach has been used as for the parallel computation of task ($\mathbb{T}6$). After the solution of the reduced eigenvalue problem $(\widehat{\mathbf{K}}_u, \widehat{\mathbf{M}}_u)$ the transformed eigenvector matrix $\widetilde{\mathbf{S}}_u := \mathbf{Z}_u \widehat{\mathbf{S}}_u$ has to be computed. For the computation of $\widetilde{\mathbf{S}}_u$ the same parallel approach has been used as for the computation of $\widetilde{\mathbf{S}} := \mathbf{Z}\,\widehat{\mathbf{S}}$ in task ($\mathbb{T}8$).

9. Numerical Results

In this chapter we analyse numerically the $\mathcal{H}$-AMLS method for the Laplace eigenvalue problem

$$\begin{cases} -\Delta u = \lambda u & \text{in } \Omega = (0,1)^3, \\ u = 0 & \text{on } \partial\Omega. \end{cases} \tag{9.1}$$

Note that the underlying domain of (9.1) is three-dimensional and that it is very costly to solve this problem by the classical AMLS method (cf. Section 5.3). Problem (9.1) has already been discussed in Section 2.3 and it is one of the few examples where the eigensolutions can be derived analytically. In particular the eigenvalues of (9.1) are given by

$$\lambda = \lambda^{(a,b,c)} := \pi^2(a^2 + b^2 + c^2) \qquad a, b, c \in \mathbb{N},$$

and correspondingly it is possible to evaluate the relative errors

$$\widehat{\boldsymbol{\delta}}_j := \underbrace{\frac{|\,\lambda_j - \widehat{\boldsymbol{\lambda}}_j^{(\mathrm{rq})}\,|}{\lambda_j}}_{\substack{\text{relative error of} \\ \text{(recursive) } \mathcal{H}\text{-AMLS}}} \qquad \text{and} \qquad \widehat{\delta}_j^{(h)} := \underbrace{\frac{|\,\lambda_j - \lambda_j^{(h)}\,|}{\lambda_j}}_{\substack{\text{relative error} \\ \text{of discretisation}}}$$

which have been already introduced and discussed in Section 7.3, so that the approximation quality of $\mathcal{H}$-AMLS can be compared with the approximation quality of a classical approach.

To solve eigenvalue problem (9.1) by the $\mathcal{H}$-AMLS method or by a classical approach it is discretised as described in Section 5.3 using the finite element space of piecewise affine functions $\mathbb{X}^1_{h,0}$ with mesh width $h = 1/(n+1)$ and $N_h := \dim \mathbb{X}^1_{h,0} = n^3$ degrees of freedom. As described in Section 4.4 the discretisation results in the algebraic eigenvalue problem $(K^{(h)}, M^{(h)})$ of size N_h whose discrete eigenvalues $\lambda_j^{(h)}$ are approximating the sought smallest n_{es} eigenvalues λ_j of the continuous eigenvalue problem (9.1). For convenience we use in the following the short notation (K, M) instead of $(K^{(h)}, M^{(h)})$ and N instead of N_h.

The theoretical behaviour of the approximation error of $\lambda_j^{(h)}$ has been investigated in Section 3.4. Since it holds $u \in C^\infty(\Omega)$ [see (2.20)] for the eigenfunctions u of problem (9.1), it follows from Remark 3.19 that the error bounds presented in Theorem 3.14 are valid for the approximations $\lambda_j^{(h)}$. In particular, these bounds show how the error due to the discretisation depends on the mesh width h. To investigate the discretisation error numerically the eigenpairs of $(K^{(h)}, M^{(h)})$ associated to the smallest 500 eigenvalues have been computed for the mesh widths

- $h_1 := 0.05 \quad \Longrightarrow \; N = 6,859,$
- $h_2 := 0.025 \quad \Longrightarrow \; N = 59,319,$
- $h_3 := 0.0125 \; \Longrightarrow \; N = 493,039.$

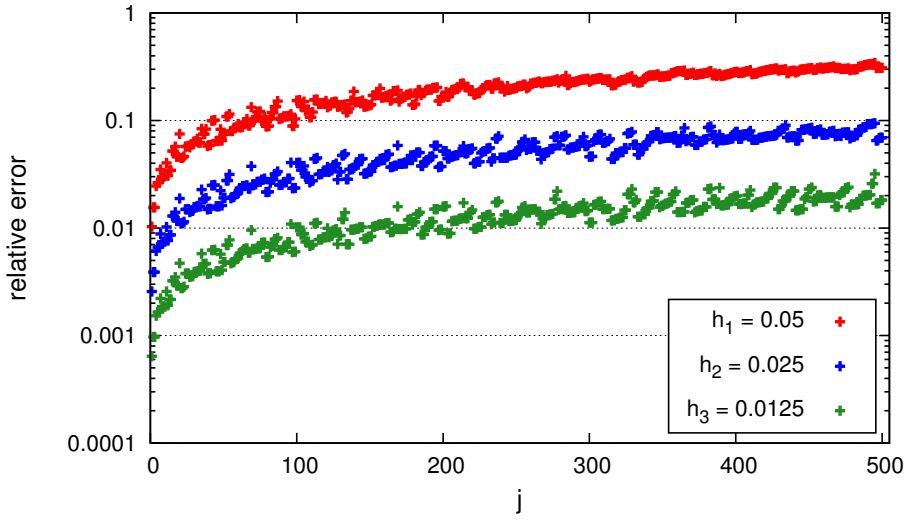


Figure 9.1.: Relative discretisation errors $\widehat{\delta}_j^{(h)}$ for the smallest 500 eigenvalues of problem (9.1) for varying mesh widths h.

For the computation of the discrete eigenpairs the eigensolver $\mathcal{H}$-ARPACK, presented in Section 8.4, has been used. For all eigenpairs computed by $\mathcal{H}$-ARPACK the relative residual error[1] has been smaller than 1e-9, however, it is noted that the discretisation error is significantly larger. Therefore, the eigenpairs computed by $\mathcal{H}$-ARPACK can be considered as numerically exact (as allowed by the finite element discretisation), and they can be used as references for the evaluation of the approximation quality of the $\mathcal{H}$-AMLS eigenpairs. In Table 9.1 and Figure 9.1 the discretisation errors of the computed eigenvalues $\lambda_j^{(h)}$ are displayed. It can be seen that smaller eigenvalues are better approximated than larger ones, and that the relative errors $\widehat{\delta}_j^{(h)}$ form more or less a monotonically increasing sequence in j. In particular, it can be observed that halving the mesh width reduces the errors by a factor of approximately 4 (as predicted by Theorem 3.14) but at the same time the system size increases by a factor of 8. To approximate more eigenvalues with the same accuracy a finer mesh width is necessary as it can be seen, e.g., in the last three columns of Table 9.1. For example, to compute the smallest 10 eigenvalues with a relative accuracy of 1e-2 the mesh width h_2 is sufficient while for the smallest 500 eigenvalues a mesh width finer than h_3 is necessary and correspondingly a model with more than 493,039 DOF is needed.

For a classical iterative eigensolver — such as $\mathcal{H}$-ARPACK or the subspace iteration presented in Section 7.4 — the leading computational costs are caused by the matrix-vector multiplications of the iteration vectors, and beside that possibly by costs caused by the computation of preconditioners (which are needed to solve the involved linear systems) and by the orthogonalisation of iteration vectors. However, neglecting possible computational costs for preconditioners and orthogonalisation, a lower bound for the best possible computational complexity of an eigensolver

[1] We define the *relative residual error* of an eigenpair approximation (λ, x) of (K, M) by $\|Kx - \lambda Mx\|_2 / \|Kx\|_2$. In literature this error is also referred to as modal error.

j	λ_j	**error** $\lvert\lambda_j - \lambda_j^{(h)}\rvert$			**relative error** $\widehat{\delta}_j^{(h)}$			$\max\{\, \delta_i^{(h)} \;:\; i = 1, \ldots, j \,\}$		
		h_1	h_2	h_3	h_1	h_2	h_3	h_1	h_2	h_3
1	29.60	0.30	0.07	0.01	1.02e-2	2.57e-3	6.42e-4	1.02e-2	2.57e-3	6.42e-4
2	59.21	0.92	0.23	0.05	1.55e-2	3.88e-3	9.71e-4	1.55e-2	3.88e-3	9.71e-4
3	59.21	0.92	0.23	0.05	1.55e-2	3.88e-3	9.71e-4	1.55e-2	3.88e-3	9.71e-4
4	59.21	1.45	0.36	0.09	2.45e-2	6.11e-3	1.52e-3	2.45e-2	6.11e-3	1.52e-3
5	88.82	2.34	0.58	0.14	2.64e-2	6.62e-3	1.65e-3	2.64e-2	6.62e-3	1.65e-3
10	108.56	3.31	0.81	0.20	3.05e-2	7.48e-3	1.86e-3	3.50e-2	8.83e-3	2.21e-3
20	167.78	12.58	3.16	0.79	7.49e-2	1.88e-2	4.71e-3	7.50e-2	1.88e-2	4.71e-3
30	207.26	13.26	3.24	0.80	6.39e-2	1.56e-2	3.90e-3	7.50e-2	1.88e-2	4.71e-3
40	256.61	15.14	3.87	0.97	5.90e-2	1.50e-2	3.79e-3	8.38e-2	2.27e-2	5.78e-3
50	286.21	21.27	5.46	1.37	7.43e-2	1.91e-2	4.81e-3	1.01e-1	2.51e-2	6.27e-3
100	414.52	63.52	16.69	4.19	1.53e-1	4.02e-2	1.01e-2	1.53e-1	4.06e-2	1.09e-2
200	641.52	106.84	33.27	10.17	1.66e-1	5.18e-2	1.58e-2	2.03e-1	5.87e-2	1.59e-2
300	819.17	188.28	37.64	9.20	2.29e-1	4.59e-2	1.12e-2	2.60e-1	7.29e-2	2.20e-2
400	967.22	273.93	75.76	19.03	2.83e-1	7.83e-2	1.96e-2	2.92e-1	8.83e-2	2.36e-2
500	1115.26	344.35	77.18	20.26	3.08e-1	6.92e-2	1.81e-2	3.42e-1	9.40e-2	3.19e-2

Table 9.1.: Errors between the eigenvalues λ_j of the continuous problem (9.1) and the eigenvalues $\lambda_j^{(h)}$ of the discretised problem $(K^{(h)}, M^{(h)})$ for varying mesh widths h. (All values given in this and the following tables are correct to two digits.)

would be

$$\mathcal{O}\big(\, n_{\text{es}}\, N \,\big), \tag{9.2}$$

when n_{es} eigenpairs of (K, M) are computed. Accordingly a possible measure for the performance of an eigensolver is the needed computational time per eigenpair and per one Million DOF, formally defined by $\text{avg}(t_{\text{all}})$, where t_{all} is the total time needed for the computation of the first n_{es} eigenpairs of (K, M) and

$$\text{avg}(t) := \text{avg}(t, n_{\text{es}}, N) := \frac{10^6\, t}{n_{\text{es}}\, N}. \tag{9.3}$$

Assume for example that a classical iterative approach has the best possible computational complexity where in average 10 iterations are needed until an iteration vector converges, and assume that the matrix-vector multiplication (by the inverse) takes 5 seconds per one million DOF, then the average computational time of this eigensolver is then given by $\text{avg}(t_{\text{all}}) = 50\text{s}$.

Applying the $\mathcal{H}$-AMLS method, the matrices of the discrete eigenvalue problem (K, M) are partitioned according to a geometric domain substructuring, represented in the corresponding $\mathcal{H}$-matrix format, and transformed using the fast $\mathcal{H}$-matrix arithmetic. Thereafter, the transformed problem $(\widetilde{K}^{\mathcal{H}}, \widetilde{M}^{\mathcal{H}})$ is projected onto a subspace derived from the partial eigensolutions of the subproblems and the $\mathcal{H}$-reduced eigenvalue problem (7.2) is obtained where the Rayleigh quotients $\widehat{\boldsymbol{\lambda}}_j^{(\text{rq})}$ of the corresponding Ritz-vectors are approximating the sought eigenvalues λ_j of (9.1). An overview of the involved eigenvalue problems and their interconnection is given in Figure 7.1. Beside the DOF of the model, the relative errors $\widehat{\boldsymbol{\delta}}_j$ depend on the number of selected eigenvectors k_i (modal truncation of the subproblems), the chosen accuracy ε of the

$\mathcal{H}$-matrix arithmetic, and if the recursive version of $\mathcal{H}$-AMLS is applied $\widehat{\boldsymbol{\delta}}_j$ depends as well on the parameters of the recursion. In the following we investigate how these parameters have to be chosen so that the eigenvalue approximations of $\mathcal{H}$-AMLS match the discretisation errors. In particular we will test for $n_{\text{es}} = N_h^{1/3}, 2N_h^{1/3}, 5N_h^{1/3}$ how the parameters have to be selected so that the inequality

$$\gamma_{n_{\text{es}}}^{(h)} < 3 \tag{9.4}$$

is fulfilled where

$$\gamma_{n_{\text{es}}}^{(h)} := \max\left\{ \widehat{\boldsymbol{\delta}}_j \,/\, \widehat{\delta}_j^{(h)} \;:\; j = 1, \ldots, n_{\text{es}} \right\} \tag{9.5}$$

is defined as the maximal ratio between the relative error $\widehat{\boldsymbol{\delta}}_j$ associated to $\mathcal{H}$-AMLS and the relative discretisation error $\widehat{\delta}_j^{(h)}$. In the case that inequality (9.4) is fulfilled it can be said that the approximation error of $\mathcal{H}$-AMLS is of the same order as the discretisation error (cf. Section 7.3).

Beside the approximation quality of $\mathcal{H}$-AMLS we investigate as well the computational time of the method, in particular its average time introduced in (9.3). We start in Section 9.1 with the analysis of non-recursive $\mathcal{H}$-AMLS, proceed in Section 9.2 with the analysis of the recursive version, and study in Section 9.3 the effect on $\mathcal{H}$-AMLS when task ($\mathbb{T}$SI) is performed instead of task ($\mathbb{T}$9). The numerical experiments presented in Section 9.1 – 9.3 apply the sequential version of $\mathcal{H}$-AMLS, and are performed on a 64-bit Linux platform with an Intel Xeon E5-4640 processor (2.40 GHz, 8 Cores). Furthermore, it is noted that $\mathcal{H}$-AMLS is implemented as described in Chapter 8 using the $\mathcal{H}$-matrix software library HLIBpro v2.3 where low-level linear algebra functions are provided by the Intel Math Kernel Library v10.3. If not indicated differently the $\mathcal{H}$-AMLS parameters are chosen as described in Chapter 8. The parallel performance of $\mathcal{H}$-AMLS (see Section 8.8 for implementation) is investigated in Section 9.4. Finally, in Section 9.5 results concerning the approximation quality of $\mathcal{H}$-AMLS are presented for more challenging elliptic PDE eigenvalue problems.

9.1. Analysis of Non-Recursive $\mathcal{H}$-AMLS

The analysis of non-recursive $\mathcal{H}$-AMLS is started by investigating how the approximation quality of the computed eigenpairs depends on the chosen modal truncation and the accuracy of the approximative $\mathcal{H}$-arithmetic. Non-recursive $\mathcal{H}$-AMLS is applied as described in Algorithm 5 (without performing the condensation process) where the multi-level domain substructuring is performed as described in Section 8.1 and Section 8.3. After determining a suitable parameter setting for non-recursive $\mathcal{H}$-AMLS the computational time of the method is analysed.

9.1.1. Influence of the Modal Truncation

To investigate solely the influence of the modal truncation — i.e., the influence of the number of selected eigenvectors k_i in task ($\mathbb{T}$4) — the parameter η from (6.2) is set to $\eta = 0$ in order to deactivate the $\mathcal{H}$-matrix approximation in (6.12). Correspondingly no subblock in the $\mathcal{H}$-matrix format is admissible, no R(k)-matrix approximation is applied and the computation of the transformed problem $(\widetilde{K}^{\mathcal{H}}, \widetilde{M}^{\mathcal{H}})$ in (6.12) is performed exactly (up to machine precision). Using this parameter setting $\mathcal{H}$-AMLS is equivalent with the classical AMLS method and correspondingly the computations will be very expensive as described in Section 5.3.

n_{es}	$\gamma_{n_{\text{es}}}^{(h)}$ for h_1		$\gamma_{n_{\text{es}}}^{(h)}$ for h_2		$\gamma_{n_{\text{es}}}^{(h)}$ for h_3	
	$\mathbb{S}1$	$\mathbb{S}2$	$\mathbb{S}1$	$\mathbb{S}2$	$\mathbb{S}1$	$\mathbb{S}2$
$N^{1/3}$	1.74	1.77	1.98	1.76	2.65	1.91
$2N^{1/3}$	1.79	1.80	2.36	2.00	5.12	2.16
$5N^{1/3}$	2.91	2.58	3.82	2.06	5.66	2.36

Table 9.2.: Influence of the mode selection strategy to the maximal ratios $\gamma_{n_{\text{es}}}^{(h)}$ for varying mesh widths and varying n_{es}. The $\mathcal{H}$-matrix approximation has been deactivated in this benchmarks by setting the parameter η to 0.

For the modal truncation the approach discussed in Remark 5.7 has been used and the following two mode selection strategies have been benchmarked:

strategy	subdomain problem	interface problem
• $\mathbb{S}1$	$k_i = 1.5N_i^{1/3}$	$k_i = 2N_i^{1/3}$
• $\mathbb{S}2$	$k_i = 1.5N_i^{1/3}$	$k_i = N_i^{1/2}$

If for example strategy $\mathbb{S}2$ is applied then in task ($\mathbb{T}4$) for a subproblem associated to a subdomain the smallest $k_i = 1.5N_i^{1/3}$ eigenpairs are computed and for a subproblem associated to an interface the smallest $k_i = N_i^{1/2}$ eigenpairs. The resulting relative errors $\widehat{\boldsymbol{\delta}}_j$ of the $\mathcal{H}$-AMLS eigenvalue approximations are displayed in Figure 9.2(a) for the mesh widths h_1, h_2 and h_3, and for comparison the discretisation errors $\widehat{\delta}_j^{(h)}$ are displayed as well. The maximal ratio $\gamma_{n_{\text{es}}}^{(h)}$ between both errors can be seen in Table 9.2 for $n_{\text{es}} = N_h^{1/3}, 2N_h^{1/3}, 5N_h^{1/3}$. Obviously, the approximation quality of $\mathcal{H}$-AMLS using mode selection strategy $\mathbb{S}1$ (where only $2N_i^{1/3}$ modes from the interface are selected) deteriorates as $h \to 0$.

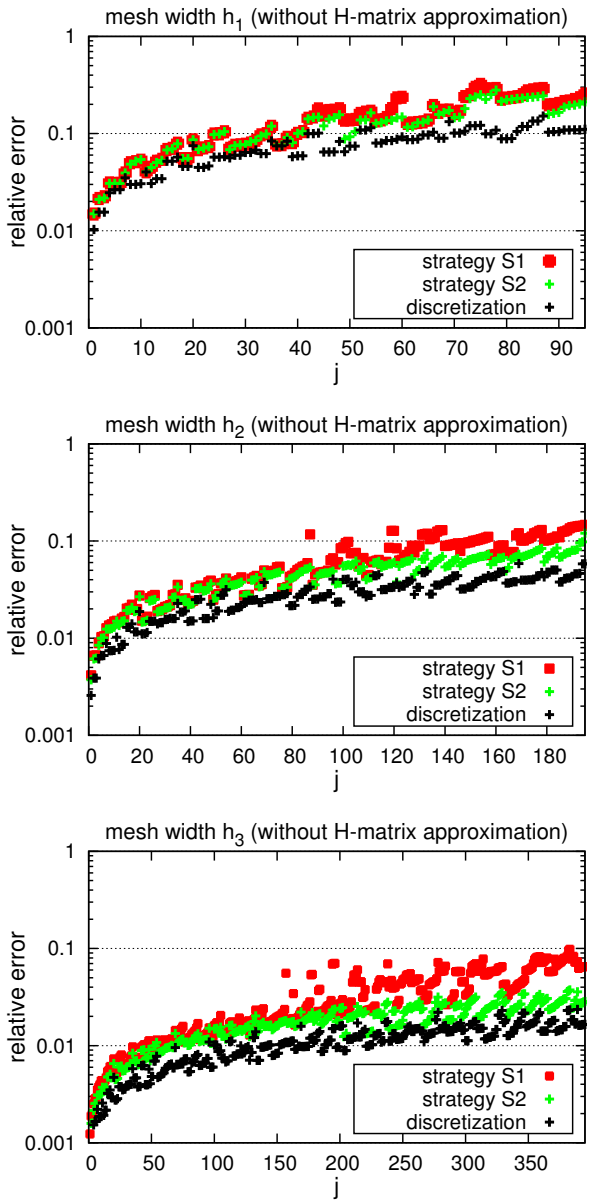
Furthermore, it can be seen in Table 9.2 that strategy $\mathbb{S}2$ is sufficient in such a way that for h_1,h_2 and h_3 postulation (9.4) is fulfilled for $n_{\text{es}} = N_h^{1/3}, 2N_h^{1/3}, 5N_h^{1/3}$. However, since $\eta = 0$ the computational costs of $\mathcal{H}$-AMLS are getting very expensive with increasing DOF.

9.1.2. Influence of the $\mathcal{H}$-Matrix Approximation

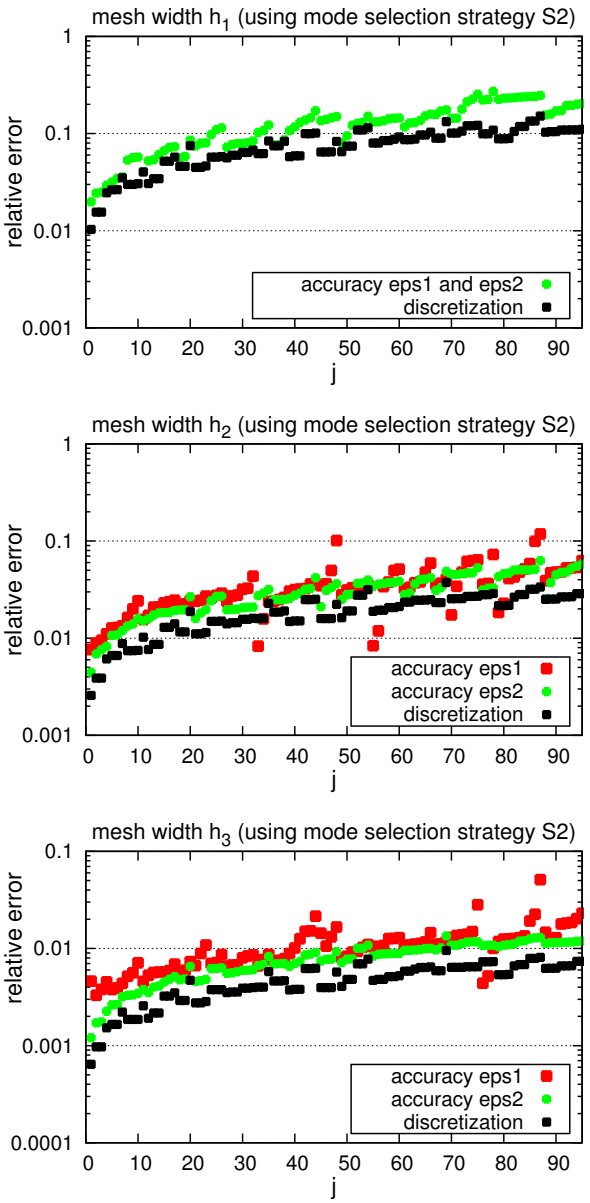
To accelerate the computation of the transformed matrices $\widetilde{K}^{\mathcal{H}}$ and $\widetilde{M}^{\mathcal{H}}$ in (6.12) the $\mathcal{H}$-matrix approximation is activated by setting the parameter η in (6.2) back to $\eta = 50$ (see Section 8.1 for details concerning the choice of η). Hence certain subblocks in the $\mathcal{H}$-matrix format get admissible and the respective submatrices are approximated by R(k)-matrices with a given approximation accuracy ε.

In the previous subsection could be seen that mode selection strategy $\mathbb{S}2$ is sufficient for the mesh widths h_1, h_2 and h_3. Using this mode selection strategy the computations from the previous subsection have been repeated applying the following accuracies for the $\mathcal{H}$-matrix approximation

- $\varepsilon_1 := \varepsilon_1(h) := 12\,h$
- $\varepsilon_2 := \varepsilon_2(h) := 240\,h^2$.



(a) Influence of the mode selection strategy to the relative errors $\widehat{\boldsymbol{\delta}}_j$. The $\mathcal{H}$-matrix approximation has been deactivated in this benchmarks by setting the parameter η to 0. Displayed are the approximation errors of the smallest $5N_h^{1/3}$ eigenvalues.



(b) Influence of the $\mathcal{H}$-matrix approximation accuracy ε to the relative errors $\widehat{\boldsymbol{\delta}}_j$. In this tests mode selection strategy $\mathbb{S}2$ has been applied. To highlight the influence of the $\mathcal{H}$-matrix accuracy on the approximation of the smallest eigenvalues only the errors of the first 95 eigenvalues are displayed.

Figure 9.2.: Influence of the mode selection strategy and the $\mathcal{H}$-matrix accuracy to the relative approximation errors $\widehat{\boldsymbol{\delta}}_j$ of $\mathcal{H}$-AMLS for varying mesh widths and comparison with the relative discretisation errors $\widehat{\delta}_j^{(h)}$.

n_{es}	$\gamma_{n_{\text{es}}}^{(h)}$ for h_1			$\gamma_{n_{\text{es}}}^{(h)}$ for h_2			$\gamma_{n_{\text{es}}}^{(h)}$ for h_3		
	ε_1	ε_2	$\eta = 0$	ε_1	ε_2	$\eta = 0$	ε_1	ε_2	$\eta = 0$
$N^{1/3}$	1.92	1.92	1.77	3.23	2.12	1.76	7.16	1.96	1.91
$2N^{1/3}$	1.97	1.97	1.80	4.53	2.12	2.00	7.16	2.26	2.16
$5N^{1/3}$	2.62	2.62	2.58	4.53	2.54	2.06	7.16	2.39	2.36

Table 9.3.: Influence of the $\mathcal{H}$-matrix approximation accuracy $\varepsilon = \varepsilon_1(h), \varepsilon_2(h)$ on the maximal ratios $\gamma_{n_{\text{es}}}^{(h)}$ for varying mesh widths and varying n_{es}. In this tests mode selection strategy $\mathbb{S}2$ has been applied.

The accuracies ε_1 and ε_2 depend on the discretisation mesh width of the underlying model, and for h_1, h_2, h_3 we obtain:

	h_1	h_2	h_3
$\varepsilon_1(h)$	0.6	0.3	0.15
$\varepsilon_2(h)$	0.6	0.15	0.0375

The relative errors $\widehat{\boldsymbol{\delta}}_j$ of this benchmark are displayed in Figure 9.2(b) and the maximal ratios $\gamma_{n_{\text{es}}}^{(h)}$ in Table 9.3. In Table 9.3 can be seen that mode selection strategy $\mathbb{S}2$ and $\mathcal{H}$-matrix accuracy ε_2 are sufficient for the mesh widths h_1, h_2 and h_3 to fulfil postulation (9.4) for all $n_{\text{es}} = N_h^{1/3}, 2N_h^{1/3}, 5N_h^{1/3}$. In particular, it is emphasised that this parameter setting adjusts the number of selected eigenpairs k_i automatically to the size of the corresponding subproblem, and the $\mathcal{H}$-matrix accuracy automatically to the underlying mesh width. Furthermore, in Figure 9.2(b) can be seen that especially the approximation of the smallest eigenvalues behaves sensitively to the chosen accuracy ε.

Altogether we could observe in our benchmarks that for the underlying model problem (9.1) the number of selected eigenpairs k_i in (7.1) should be of the order $\mathcal{O}\big(N_i^{1/3}\big)$ for subdomain problems (which are associated to three-dimensional subdomains) and of the order $\mathcal{O}\big(N_i^{1/2}\big)$ for interface problems (which are associated to hyperplanes in $\mathbb{R}^3$). The accuracy of the approximative $\mathcal{H}$-matrix arithmetic in (6.12) should be proportional to h^2, or respectively, expressed in DOF proportional to $N^{-2/3}$. This parameter setting is recommended for similar problems. If more accuracy of the $\mathcal{H}$-AMLS approximations is needed, k_i should be scaled by a constant larger than 1 and ε_2 by a constant smaller than 1.

9.1.3. Timing of the Method

The computational costs of non-recursive $\mathcal{H}$-AMLS, using mode selection strategy $\mathbb{S}2$ and $\mathcal{H}$-matrix accuracy ε_2, are given in Table 9.4 for $n_{\text{es}} = 5N_h^{1/3}$ and the mesh widths h_1, h_2 and h_3. Displayed are the costs of the different tasks (indicated in Table 7.1) and basic characteristics of the method such as the level of the applied domain substructuring (lvl), the resulting number of subproblems m, and the order $\bar{k}$ of the $\mathcal{H}$-reduced eigenvalue problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$. We observe that the computation of the block diagonalisation $K^{\mathcal{H}} \approx L^{\mathcal{H}} \widetilde{K}^{\mathcal{H}} (L^{\mathcal{H}})^T$ and the matrix transformation

parameter setting of $\mathcal{H}$-AMLS

- mode selection strategy $\mathbb{S}2$
- $\mathcal{H}$-matrix accuracy ε_2
- condensation process has been deactivated
- task ($\mathbb{T}9$) has been performed instead of task ($\mathbb{T}SI$)

	characteristics of non-recursive $\mathcal{H}$-AMLS					
	n_{es}	N_h	lvl	m	$\bar{k}$	$\gamma_{n_{\text{es}}}^{(h)}$
h_1	95	6,859	3	15	185	2.62
h_2	195	59,319	6	127	1,649	2.54
h_3	395	493,039	9	1023	13,537	2.39

	computational time of tasks in relation to total time								**computational time**	
	($\mathbb{T}1$)	($\mathbb{T}2$)	($\mathbb{T}3$)	($\mathbb{T}4$)	($\mathbb{T}6$)	($\mathbb{T}7$)	($\mathbb{T}8$)	($\mathbb{T}9$)	t_{all}	avg(t_{all})
h_1	1.8%	19.5%	55.2%	21.1%	0.4%	0.2%	0.9%	0.4%	5s	8.70s
h_2	1.3%	13.2%	69.0%	11.8%	0.5%	1.5%	1.7%	0.7%	1min 24s	7.34s
h_3	0.3%	6.0%	41.7%	3.8%	0.4%	45.4%	1.4%	0.5%	40min 09s	12.37s

Table 9.4.: Characteristics and computational costs of non-recursive $\mathcal{H}$-AMLS computing the smallest $n_{\text{es}} = 5N_h^{1/3}$ eigenpairs for varying mesh widths. t_{all} is the total computational time and avg(t_{all}) the average time defined in (9.3) using $n_{\text{es}} = 5N_h^{1/3}$. Since no computational costs are associated with task ($\mathbb{T}5$) it is left out in this and the following tables.

$\widetilde{M}^{\mathcal{H}} \approx (L^{\mathcal{H}})^{-1} M^{\mathcal{H}} (L^{\mathcal{H}})^{-T}$, task ($\mathbb{T}2$) and ($\mathbb{T}3$), are dominating the costs of the other tasks. However, with increasing DOF the portion of task ($\mathbb{T}2$) and ($\mathbb{T}3$) to the total computational time is decreasing. To keep in the benchmarks the computational costs of task ($\mathbb{T}4$) small the domain Ω has been substructured several times in order to obtain in (7.1) small subdomain eigenvalue problems which can be solved easily (see Section 8.1 and Section 8.3 for details). In particular, in order to keep the size of the subdomain problems constant when h is decreased, the number of substructuring levels in $\mathcal{H}$-AMLS (cf. Figure 8.1) has to increase as it can be seen in the column (lvl) of Table 9.4. The downside of a multi-level substructuring with constant sized subdomain problems is that the size $\bar{k}$ of the $\mathcal{H}$-reduced eigenvalue problem is $\mathcal{O}(N)$. Note particularly that the $\mathcal{H}$-reduced problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$ is only partially structured[2] and not sparse. Nevertheless, the cost savings achieved in task ($\mathbb{T}4$) outweigh the additional computational costs in tasks ($\mathbb{T}6$)–($\mathbb{T}8$) due to the larger $\bar{k}$. The eigenpairs of the $\mathcal{H}$-reduced problem have been computed by the dense LAPACK eigensolver `dsygvx` (cf. Section 8.5), and correspondingly we observe in Table 9.4 that the computational costs of task ($\mathbb{T}7$) are increasing much stronger than the costs of tasks ($\mathbb{T}6$) and ($\mathbb{T}8$). However, this issue can be resolved by applying the recursive version of $\mathcal{H}$-AMLS as it is shown in the following.

9.2. Analysis of Recursive $\mathcal{H}$-AMLS

In the previous section we observed that the transformed eigenvalue problem, task ($\mathbb{T}2$) and ($\mathbb{T}3$), can be successfully computed using the fast $\mathcal{H}$-matrix arithmetic which massively reduces

[2] $\widehat{\mathbf{K}}$ is a block diagonal matrix and $\widehat{\mathbf{M}}$ has a block-sparsity structure similar to the structure of $\widetilde{M}^{\mathcal{H}}$ (cf. Figure 8.2). See Section 8.5 for details.

		$\bar{k}$ for h_3	$\gamma_{n_{\text{es}}}^{(h)}$ for h_3 with n_{es} equal to			$\bar{k}$ for h_4	$\gamma_{n_{\text{es}}}^{(h)}$ for h_4 with n_{es} equal to		
			$N_h^{1/3}$	$2N_h^{1/3}$	$5N_h^{1/3}$		100	200	267
recursive $\mathcal{H}$-AMLS with $C_{\text{dom}}^{\approx}$ equal to	6	2,209	2.91	3.07	3.07	4,483	4.49	4.79	4.79
	8	2,833	2.58	2.80	2.85	5,747	3.92	4.07	4.07
	10	3,457	2.48	2.66	2.79	7,011	3.45	3.53	3.53
	12	4,081	2.26	2.55	2.73	8,275	3.21	3.36	3.36
	14	4,705	2.20	2.47	2.61	9,539	3.01	3.18	3.18
	16	5,329	2.12	2.42	2.55	10,803	2.84	2.95	2.95
non-recursive $\mathcal{H}$-AMLS		13,537	1.96	2.26	2.39	108,995	$(\star)$	$(\star)$	$(\star)$

Table 9.5.: Influence of the parameter $C_{\text{dom}}^{\approx}$ to the maximal ratios $\gamma_{n_{\text{es}}}^{(h)}$ for recursive $\mathcal{H}$-AMLS and the mesh widths h_3 and h_4. The error ratios $(\star)$ could not be evaluated since it was beyond the available computing capabilities to apply non-recursive $\mathcal{H}$-AMLS to the problem associated with mesh width h_4.

the computational time of AMLS. Task ($\mathbb{T}2$) and ($\mathbb{T}3$) are the computational bottleneck of the classical AMLS method, each with costs of at least the order $\mathcal{O}(N^2)$ for problems with a three-dimensional domain (cf. Section 5.3). Using the fast $\mathcal{H}$-matrix arithmetic these two tasks are now computed in almost linear complexity $\mathcal{O}(N \log^{\alpha} N)$ where the costs are independent of the number n_{es} of sought eigenvectors.

In this section we consider the recursive version of $\mathcal{H}$-AMLS where the size $\bar{k}$ of the $\mathcal{H}$-reduced eigenvalue problem can be bounded by $\mathcal{O}(N^{1/3})$, cf. Section 5.2.3. To apply the recursive version of $\mathcal{H}$-AMLS the condensation process in Algorithm 5 has to be performed. The condensation process, which is described in Section 8.7, implements the recursive call of $\mathcal{H}$-AMLS in a bottom-up fashion where the spectral information of several subproblems is condensed using $\mathcal{H}$-AMLS into the spectral information of a single subproblem of larger size.

The parameters for recursive $\mathcal{H}$-AMLS have been chosen as follows:

1. The $\mathcal{H}$-matrix accuracy $\varepsilon_2(h)$ from the previous section has been used in task ($\mathbb{T}2$) and ($\mathbb{T}3$) for the computation of the transformed eigenvalue problem.

2. The multi-level substructuring is applied in such a way that the size of the subdomain eigenvalue problems is smaller than the threshold $n_{\min}^{\text{AMLS}} = 1000$ (cf. Section 8.3).

3. In task ($\mathbb{T}4$) mode selection strategy $\mathbb{S}2$ has been applied, i.e., the smallest $k_i = 1.5N_i^{1/3}$ eigenpairs have been computed if the subproblem is associated to a subdomain and the smallest $k_i = N_i^{1/2}$ eigenpairs if the subproblem is associated to an interface. The small-sized subdomain eigenvalue problems have been solved directly by the dense LAPACK solver `dsygvx`. For small-sized interface problems the dense LAPACK solver has been used as well, for large-sized interface problems the iterative eigensolver $\mathcal{H}$-SIL introduced in Section 8.4 has been used.

4. The condensation process, which is performed after task (𝕋4) in Algorithm 5, is applied in such a way that subdomain eigenvalue problems are condensed every 3 levels and only if condition (8.15) has been fulfilled (cf. Algorithm 17). In the case that several subproblems are condensed by $\mathcal{H}$-AMLS into a single subdomain problem $(\widetilde{K}_u^{\mathcal{H}}, \widetilde{M}_u^{\mathcal{H}})$ of larger size, then the smallest $k_u = C_{\text{dom}}^{\approx}(N_u)^{1/3}$ eigenpairs are computed with some constant $C_{\text{dom}}^{\approx} > 0$.

In plain words the parameter setting described above implements a recursive approach, where after 3 levels of multi-level substructuring a total of $m = 15$ subproblems arise (8 subdomain and 7 interface eigenvalue problems), where small-sized subdomain problems are solved by the LAPACK solver `dsygvx` computing the smallest $k_i = 1.5N_i^{1/3}$ eigenpairs, and where large-sized subdomain problems are solved recursively by $\mathcal{H}$-AMLS computing the smallest $k_i = C_{\text{dom}}^{\approx} N_i^{1/3}$ eigenpairs.

Using the parameter setting described above recursive $\mathcal{H}$-AMLS has been benchmarked for the mesh widths h_1, h_2 and h_3; and even for the finer mesh width $h_4 := h_3/2$ which leads to a discrete eigenvalue problem (K, M) with roughly 4 million DOF. Since the problems associated to the mesh widths h_1 and h_2 are too small to fulfil condition (8.15) no condensation has been performed, and the same results are obtained as when the non-recursive version of $\mathcal{H}$-AMLS would have been applied (cf. Table 9.4). However, the problems associated to h_3 and h_4 are large enough so that subproblems can be condensed, and recursive $\mathcal{H}$-AMLS has been benchmarked for these problems using different choices of $C_{\text{dom}}^{\approx}$. In Table 9.5 the influence of the parameter $C_{\text{dom}}^{\approx}$ to the maximal ratios $\gamma_{n_{\text{es}}}^{(h)}$ has been displayed, as well as the resulting size $\bar{k}$ of the $\mathcal{H}$-reduced eigenvalue problem, and for comparison the corresponding results of the non-recursive approach. However, for the mesh width h_4 the maximal ratios $\gamma_{n_{\text{es}}}^{(h)}$ are displayed in Table 9.5 only up to $n_{\text{es}} = 267$. Due to computational limits it was only possible to compute with $\mathcal{H}$-ARPACK the smallest 267 eigenpairs of the discrete problem (K, M) associated to mesh width h_4. In Table 9.5 we observe that for mesh width h_3 already $C_{\text{dom}}^{\approx} = 8$ is sufficient to fulfil postulation (9.4) — i.e. that the approximation error of recursive $\mathcal{H}$-AMLS matches the discretisation error — while for mesh width h_4 at least $C_{\text{dom}}^{\approx} = 16$ is needed. Furthermore, the size $\bar{k}$ of the problem $(\widehat{\mathbf{K}}, \widehat{\mathbf{M}})$ is massively reduced by the recursive approach while nearly the same approximation quality is obtained as when the non-recursive approach is applied (cf. results for mesh width h_3 in Table 9.5).

In Table 9.6 the computational costs of recursive $\mathcal{H}$-AMLS are displayed for the computation of the smallest $n_{\text{es}} = 5N_h^{1/3}$ eigenpairs and using the parameter setting described above with $C_{\text{dom}}^{\approx} = 16$. This parameter setting is sufficient to obtain for the mesh widths h_1, h_2 and h_3 that $\gamma_{n_{\text{es}}}^{(h)} < 3$ is valid for $n_{\text{es}} = 5N_h^{1/3}$ (cf. Table 9.6). For the mesh width h_4 postulation $\gamma_{n_{\text{es}}}^{(h)} < 3$ could be validated due to computational limits only up to $n_{\text{es}} = 267$, but it is noted that the error ratios $\gamma_{n_{\text{es}}}^{(h)}$ are only slowly increasing in n_{es} (cf. Table 9.5). Beside the computational costs of the different tasks, in Table 9.6 are displayed basic characteristics of recursive $\mathcal{H}$-AMLS such as the order $\bar{k}$ of the $\mathcal{H}$-reduced problem (after condensation), and an overview of the performed condensation process. For example, for the problem associated to mesh width h_3 in total 8 condensations have been performed, all on level 3 of the AMLS tree (cf. Section 8.7). We remark that in Table 9.6 and in the following the time measurements concerning the tasks (𝕋6)–(𝕋8) are accumulative, e.g., for task (𝕋7) the computational time indicated by $\sum$(𝕋7) in Table 9.6 includes beside the time spent for task (𝕋7) as well the time spent for the solution of the $\mathcal{H}$-reduced eigenvalue problems of all recursive calls of $\mathcal{H}$-AMLS. In the benchmarks of

parameter setting of $\mathcal{H}$-AMLS

- mode selection strategy $\mathbb{S}2$
- $\mathcal{H}$-matrix accuracy ε_2
- condensation process has been activated with $C^{\approx}_{\mathrm{dom}} = 16$
- task ($\mathbb{T}9$) has been performed instead of task ($\mathbb{T}$SI)

	characteristics of recursive $\mathcal{H}$-AMLS				
	n_{es}	N_h	#condensations	$\bar{k}$	$\gamma^{(h)}_{n_{\mathrm{es}}}$
h_1	95	6,859	0	185	2.62
h_2	195	59,319	0	1,649	2.54
h_3	395	493,039	8 on lvl 3	5,329	2.55
h_4	795	4,019,679	64 on lvl 6, 8 on lvl 3	10,803	($\star$)

	computational time of tasks in relation to total time								**computational time**	
	($\mathbb{T}1$)	($\mathbb{T}2$)	($\mathbb{T}3$)	($\mathbb{T}4$)	$\sum(\mathbb{T}6)$	$\sum(\mathbb{T}7)$	$\sum(\mathbb{T}8)$	($\mathbb{T}9$)	t_{all}	avg(t_{all})
h_1	1.9%	22.4%	50.8%	22.4%	0.5%	0.2%	1.0%	0.4%	5s	8.14s
h_2	1.1%	13.1%	70.0%	11.2%	0.5%	1.4%	1.6%	0.6%	1min 30s	7.83s
h_3	0.6%	9.4%	65.9%	5.7%	10.4%	3.7%	3.2%	0.8%	25min 19s	7.80s
h_4	0.2%	6.1%	51.3%	3.2%	27.1%	5.4%	5.7%	0.6%	8h 9min 37s	9.19s

Table 9.6.: Characteristics and computational costs of recursive $\mathcal{H}$-AMLS computing the smallest $n_{\mathrm{es}} = 5N_h^{1/3}$ eigenpairs for varying mesh widths. The error ratio ($\star$) was beyond our computing capabilities (for mesh width h_4 only the smallest 267 discrete eigenpairs $\lambda_j^{(h)}$ could be computed, cf. Table 9.5)

non-recursive $\mathcal{H}$-AMLS we observed that the computational costs of task ($\mathbb{T}7$) became dominant with decreasing mesh width h, the cost became even so expensive that it was infeasible to apply non-recursive $\mathcal{H}$-AMLS to the problem associated with mesh width h_4. Since the size $\bar{k}$ of the $\mathcal{H}$-reduced eigenvalue problem has been bounded by $\mathcal{O}(N^{1/3})$ by the recursive approach, we observe in Table 9.6 that the computational costs associated with task ($\mathbb{T}7$) are massively reduced. The computational costs of recursive $\mathcal{H}$-AMLS are dominated by the computation of the transformed eigenvalue problem, i.e., by the computation of the LDL$^{\mathrm{T}}$-factorisation $K^{\mathcal{H}} \approx L^{\mathcal{H}} \widetilde{K}^{\mathcal{H}} (L^{\mathcal{H}})^T$ in task ($\mathbb{T}2$) and the computation of the matrix $\widetilde{M}^{\mathcal{H}} \approx (L^{\mathcal{H}})^{-1} M^{\mathcal{H}} (L^{\mathcal{H}})^{-T}$ in task ($\mathbb{T}3$).

9.3. Analysis of Recursive $\mathcal{H}$-AMLS with ($\mathbb{T}$SI)-improvement

In this section we analyse recursive $\mathcal{H}$-AMLS where the improvement task ($\mathbb{T}$SI) is performed instead of task ($\mathbb{T}9$). See Section 7.4 and Algorithm 3 for a description of task ($\mathbb{T}$SI). The benchmarks from the previous section are repeated and in Table 9.7 the resulting maximal error ratios $\gamma^{(h)}_{n_{\mathrm{es}}}$ are displayed for $n_{\mathrm{es}} = 5N_h^{1/3}$ with mesh widths h_3 and h_4, and for varying $C^{\approx}_{\mathrm{dom}}$. Analogously to (9.5), the value $\gamma^{(h)}_{n_{\mathrm{es}}}$ is the maximal ratio between the relative error associated to recursive $\mathcal{H}$-AMLS with ($\mathbb{T}$SI)-improvement and the relative discretisation error. Note that in contrast to the version of $\mathcal{H}$-AMLS where task ($\mathbb{T}9$) is applied, the maximal error ratios $\gamma_j^{(h)}$ (with $j \leq n_{\mathrm{es}}$) depend as well on the number n_{es} of computed eigenvectors since in the improvement task ($\mathbb{T}$SI), where an approximative iteration step of the subspace iteration is applied (cf. Section 7.4), the improvement of each eigenpair approximation becomes better the

		$\bar{k}$ for h_3	$\gamma_j^{(h)}$ for h_3 with j equal to			$\bar{k}$ for h_4	$\gamma_j^{(h)}$ for h_4 with j equal to		
			$N_h^{1/3}$	$2N_h^{1/3}$	$5N_h^{1/3}$		100	200	267
recursive $\mathcal{H}$-AMLS applied with (TSI) and $C_{\mathrm{dom}}^{\approx}$ equal to	0.5	497	1.64	5.39	44.31	1,011	1.57	3.72	5.86
	1	649	1.20	1.74	7.72	1,323	1.21	1.52	1.77
	1.5	809	1.21	1.53	3.93	1,643	1.22	1.47	1.87
	2	961	1.21	1.27	2.60	1,955	1.22	1.24	1.41
	4	1,585	1.21	1.21	1.39	3,219	1.22	1.22	1.22
	6	2,209	1.21	1.21	1.23	4,483	1.22	1.22	1.22
non-recursive $\mathcal{H}$-AMLS applied with task (TSI)		13,537	1.20	1.20	1.20	108,995	$(\star)$	$(\star)$	$(\star)$

Table 9.7.: Influence of the parameter $C_{\mathrm{dom}}^{\approx}$ to the maximal ratios $\gamma_j^{(h)}$ for recursive $\mathcal{H}$-AMLS with (TSI)-improvement with $n_{\mathrm{es}} = 5N_h^{1/3}$ and mesh widths h_3 and h_4. The error ratios $(\star)$ could not be evaluated since it was beyond the available computing capabilities to apply non-recursive $\mathcal{H}$-AMLS to the problem associated with mesh width h_4.

larger the dimension of the iterative subspace is [see (7.9)]. Comparing Table 9.5 and Table 9.7 we observe that replacing task (T9) by task (TSI) leads to a clear improvement of the approximation quality. In particular, we observe that already for $C_{\mathrm{dom}}^{\approx} = 2$ postulation (9.4) is fulfilled (for h_4 it could be only verified up to the 267-th eigenvalue), and that for the mesh width h_3 the approximation quality of recursive $\mathcal{H}$-AMLS is nearly as good as of non-recursive $\mathcal{H}$-AMLS [where both versions are applied with task (TSI)] when the parameter $C_{\mathrm{dom}}^{\approx} = 6$ is used.

In Table 9.8 basic characteristics and the computational costs of recursive $\mathcal{H}$-AMLS with (TSI)-improvement are displayed for the computation of the smallest $n_{\mathrm{es}} = 5N_h^{1/3}$ eigenpairs. In addition to that in Table 9.9 the memory consumption is displayed of the different matrices that are involved in the corresponding computations. Although the computational costs of tasks (TSI) are in general larger than those of task (T9) the overall costs of recursive $\mathcal{H}$-AMLS could be decreased which is due to the fact that instead of the parameter $C_{\mathrm{dom}}^{\approx} = 16$ only $C_{\mathrm{dom}}^{\approx} = 6$ could be used for the computations. Of particular note here is that in the benchmarks also the approximation quality of recursive $\mathcal{H}$-AMLS has been further improved, leading to an approximation error of the eigenvalues which is only slightly larger than the discretisation error.

To get a better impression of the practical performance of $\mathcal{H}$-AMLS we investigate the average computational time $\mathrm{avg}(t_{\mathrm{all}})$ of the method. The average computational time (per eigenpair and per DOF) is defined in (9.3) and in the following we analyse $\mathrm{avg}(t_{\mathrm{all}})$ in detail only for recursive $\mathcal{H}$-AMLS with (TSI)-improvement since this $\mathcal{H}$-AMLS version shows the best performance regarding computational time and approximation quality. In Figure 9.3 the average time of this $\mathcal{H}$-AMLS version is displayed for the computation of the smallest $n_{\mathrm{es}} = 5N^{1/3}$ eigenpairs for varying DOF with N up to 6 million. Beside the total average time also the average time of each task (for (T6)–(T8) accumulated as explained above) has been measured separately and displayed in Figure 9.3 in order to profile the complexity of the involved tasks [(T1),...,(T8),(TSI)]

parameter setting of $\mathcal{H}$-AMLS

- mode selection strategy $\mathbb{S}2$
- $\mathcal{H}$-matrix accuracy ε_2
- condensation process has been activated with $C_{\mathrm{dom}}^{\approx} = 6$
- task ($\mathbb{T}$SI) has been performed instead of task ($\mathbb{T}$9)

	characteristics of recursive $\mathcal{H}$-AMLS				
	n_{es}	N_h	#condensations	$\bar{k}$	$\gamma_{n_{\mathrm{es}}}^{(h)}$
h_1	95	6,859	0	185	1.77
h_2	195	59,319	0	1,649	1.53
h_3	395	493,039	8 on lvl 3	2,209	1.23
h_4	795	4,019,679	64 on lvl 6, 8 on lvl 3	4,483	($\star$)

	computational time of tasks in relation to total time								**computational time**	
	($\mathbb{T}$1)	($\mathbb{T}$2)	($\mathbb{T}$3)	($\mathbb{T}$4)	$\sum(\mathbb{T}6)$	$\sum(\mathbb{T}7)$	$\sum(\mathbb{T}8)$	($\mathbb{T}$SI)	t_{all}	avg(t_{all})
h_1	1.7%	21.2%	49.9%	21.7%	0.4%	0.2%	1.0%	3.4%	5s	8.39s
h_2	1.3%	12.8%	65.8%	11.2%	0.4%	1.4%	1.6%	5.1%	1min 30s	7.84s
h_3	0.6%	9.8%	65.7%	5.9%	3.9%	0.9%	2.8%	10.0%	24min 55s	7.67s
h_4	0.3%	7.1%	59.9%	3.7%	10.6%	0.6%	4.0%	13.5%	6h 58min 14s	7.85s

Table 9.8.: Characteristics and computational costs of recursive $\mathcal{H}$-AMLS with with ($\mathbb{T}$SI)-improvement computing the smallest $n_{\mathrm{es}} = 5N_h^{1/3}$ eigenpairs for varying mesh widths. The error ratio ($\star$) was beyond our computing capabilities (for mesh width h_4 only the smallest 267 discrete eigenpairs $\lambda_j^{(h)}$ could be computed, cf. Table 9.7).

	K & M	$K^{\mathcal{H}}$ & $M^{\mathcal{H}}$	$\widetilde{K}^{\mathcal{H}}$	$L^{\mathcal{H}}$	$\widetilde{M}^{\mathcal{H}}$	$\widehat{\mathbf{K}}$ & $\widehat{\mathbf{M}}$	$\mathbf{S}$
h_1	3.12	32.38	1.66	19.12	16.86	0.54	5.21
h_2	28.26	376.32	15.00	247.99	212.17	43.50	92.53
h_3	239.78	3,580.34	127.34	2,527.60	2,478.40	78.06	1,558.00
h_4	1,974.4	31,340.46	1,049.00	24,058.38	29,103.95	321.54	25,565.15

Table 9.9.: Corresponding to the benchmarks presented in Table 9.8 the memory consumption of the matrices in megabyte which are involved in the corresponding computations. Note that $\mathbf{S} \in \mathbb{R}^{N \times n_{\mathrm{es}}}$ is the full matrix containing column-wise the computed eigenvector approximations (cf. Algorithm 5).

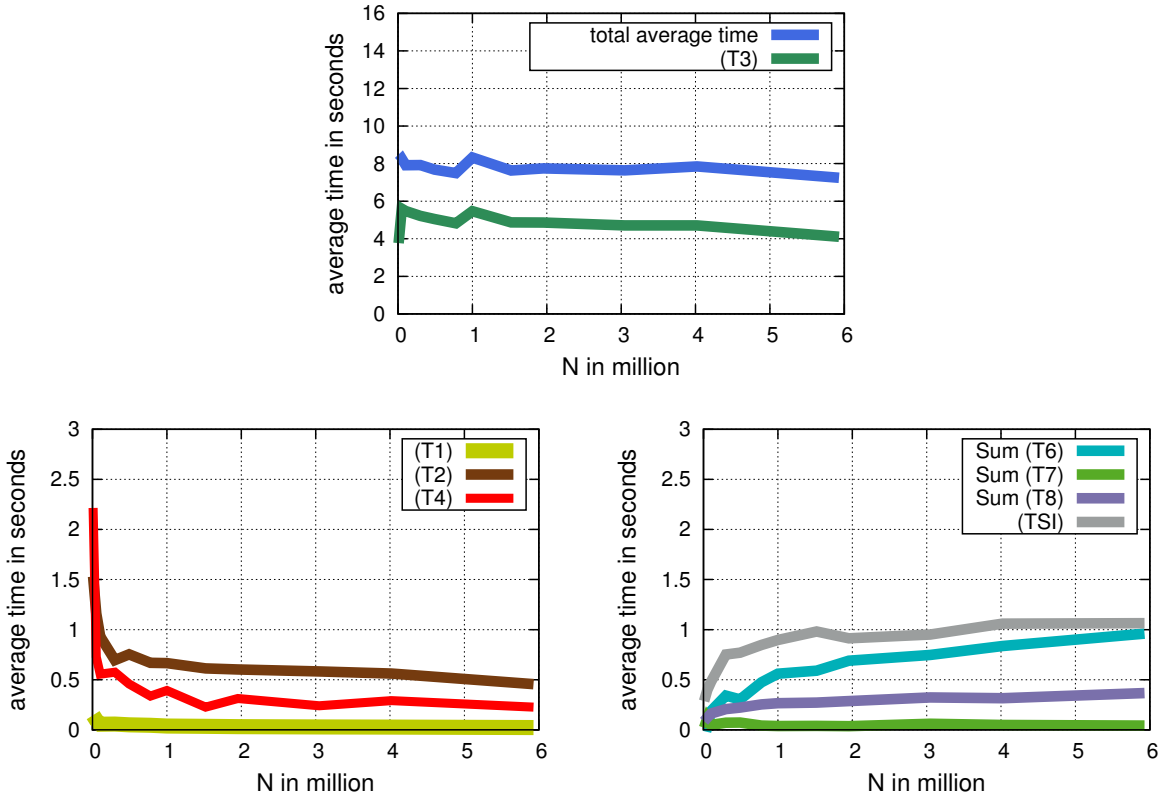


Figure 9.3.: Average computational time (per eigenpair and per one million DOF) of recursive $\mathcal{H}$-AMLS with (TSI)-improvement for the computation of the smallest $n_{\text{es}} = 5N^{1/3}$ eigenpairs in relation to the degrees of freedom N. Displayed are the total average computational time $\text{avg}(t_{\text{all}})$ and the average computational time of the different tasks. To provide a better presentation these times are displayed in separate figures.

in more detail. First of all it can be observed in Figure 9.3 that the total average time of the method is constant, which means in particular that recursive $\mathcal{H}$-AMLS reaches in the presented benchmarks the optimal complexity $\mathcal{O}(n_{\text{es}}N)$. Furthermore, the results in Figure 9.3 show that for all involved tasks the average computational time is roughly constant, and that the computational costs are clearly dominated by task (T3), i.e., by the computation of transformed mass matrix $\widetilde{M}^{\mathcal{H}}$.

In the benchmarks presented in Section 9.1 – Section 9.3 the parameter setting of $\mathcal{H}$-AMLS (for both the recursive and the non-recursive version) has been chosen in such a way that postulation (9.4) is fulfilled, i.e., that the error of the $\mathcal{H}$-AMLS eigenvalue approximations matches the discretisation error. In Table 9.3, Table 9.5 and Table 9.7 we observe that the ratio between the errors $\widehat{\boldsymbol{\delta}}_j$ and $\widehat{\delta}_j^{(h)}$ is only slowly increasing in j. Correspondingly it seems that in the presented benchmarks much more than $5N_h^{1/3}$ eigenvalue approximations can be computed with nearly the same approximation quality as the discretisation. Increasing the number of sought eigenpairs, however, increases only slightly the computational costs of $\mathcal{H}$-AMLS as it can be seen for example

n_{es}	$\mathcal{H}$-ARPACK	recursive $\mathcal{H}$-AMLS with ($\mathbb{T}$9)				recursive $\mathcal{H}$-AMLS with ($\mathbb{T}$SI)			
	avg(t_{all})	avg(t_{all})	$\gamma_j^{(h)}$ with j equal to			avg(t_{all})	$\gamma_j^{(h)}$ with j equal to		
			$\lceil n_{\text{es}}/4 \rceil$	$\lceil n_{\text{es}}/2 \rceil$	n_{es}		$\lceil n_{\text{es}}/4 \rceil$	$\lceil n_{\text{es}}/2 \rceil$	n_{es}
10	269.86s	284.73s	2.05	2.05	2.07	286.41s	1.23	1.23	1.23
20	164.24s	143.30s	2.05	2.05	2.07	143.98s	1.23	1.23	1.23
50	100.62s	57.96s	2.07	2.07	2.25	58.48s	1.22	1.22	1.22
$N^{1/3} = 79$	119.85s	36.13s	2.07	2.07	2.25	36.68s	1.22	1.22	1.22
$2N^{1/3}$	100.46s	17.14s	2.07	2.25	2.56	17.50s	1.22	1.22	1.22
$5N^{1/3}$	109.92s	7.68s	2.25	2.56	2.73	8.20s	1.20	1.20	1.20
$10N^{1/3}$	128.71s	3.83s	2.56	2.73	2.96	4.64s	1.19	1.19	1.77
$20N^{1/3}$	128.80s	2.25s	2.73	2.96	3.32	3.24s	1.18	1.67	1.96
$50N^{1/3}$	($\star$)	1.20s	($\star$)	($\star$)	($\star$)	2.83s	($\star$)	($\star$)	($\star$)

Table 9.10.: The average computational time of recursive $\mathcal{H}$-AMLS for computing the smallest n_{es} eigenpairs of the problem associated to mesh width h_3 with $N = 79^3$ degrees of freedom. The parameter setting of $\mathcal{H}$-AMLS has been chosen as described in Section 9.2 with the fixed parameter $C_{\text{dom}}^{\approx} = 12$ used for all n_{es} (cf. Figure 9.4). For the sake of completeness, the average computational time of the reference solver $\mathcal{H}$-ARPACK is displayed as well, and the maximal ratios $\gamma_j^{(h)}$ for $j \leq n_{\text{es}}$. The values ($\star$) are unavailable since the computation of the smallest $n_{\text{es}} = 50N_h^{1/3}$ eigenpairs using $\mathcal{H}$-ARPACK was beyond the computing capabilities.

in Figure 9.4. In Figure 9.4 the total computational time of recursive $\mathcal{H}$-AMLS, applied once with task ($\mathbb{T}$9) and once with task ($\mathbb{T}$SI) instead, is displayed for varying n_{es} with n_{es} up to $50N^{1/3}$. We observe that the computational costs of recursive $\mathcal{H}$-AMLS are nearly constant in n_{es} when task ($\mathbb{T}$9) is applied, and that the costs are slowly increasing in n_{es} when instead of task ($\mathbb{T}$9) the improvement task ($\mathbb{T}$SI) is applied. In Figure 9.5 the corresponding average computational time avg(t_{all}) is displayed which is decreasing very fast until it reaches (already for $n_{\text{es}} \geq 10N^{1/3}$) a range of less than 5s. In particular, this runtime behaviour makes the $\mathcal{H}$-AMLS method very attractive when many eigenpairs are sought. Furthermore, in Table 9.10 can be seen that the accuracy of the computed $\mathcal{H}$-AMLS eigenpair approximations reaches nearly the approximation quality of a classical approach, in particular, when task ($\mathbb{T}$SI) is applied. For the sake of completeness, the average computational time of $\mathcal{H}$-ARPACK is displayed in Table 9.10 as well. Note that the classical eigensolver $\mathcal{H}$-ARPACK uses the efficient and in practice widely used ARPACK library for the computation of the sought eigensolutions. Furthermore, $\mathcal{H}$-ARPACK uses the fast $\mathcal{H}$-arithmetic for the computation of the needed preconditioner (cf. Section 8.4) which is performed in (nearly) optimal complexity $\mathcal{O}(N \log^{\alpha} N)$ and hence is very inexpensive especially for three-dimensional problems.

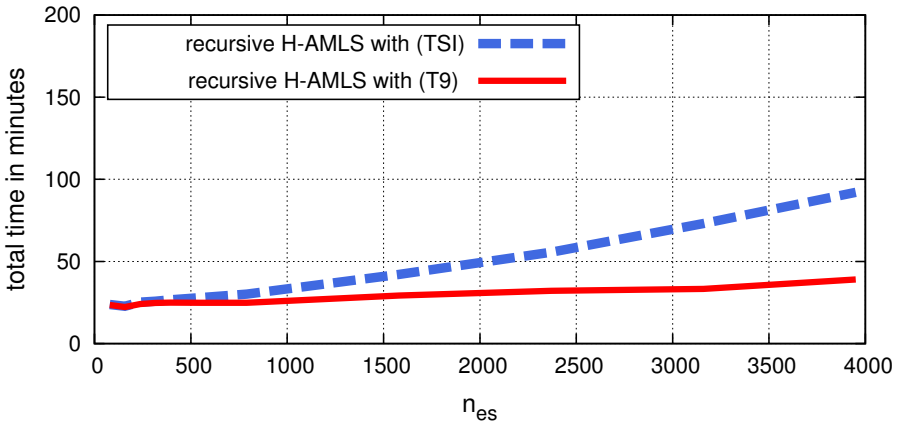


Figure 9.4.: Total computational time of recursive $\mathcal{H}$-AMLS, applied once with task (T9) and once with task (TSI) instead, for computing the smallest n_{es} eigenpairs of the problem associated to mesh width h_3. n_{es} is varying from $N^{1/3} = 79$ up to $50N^{1/3} = 3950$. The parameter setting of $\mathcal{H}$-AMLS has been chosen as described in Section 9.2. In order to have a sufficiently large reduced eigenvalue problem ($\bar{k} \geq 3950$ is needed) the parameter $C_{\text{dom}}^{\approx}$ has been set to 12 for all n_{es} in both benchmarks.

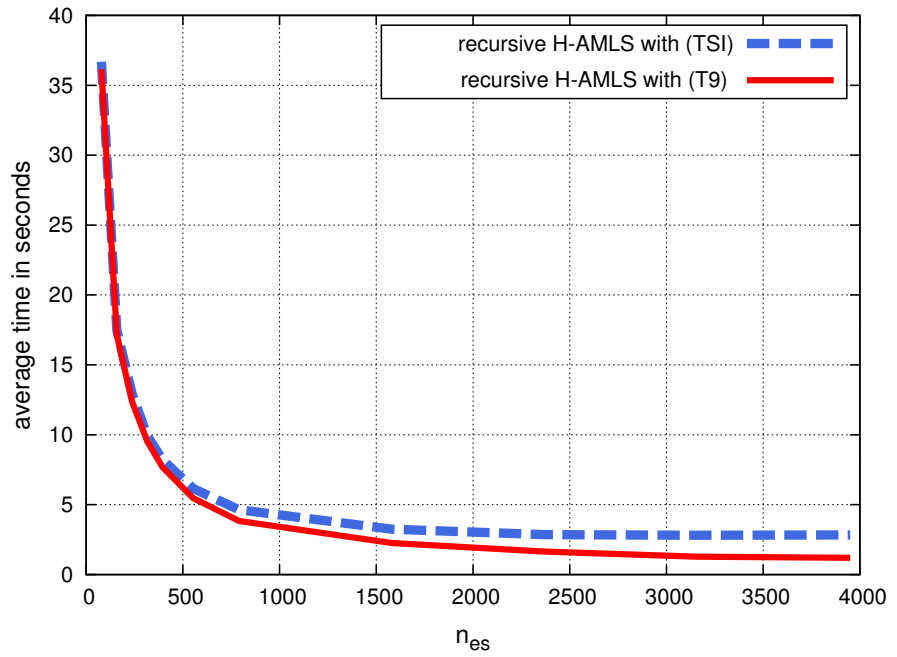


Figure 9.5.: The average time of recursive $\mathcal{H}$-AMLS corresponding to the benchmarks presented in Figure 9.4.

9.4. Parallel Performance of $\mathcal{H}$-AMLS

The $\mathcal{H}$-AMLS method has been parallelised for shared memory systems using Threading Building Blocks v4.2 [46] to provide thread parallelism. See Section 8.8 for the corresponding implementation and for further details. In the following the parallel performance of the $\mathcal{H}$-AMLS implementation is analysed, however, the discussion is restricted to the recursive version applying the improvement task ($\mathbb{T}$SI) and using the parameter setting described in Section 9.2 with $C_{\text{dom}}^{\approx} = 6$. The benchmarks presented in this section have been performed on the following two NUMA-architectures, both of which equipped with 32 cores.

System A: 4-socket system with Intel Xeon processors X7550 (Nehalem-microarchitecture, 2.00 GHz, 8 cores)

System B: 4-socket system with Intel Xeon processors E5-4640 (Sandy-Bridge-microarchitecture, 2.40 GHz, 8 Cores)

A basic difference between both NUMA systems is that system B is equipped with processors of the more modern *Intel Sandy-Bridge-microarchitecture* (which has replaced Intel's *Nehalem-microarchitecture*) and that system B has more memory bandwidth than system A.

In Figure 9.6 and Figure 9.7 the parallel performance of $\mathcal{H}$-AMLS (using the parameter setting described above) has been displayed computing the smallest $n_{\text{es}} = 5N_h^{1/3}$ eigenpairs for the mesh widths h_1, h_2, h_3 and h_4. Figure 9.6 displays the parallel speedup and the parallel efficiency on system A for up to 32 threads, and Figure 9.7 displays the corresponding results on system B. First of all it can be observed that the parallel performance of $\mathcal{H}$-AMLS on NUMA-system B is better than on system A for the larger problems associated to mesh widths h_3 and h_4. This effect is primarily caused by the von Neumann bottleneck (see Section 8.8) where the speedup of a parallel program is limited by the data transfer rate between the CPU cores and the shared memory. Since NUMA-system B is equipped with a higher memory bandwidth than system A, the parallel performance of $\mathcal{H}$-AMLS on this system is better. In particular the results show how strongly the speedup of $\mathcal{H}$-AMLS depends on the memory bandwidth and the NUMA-architecture. Furthermore, it can be observed in Figure 9.6 and Figure 9.7 that the parallel efficiency of $\mathcal{H}$-AMLS increases with increasing DOF, in particular that it is inefficient to apply $\mathcal{H}$-AMLS with many threads when the problem size is small. For example, the problem associated to mesh width h_1 is too small to benefit from a larger number of threads, as it can be seen in Table 9.11 the computational time using one thread (i.e., the sequential version of $\mathcal{H}$-AMLS is applied) is less than 6s on system B. For small problems the parallel overhead (see Section 8.8) is too large and the granularity of working tasks, which can be performed in parallel, is too small.

However, also on the more efficient NUMA-system B the implementation of $\mathcal{H}$-AMLS has for the largest problem, the problem associated to mesh width h_4, only a parallel speedup of around 16 when 32 threads are used. To investigate the parallel performance on system B in more detail the speedup of each involved task has been benchmarked separately for the mesh width h_4. Since in the parallel implementation of $\mathcal{H}$-AMLS in the condensation process different condensations (which are basically recursive calls of $\mathcal{H}$-AMLS) are performed concurrently it is not possible to measure the speedups concerning the tasks ($\mathbb{T}$6)–($\mathbb{T}$8) accumulatively. In contrast to the previous section, in the following the time measurements concerning tasks ($\mathbb{T}$6)–($\mathbb{T}$8) do not include the times of the recursive calls, instead the computational costs of the recursive call

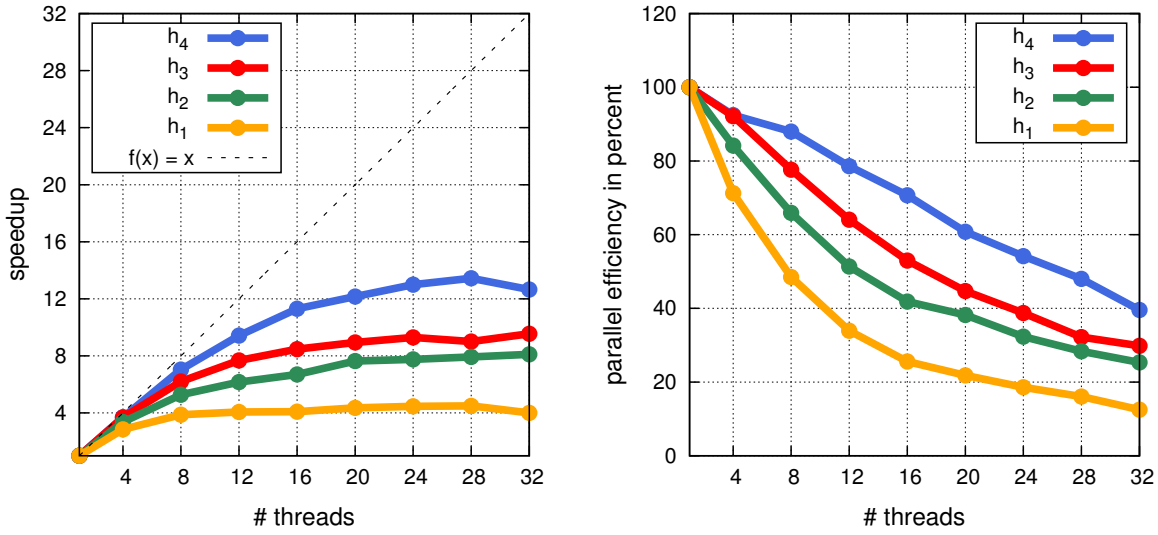


Figure 9.6.: Parallel performance on NUMA-system A of recursive $\mathcal{H}$-AMLS with (TSI)-improvement for computing the smallest $n_{\text{es}} = 5N_h^{1/3}$ eigenpairs for different mesh widths.

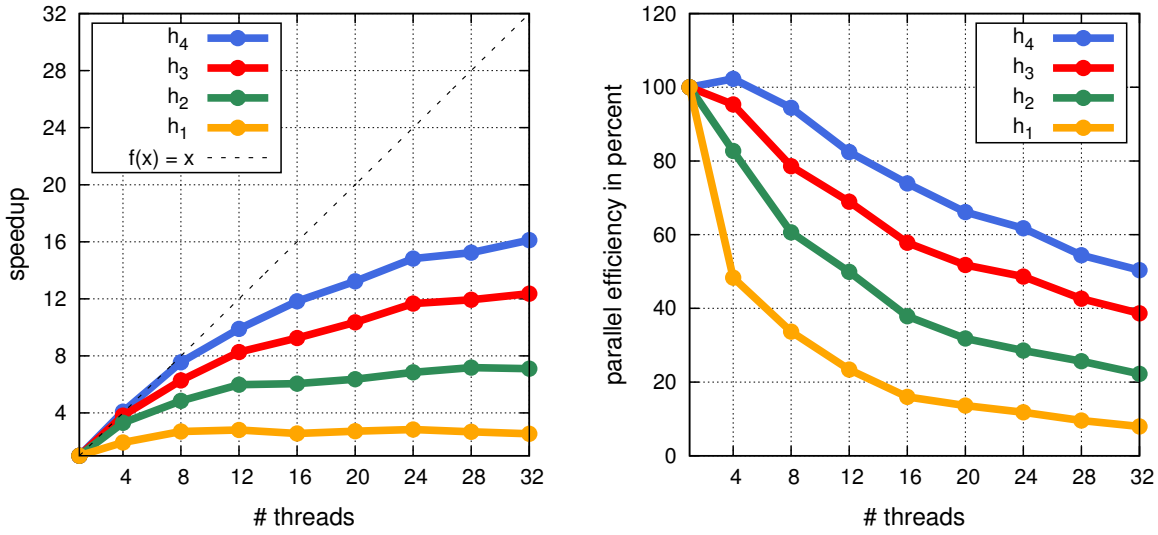


Figure 9.7.: Parallel performance on NUMA-system B of recursive $\mathcal{H}$-AMLS with (TSI)-improvement for computing the smallest $n_{\text{es}} = 5N_h^{1/3}$ eigenpairs for different mesh widths.

#threads	for h_1		for h_2		for h_3		for h_4	
p	t_{all}	$E(p)$	t_{all}	$E(p)$	t_{all}	$E(p)$	t_{all}	$E(p)$
1	5.47s	100.0%	90.75s	100.0%	1,495.67s	100.0%	25,094.71s	100.0%
8	2.03s	33.6%	18.71s	60.6%	237.97s	78.5%	3,321.88s	94.4%
16	2.14s	15.9%	14.98s	37.8%	161.66s	57.8%	2,122.20s	73.9%
24	1.93s	11.8%	13.25s	28.5%	128.20s	48.6%	1,693.65s	61.7%
32	2.15s	7.9%	12.77s	22.2%	121.00s	38.6%	1,556.72s	50.3%

Table 9.11.: Parallel performance on NUMA-system B of recursive $\mathcal{H}$-AMLS with (TSI)-improvement for computing the smallest $n_{\text{es}} = 5N_h^{1/3}$ eigenpairs for different mesh widths.

are included in the measured time of the condensation process, and the parallel performance of the condensation process has been measured on its own. In Figure 9.8 the separate speedup of each task [(T1)–(T8), (TSI), condensation process] is displayed, and in order to assess the speedup of each task regarding the total speedup of the method in Figure 9.9 the computational time of each task relative to the total computational time is displayed in dependence of the number p of used threads. As described in Section 8.8 task (T1) is always applied sequentially, since the performance of the task depends heavily on the memory bandwidth of the system and the performance did not benefit from a parallel implementation (more details in the following). Correspondingly the speedup of (T1) is equal to 1 for all $p \in \mathbb{N}$ and is not displayed in Figure 9.8. In order to evaluate the limitations of the parallel speedup on NUMA-system B due to the limited memory bandwidth the speedup of the following reference task is displayed as well in Figure 9.8.

Definition 9.1 (Reference Task) *The reference task associated to problem (K, M) with N degrees of freedom and n_{es} sought eigenpairs is given by computing the matrix vector product*

$$y_i := KMx \quad \text{for } i = 1, \ldots, n_{\text{es}} \qquad \text{where } x := (1, \ldots, 1)^T \in \mathbb{R}^N$$

where the matrix-vector multiplication is performed using the compressed row storage format *of the sparse matrices K and M. If the number of used threads is larger than 1 the computation of $(y_i)_{i=1}^{n_{\text{es}}}$ is performed in a parallel for loop using the corresponding TBB routine. In the following the reference task is referred to as* (TRef).

Task (TRef) is perfectly parallelisable when $n_{\text{es}} = kp$ for some $k \in \mathbb{N}$ or when $n_{\text{es}} \gg p$, however, the performance of the sparse matrix-vector multiplication is highly dependent on memory bandwidth.

In Figure 9.8 and Figure 9.9 the following observations can be made:

- As expected the speedup of the reference task (TRef) is limited. In particular, the speedup of (TRef) indicates in general the limitation of the parallel speedup on NUMA-system B for tasks whose parallel performance depends increasingly on the memory bandwidth of the system.

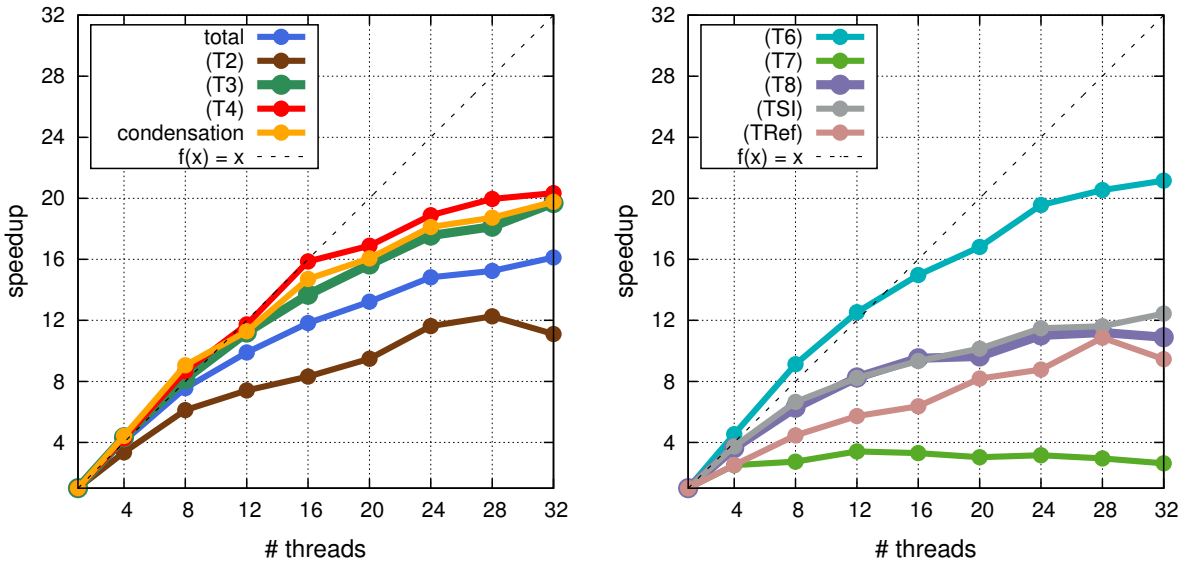


Figure 9.8.: Parallel speedup on NUMA-system B of recursive $\mathcal{H}$-AMLS with ($\mathbb{T}$SI)-improvement for computing the smallest $n_{\text{es}} = 5N^{1/3}$ eigenpairs of the problem associated to mesh width h_4. Displayed are the total speedup and the speedup of the involved tasks. To provide a better presentation the speedups are displayed in separate figures. Beside this, also the speedup of the reference task ($\mathbb{T}$Ref) from Remark 9.1 is shown.

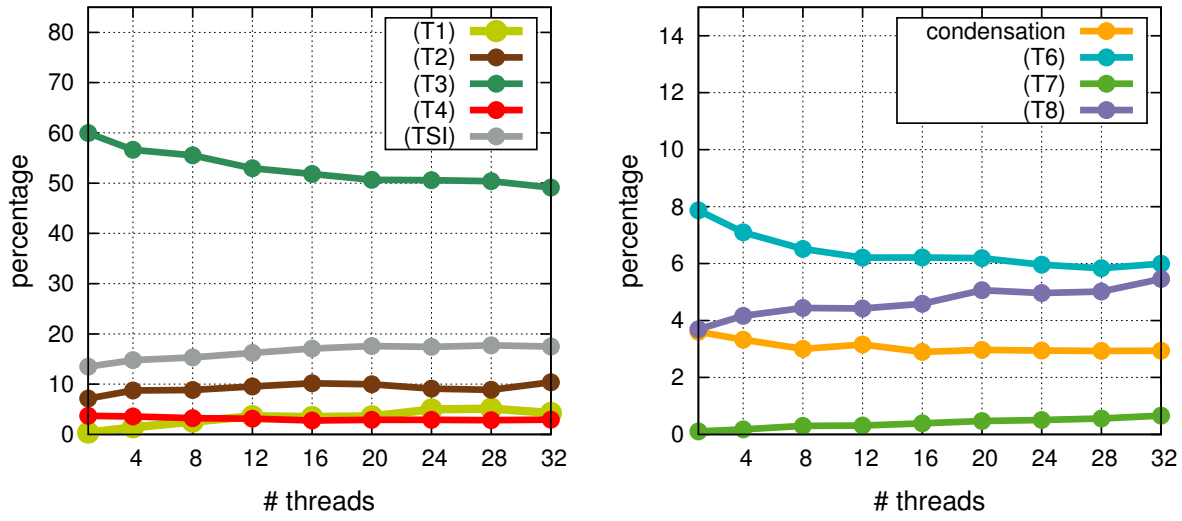


Figure 9.9.: Corresponding to the benchmark presented in Figure 9.8 the portion (in percent) of the computational time of each involved task in relation to the total computational time of $\mathcal{H}$-AMLS.

- Although task ($\mathbb{T}1$) is the only task which is applied sequentially (see Amdahl's Law in Section 8.8) its portion to the total computational time is even for 32 threads only around 10% and correspondingly limits the overall speedup only moderately. Benchmarks on NUMA-system B have shown that even when the construction of each $\mathcal{H}$-matrix representations $K^{\mathcal{H}}$ and $M^{\mathcal{H}}$ is performed concurrently by two threads then no speedup could be observed, which is due to the limited memory bandwidth of the system. More modern NUMA-systems with a higher memory bandwidth could overcome this issue so that task ($\mathbb{T}1$) benefits from multiple cores.

- The speedup of task ($\mathbb{T}2$), i.e, the computation of the $\mathrm{LDL}^{\mathrm{T}}$-factorisation $K^{\mathcal{H}} \approx L^{\mathcal{H}} \widetilde{K}^{\mathcal{H}} (L^{\mathcal{H}})^T$, is worse than most of the other tasks. However, the portion of task ($\mathbb{T}2$) to the total computational time is only around 10% and keeps nearly constant for varying number of used threads. As mentioned in Section 8.8.2 for the parallel computation of task ($\mathbb{T}2$) a HLIBpro routine is used which implements a task-based approach of the $\mathrm{LDL}^{\mathrm{T}}$-factorisation. This implementation, however, exhibits in HLIBpro v2.3 still a sequential part which can be parallelised and possibly increases the speedup (cf. [54]). Moreover, the $\mathcal{H}$-matrix format which is used for K can be replaced by the refined and improved $\mathcal{H}^2$-matrix format (cf. [17]) and the computation of $\widetilde{K}$ can be performed using the corresponding parallel $\mathcal{H}^2$-matrix arithmetic [19, 61] which is possibly leading to a general reduction of computational time of task ($\mathbb{T}2$) and to a better speedup.

- The speedup of task ($\mathbb{T}3$) is moderate, in particularly the task ($\mathbb{T}3$) clearly remains the computational most dominant part of $\mathcal{H}$-AMLS also when up to 32 threads are used. As described in Section 8.2 the computation of $\widetilde{M}^{\mathcal{H}} \approx (L^{\mathcal{H}})^{-1} M^{\mathcal{H}} (L^{\mathcal{H}})^{-T}$ is performed by solving triangular systems which is done by the corresponding parallel HLIBpro routines for $\mathcal{H}$-matrices. The solution of the triangular system is performed in a recursive approach applied to the block hierarchy of the $\mathcal{H}$-matrix structure and inherits a sequential bottleneck. However, a better parallel performance of the computation can be expected when the same task-based approach (with a global scope on the computational tasks) is implemented which is used for the parallel $\mathrm{LDL}^{\mathrm{T}}$-factorisation (cf. Section 8.8.2). Furthermore, as proposed for task ($\mathbb{T}2$), also in task ($\mathbb{T}3$) the refined and improved $\mathcal{H}^2$-matrix format can be used for the computation of $\widetilde{M}$ which is possibly leading to a general reduction of computational time of task ($\mathbb{T}3$) and to a better speedup.

- The speedup of task ($\mathbb{T}4$) is better than most of the remaining tasks, however, as already mentioned in Section 8.8.3, the speedup is partially limited by the speedup of the $\mathcal{H}$-SIL eigensolver which is applied to the large interface eigenvalue problems. Beside many small-sized subdomain and many small-sized interface eigenvalue problems (which create an even workload per thread) in task ($\mathbb{T}4$) also few large-sized interface eigenvalue problems have to be solved whose size is varying. Since the speedup of the $\mathcal{H}$-SIL eigensolver for a single subproblem is limited due to sequential parts of the Lanczos method, the overall speedup can be partially limited by the solution of one single large interface eigenvalue problem. This issue can be partially overcome by applying a scheduling which prioritises the processing of the large interface problems, and by improving the speedup of the $\mathcal{H}$-SIL eigensolver. Nonetheless, task ($\mathbb{T}4$) only makes a small portion of the total computational time.

- Task ($\mathbb{T}6$) shows the best parallel performance, however, an even better speedup is expected on shared memory system with a higher memory bandwidth. Task ($\mathbb{T}6$) is perfectly parallelisable and according to Section 8.8.4 the parallel implementation of task ($\mathbb{T}6$) provides a very high granularity of working tasks that can be performed concurrently.

- As described in Section 8.8.5 task ($\mathbb{T}7$) has been parallelised using the multi-threaded MKL version of the LAPACK eigensolver `dsygvx`. The speedup of task ($\mathbb{T}7$) is quite weak, however, a better speedup is expected when the size $\bar{k}$ of the $\mathcal{H}$-reduced problem is getting larger (in the benchmark $\bar{k}$ was equal to $4,483$). Nonetheless, the portion of task ($\mathbb{T}7$) to the total computational time is only minor also for 32 threads.

- Task ($\mathbb{T}8$) shows only a moderate speedup, nonetheless the portion of this task to the total computational time remains relative small even when 32 threads are used. Theoretically, task ($\mathbb{T}8$) is perfectly parallelisable, see Section 8.8.6 for the corresponding implementation. The majority of the computational costs of task ($\mathbb{T}8$) are caused by the computation of the matrix $\mathbf{S}_{\text{temp}}$ in (8.24). As already mentioned in Section 8.8.6 the efficiency of the matrix-wise approach for the parallel computation of $\mathbf{S}_{\text{temp}}$ depends on the column size k' of the block columns $\mathbf{S}_{\text{temp}}^{(l)} \in \mathbb{R}^{N \times k'}$. With the number of used threads the column size k' is decreasing and with it the efficiency of the matrix-wise approach for the computation of $\mathbf{S}_{\text{temp}}$. For example, using 32 threads we have $k' \approx 25$ while for one thread $k' = 795$. Correspondingly, if number n_{es} of sought eigenpairs is getting larger also a better speedup is expected.

- Task ($\mathbb{T}SI$) shows as well only a moderate speedup, although most of the individual working steps of task ($\mathbb{T}SI$) are perfectly parallelisable. The implementation of task ($\mathbb{T}SI$) is described in Algorithm 4 and its parallelisation in Section 8.8.7. The computational costs of task ($\mathbb{T}SI$) are dominated by the costs of the sparse matrix multiplications MS, KQ, MQ and by the costs of the matrix computations $A_2 := (L^{\mathcal{H}})^{-1}A_1$, $A_4 := (L^{\mathcal{H}})^{-T}A_3$ (cf. Section 8.8.7). The parallel performance of the sparse matrix multiplication, however, depends heavily on the memory bandwidth of the system which thus limits the speedup of task ($\mathbb{T}SI$). Furthermore, the matrices A_2 and A_4 are computed by the same parallel approach which is used for the matrix $\mathbf{S}_{\text{temp}}$ in task ($\mathbb{T}8$), and hence the parallel performance of their computation benefits as well from a larger number of sought eigenpairs.

- The parallelisation of the condensation process is described in Section 8.8.8. In the benchmark the condensation process shows a satisfying speedup and its portion to the total computational time is minor.

In summary, the implementation of $\mathcal{H}$-AMLS shows a reasonable parallel performance. However, better speedups are expected when $\mathcal{H}$-AMLS is applied to larger problems and, most importantly, when $\mathcal{H}$-AMLS is performed on more modern NUMA-architectures with improved memory performance. In particular, a better speedup is expected when the distribution of the data to the memory is optimised to the NUMA-architecture such that locality of memory access is guaranteed. Some tasks of $\mathcal{H}$-AMLS are perfectly parallelisable, however, the optimal speedup is not achieved due to the limited memory bandwidth of NUMA-system B and due to some parallel overhead [as described, e.g., for task ($\mathbb{T}8$)].

Furthermore, as noted above, a better speedup of $\mathcal{H}$-AMLS is expected when the parallel implementation of the $\mathcal{H}$-matrix routines used in task ($\mathbb{T}2$) and ($\mathbb{T}3$) is further improved, or

possibly when the used $\mathcal{H}$-matrix format is replaced by the refined $\mathcal{H}^2$-matrix format. Also the scheduling in ($\mathbb{T}4$) can be improved. Beyond that, the execution of the different tasks of $\mathcal{H}$-AMLS can be merged partly — for example, as soon as the eigensolutions of problem $(\widetilde{K}_{ii}^{\mathcal{H}}, \widetilde{M}_{ii}^{\mathcal{H}})$ and $(\widetilde{K}_{jj}^{\mathcal{H}}, \widetilde{M}_{jj}^{\mathcal{H}})$ have been computed the computation of submatrix $\widehat{\mathbf{M}}_{ij}$ can be started — which increases the parallel granularity of $\mathcal{H}$-AMLS even further and might get relevant for many-core shared memory systems with high memory bandwidth.

9.5. $\mathcal{H}$-AMLS for Challenging Problems

In the previous sections we analysed the $\mathcal{H}$-AMLS method for the Laplace eigenvalue problem on the unit cube. In this section additional numerical results are presented which investigate the approximation accuracy of the $\mathcal{H}$-AMLS method for elliptic PDE eigenvalue problems

$$\begin{cases} Lu = \lambda u & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega \end{cases} \tag{9.6}$$

with a more challenging domain Ω and a general PDE operator of the form

$$L[u](x) = -\operatorname{div}\big(A\nabla u\big)(x) + c(x)u(x) \quad \text{for all } x \in \Omega. \tag{9.7}$$

We consider the following three-dimensional elliptic PDE eigenvalue problems:

Example 1: Eigenvalue problem (9.6) with PDE operator $Lu = -\Delta u$ and domain

$$\Omega := (0,1)^3 \setminus [\tfrac{1}{2},1] \times [0,\tfrac{1}{2}] \times [\tfrac{1}{2},1] \quad \rightarrow \text{unit cube with internal corner, cf. Figure 9.10(a)}$$

Example 2: Eigenvalue problem (9.6) with PDE operator $Lu = -\Delta u$ and domain

$$\Omega := (0,1)^3 \setminus [\tfrac{1}{5},1] \times [\tfrac{2}{5},\tfrac{3}{5}] \times [0,1] \quad \rightarrow \text{unit cube with slit, cf. Figure 9.10(b)}$$

Example 3: Eigenvalue problem (9.6) with the same domain as in Example 2 and where the coefficients of the PDE operator L in (9.7) are of the form

$$A(x) = \begin{bmatrix} 3 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \text{ and } c(x) = 2 \qquad \forall\, x \in \Omega.$$

Example 4: Eigenvalue problem (9.6) with the same domain as in Example 2 and where the coefficients of the PDE operator L in (9.7) are of the form

$$A(x) = \begin{cases} B & \text{if } x \notin U \\ \mathrm{Id} & \text{if } x \in U \end{cases} \text{ and } c(x) = \begin{cases} 2 & \text{if } x \notin U \\ 0 & \text{if } x \in U \end{cases} \quad \text{with } B := \begin{bmatrix} 3 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \forall\, x \in \Omega$$

and where

$$U := \Big\{ x \in \mathbb{R}^d \,:\, x \in [k_1, k_1 + \tfrac{1}{10}) \times [k_2, k_2 + \tfrac{1}{10}) \times [k_3, k_3 + \tfrac{1}{10}) \text{ with } k_1, k_2, k_3 \in 2\mathbb{Z} \Big\}.$$

To solve the eigenvalue problems described in Example 1 – Example 4 by $\mathcal{H}$-AMLS or a classical approach, the problems have to be discretised. This is done by using the finite element space of piecewise affine functions $\mathbb{X}^1_{h,0}$ and where the triangulation of the corresponding domain is obtained by decomposing Ω into equispaced subintervals in each direction [cf. Figure 9.10(c)]. For the Laplace eigenvalue problem on the unit cube the approximation quality of $\mathcal{H}$-AMLS could be evaluated by comparing the relative errors of the $\mathcal{H}$-AMLS approximation and of the discretisation

$$\widehat{\boldsymbol{\delta}}_j := \underbrace{\frac{|\,\lambda_j - \widehat{\boldsymbol{\lambda}}_j^{(\mathrm{rq})}\,|}{\lambda_j}}_{\substack{\text{relative error of}\\ \mathcal{H}\text{-AMLS}}} \quad \text{and} \quad \widehat{\delta}_j^{(h)} := \underbrace{\frac{|\,\lambda_j - \lambda_j^{(h)}\,|}{\lambda_j}}_{\substack{\text{relative error}\\ \text{of discretisation}}}. \tag{9.8}$$

However, since for Example 1 – Example 4 the exact eigenvalues λ_j are not known, the eigenvalues λ_j in (9.8) are approximated by the discrete eigenvalues $\lambda_j^{(h_4)}$ associated to the finest mesh width h_4. Using these approximated relative errors the approximation quality of $\mathcal{H}$-AMLS has been benchmarked for Example 1 – Example 4 by investigating the corresponding maximal error ratios

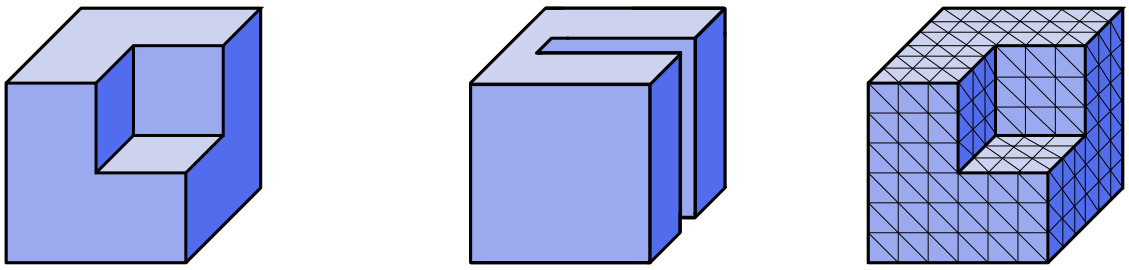
$$\gamma_{n_{\mathrm{es}}}^{(h)} := \max\left\{\widehat{\boldsymbol{\delta}}_j \,/\, \widehat{\delta}_j^{(h)} \;:\; j = 1, \ldots, n_{\mathrm{es}}\right\} \tag{9.9}$$

where the following setting has been used:

- The discrete eigenvalues $\lambda_j^{(h)}$ have been computed for $h = h_1, h_2, h_3$ and for the reference mesh width h_4 by the eigensolver $\mathcal{H}$-ARPACK. The relative residual errors[3] of the computed discrete eigenpairs has been smaller than 1e-9, so that they can be considered as numerically exact (as allowed by the finite element discretisation). However, due to computational limits it was only possible to compute for the mesh width h_4 with $\mathcal{H}$-ARPACK the smallest 267 discrete eigenpairs.
- The $\mathcal{H}$-AMLS method has been applied in the recursive version with ($\mathbb{T}$SI)-improvement and $C_{\mathrm{dom}}^{\approx} = 6$ for the computation of the smallest $n_{\mathrm{es}} = 3N_h^{1/3}$ discrete eigenpairs of the problems associated to the mesh widths h_1, h_2, h_3.

The results of these benchmarks are summarised in Table 9.12 and show that the approximation quality of $\mathcal{H}$-AMLS is comparable to the approximation quality of a classical approach.

[3] We define the *relative residual error* of an eigenpair approximation (λ, x) of (K, M) by $\|Kx - \lambda Mx\|_2 / \|Kx\|_2$.



(a) Unit cube with internal corner $\Omega = (0,1)^3 \setminus [\frac{1}{2},1] \times [0,\frac{1}{2}] \times [\frac{1}{2},1]$.

(b) Unit cube with slit $\Omega = (0,1)^3 \setminus [\frac{1}{5},1] \times [\frac{2}{5},\frac{3}{5}] \times [0,1]$.

(c) Triangulation of domain from Figure 9.10(a)

Figure 9.10.: The underlying domains and associated triangulations of Example 1 – 4.

	Example 1			Example 2			Example 3			Example 4		
	N_h	n_{es}	$\gamma_{n_{\text{es}}}^{(h)}$	N_h	n_{es}	$\gamma_{n_{\text{es}}}^{(h)}$	N_h	n_{es}	$\gamma_{n_{\text{es}}}^{(h)}$	N_h	n_{es}	$\gamma_{n_{\text{es}}}^{(h)}$
h_1	5,859	54	1.14	5,339	51	1.16	5,339	51	1.43	5,339	51	1.40
h_2	51,319	111	1.10	48,087	108	1.60	48,087	108	1.23	48,087	108	1.13
h_3	429,039	225	1.16	407,087	222	1.08	407,087	222	1.19	407,087	222	1.12

Table 9.12.: Approximation accuracy of the recursive $\mathcal{H}$-AMLS with (TSI)-improvement for computing the smallest $n_{\text{es}} = 3N_h^{1/3}$ eigenpairs with varying mesh widths.

10. Conclusion

To solve an elliptic PDE eigenvalue problem in practice typically the finite element discretisation is used. From approximation theory it is known that only the smaller eigenvalues λ_j and their corresponding eigenfunctions u_j can be well approximated by the finite element discretisation because the approximation error increases with increasing size of the eigenvalue. However, results on the number of well approximable eigenvalues or eigenfunctions are not available in literature (to the best of the author's knowledge). In this work asymptotic estimates on these quantities could be derived. For example, it is shown that for three-dimensional problems under certain smoothness assumptions on the data only the first $\Theta(N^{2/5})$ eigenvalues and only the first $\Theta(N^{1/4})$ eigenfunctions can be well approximated by the finite element discretisation using N-dimensional finite element spaces of piecewise affine functions with uniform mesh refinement.

To solve the discretised elliptic PDE eigenvalue problem and to compute all well approximable eigenvalues and eigenfunctions, in this work a recursive version of the automated multilevel substructuring has been combined with the concept of hierarchical matrices. Whereas the classical AMLS method is very efficient for two-dimensional problems, it is getting very expensive in the three-dimensional case. One computational bottleneck of classical AMLS is in the three-dimensional case the required computation of the transformed eigenvalue problem $(\widetilde{K}, \widetilde{M})$. Using the fast $\mathcal{H}$-matrix arithmetic, however, the transformed problem can be computed very efficiently in almost linear complexity $\mathcal{O}(N \log^{\alpha} N)$ which is even independent of the number of sought eigenpairs. Also the solution of the interface eigenvalue problems $(\widetilde{K}_{ii}, \widetilde{M}_{ii})$ and the computation of the reduced eigenvalue problem $(\widehat{K}, \widehat{M})$ are performed much more efficiently using the fast $\mathcal{H}$-matrix arithmetic. Moreover, the new recursive version of AMLS allows us to bound the size of the reduced eigenvalue problem $(\widehat{K}, \widehat{M})$ which substantially further reduces the costs for its computation and solution. Overall, this leads to the new $\mathcal{H}$-AMLS method which is well suited for three-dimensional problems and which allows us to compute a large amount of eigenpair approximations in optimal complexity. Furthermore, $\mathcal{H}$-AMLS is well parallelisable and shows in numerical experiments a satisfying parallel performance.

The $\mathcal{H}$-AMLS method has to be benchmarked in further examples, especially for problems arising from applications. The numerical results presented in this work, however, demonstrate the potential of the method in solving large-scale elliptic PDE eigenvalue problems. Furthermore, a black box approach allows $\mathcal{H}$-AMLS to be applied in a purely algebraic way without the need of geometry information. Beside that, $\mathcal{H}$-AMLS can be combined very efficiently with a subsequent subspace iteration in order to improve the accuracy of the $\mathcal{H}$-AMLS eigenpair approximations when needed. These aspects possibly allow $\mathcal{H}$-AMLS to become applicable to a wide range of problems.

A. Abstract Variational Eigenvalue Problems

The elliptic PDE eigenvalue problem (2.1) is analysed in Chapter 2. The corresponding existence result of (weak) eigensolutions which is presented in Theorem 2.6 is based on this section which provides a general analysis of abstract variational eigenvalue problems of the form

$$\begin{cases} \text{find } (\lambda, u) \in \mathbb{C} \times H \setminus \{0\} \text{ such that} \\ a(u,v) \;=\; \lambda\, d(u,v) \qquad \forall\, v \in H \end{cases} \tag{A.1}$$

where $a(\cdot,\cdot)$ and $d(\cdot,\cdot)$ are bilinear forms defined on a Hilbert space H. In this section it is shown that under certain assumptions on the bilinear forms the eigenvalue problem (A.1) possesses a countable family of eigensolutions. The key in the corresponding existence proof is to reformulate the variational eigenvalue problem (A.1) as an eigenvalue problem of a linear operator, and to show that this operator is compact. The Fredholm-Riesz-Schauder theory allows to characterise the spectrum of that compact operator, which finally proves the existence of a countable family of eigensolutions of the variational eigenvalue problem (A.1).

The remainder of this section is organised as follows: In Section A.1 basic definitions from functional analysis are recalled and the mathematical framework for compact operators is stated, and in Section A.2 the Fredholm-Riesz-Schauder theory for compact operators is formulated. The statements made in Section A.1 and Section A.2 are basic results from functional analysis and can be found, e.g., in [43] or any textbook on this topic. In Section A.3 the variational eigenvalue problem (A.1) is reformulated as an eigenvalue problem of an operator and it is proven by the author that, under certain assumptions, this operator is compact. As described above, this finally proves the existence result for eigensolutions of problem (A.1).

A.1. Basic Definitions and Compact Operators

In this section basic results from functional analysis are recalled and the mathematical framework for compact operators is stated. The following discussion is restricted to real-valued linear spaces.

Definition A.1 (Normed Space) *Let X be a linear space (vector space) over the field $\mathbb{R}$. A mapping $\|\cdot\| : X \to [0,\infty)$ is called a norm if it fulfils*

$$\begin{aligned} \forall\, x \in X: &\quad \|x\| = 0 \Rightarrow x = 0, \\ \forall\, \alpha \in \mathbb{R}, x \in X: &\quad \|\alpha x\| = |\alpha| \|x\|, \\ \forall\, x, y \in X: &\quad \|x + y\| \le \|x\| + \|y\|. \end{aligned}$$

The linear space X associated with the norm $\|\cdot\|$ is called normed space and is identified by the pair $(X, \|\cdot\|)$. If the underlying space X is not clear the notation $\|\cdot\|_X$ is used. On a linear space several different norms can be defined. Two norms $\|\cdot\|_1$ and $\|\cdot\|_2$ on X are called equivalent if a constant $C > 0$ exists such that

$$\frac{1}{C}\|x\|_1 \le \|x\|_2 \le C\|x\|_1 \quad \forall\, x \in X.$$

Definition A.2 (Bounded Operator) *Let X, Y be normed spaces. A linear mapping $T : X \to Y$ is called operator. The operator T is called bounded or continuous if it holds*

$$\|T\|_{Y \leftarrow X} := \sup \left\{ \frac{\|Tx\|_Y}{\|x\|_X} \ : \ x \in X \setminus \{0\} \right\} < \infty$$

where $\|\cdot\|_{Y \leftarrow X}$ is the corresponding operator norm. The set of all bounded operators $T : X \to Y$ is denoted by $L(X, Y)$. In the case of $X = Y$ we simply write $L(X)$ instead of $L(X, X)$. With the addition operation and the scalar multiplication

$$(T_1 + T_2)x := T_1 x + T_2 x, \quad (\alpha T_1)x := T_1(\alpha x) \quad \forall\, x \in X, \alpha \in \mathbb{R} \text{ and } T_1, T_2 \in L(X, Y) \tag{A.2}$$

the set of all bounded operators $L(X, Y)$ forms a linear space, and together with the operator norm $\|\cdot\|_{Y \leftarrow X}$ it is a normed space. Introducing the multiplication operation

$$(T_1 T_2)x := T_1(T_2 x) \quad \forall\, x \in X \text{ and } T_1, T_2 \in L(X)$$

the space $L(X)$ constitutes an algebra with the identity (neutral) element

$$I \in L(X) \quad \text{with} \quad I(x) := x \quad \forall\, x \in X.$$

In this context $T^n \in L(X)$ denotes for $n \in \mathbb{N}$ the n-times operator composition of $T \in L(X)$.

Definition A.3 (Banach Space) *A sequence $(x_n)_{n \in \mathbb{N}} \subset X$ is called Cauchy convergent or Cauchy sequence if it holds*

$$\sup \Big\{ \|x_n - x_m\| \ : \ n, m \geq k \Big\} \xrightarrow{k \to \infty} 0.$$

A normed space X is called complete if every Cauchy sequence $(x_n)_{n \in \mathbb{N}}$ converges to an element $x \in X$, i.e., it holds $\lim_{n \to \infty} \|x_n - x\| = 0$. A normed and complete space is called Banach space.

Theorem A.4 (Bounded Inverse) *Let X, Y be Banach spaces and $T \in L(X, Y)$. If T is bijective then it holds $T^{-1} \in L(Y, X)$.*

Definition A.5 (Dual Space) *Let X be a normed linear space over $\mathbb{R}$. The dual space X' of X is the space of all linear and bounded mappings, i.e., it holds $X' := L(X, \mathbb{R})$. The dual space X' is a Banach space with the norm*

$$\|x'\|_{X'} := \|x'\|_{\mathbb{R} \leftarrow X} = \sup \left\{ \frac{|x'(x)|}{\|x\|_X} \ : x \in X \setminus \{0\} \right\}.$$

An element $x' \in X'$ is called linear functional on X.

Definition A.6 (Precompact, Compact) *Let X be a Banach space. A subset $M \subset X$ is called precompact if every sequence $(x_n)_{n \in \mathbb{N}} \subset M$ has a convergent subsequence $(x_{n_i})_{i \in \mathbb{N}}$, i.e, there exists an element $x \in X$ such that $\lim_{i \to \infty} \|x_{n_i} - x\|_X = 0$. The subset M is called compact if it additionally holds $x \in M$.*

Definition A.7 (Compact Operator) *Let X, Y be Banach spaces. A bounded operator $T \in L(X,Y)$ is called compact if the set*

$$\Big\{Tx \ : \ x \in X \text{ with } \|x\|_X \leq 1\Big\}$$

is precompact in Y. The set of all compact operators from X into Y is denoted by $K(X,Y)$ and in the case of $X = Y$ we write $K(X)$ instead of $K(X,X)$.

Definition A.8 (Embedding) *Let X, Y be Banach spaces with $X \subset Y$. The linear mapping*

$$I : X \to Y \quad \text{with} \quad Ix := x \quad \forall\, x \in X$$

is called embedding of X in Y. If it holds $I \in L(X,Y)$, i.e., there exists a constant $C_B > 0$ such that $\|x\|_Y \leq C_B \|x\|_X$ for all $x \in X$, we call X continuously embedded in Y and formally write $X \hookrightarrow Y$. If it holds $I \in K(X,Y)$ the embedding is called compact. If X is dense[1] in Y the embedding I is called dense.

A.2. The Fredholm-Riesz-Schauder Theory

In this section the spectrum of bounded operators is introduced. The spectrum generalises the concept of eigenvalues of matrices and is used for the characterisation of operators. We say a complex number $\lambda \in \mathbb{C}$ is in the spectrum of an operator $T \in L(X)$ iff the mapping $T - \lambda I$ is not bijective. If the underlying linear space X is finite-dimensional the operator T can be described by a matrix, and the spectrum and the set of eigenvalues are the same. In the infinite-dimensional case, however, not every spectral value has to be an eigenvalue and a more detailed distinction is made.

Throughout the whole section it is assume that $(X, \|\cdot\|)$ is a Banach space with $X \neq \{0\}$.

Definition A.9 (Resolvent Set, Spectrum) *The resolvent set $\rho(T)$ of an operator $T \in L(X)$ is defined by*

$$\rho(T) := \Big\{\lambda \in \mathbb{C} \ : \ \mathrm{Ker}(T - \lambda I) = \{0\} \wedge \mathrm{Im}(T - \lambda I) = X\Big\},$$

where $\mathrm{Ker}(\cdot)$ is the kernel and $\mathrm{Im}(\cdot)$ is the image of an operator. The spectrum of T is defined by $\sigma(T) := \mathbb{C} \setminus \rho(T)$ where it is distinguished between the point spectrum $\sigma_p(T)$, the residual spectrum $\sigma_r(T)$ and the continuous spectrum $\sigma_c(T)$ which are defined by

$$\sigma_p(T) := \Big\{\lambda \in \sigma(T) \ : \ \mathrm{Ker}(T - \lambda I) \neq \{0\}\Big\}, \tag{A.3}$$

$$\sigma_r(T) := \Big\{\lambda \in \sigma(T) \ : \ \mathrm{Ker}(T - \lambda I) = \{0\} \wedge \overline{\mathrm{Im}(T - \lambda I)} \neq X\Big\}, \tag{A.4}$$

$$\sigma_c(T) := \Big\{\lambda \in \sigma(T) \ : \ \mathrm{Ker}(T - \lambda I) = \{0\} \wedge \overline{\mathrm{Im}(T - \lambda I)} = X \wedge \mathrm{Im}(T - \lambda I) \neq X\Big\}. \tag{A.5}$$

The statement that it holds $\lambda \in \rho(T)$ is equivalent to the statement that the operator $T - \lambda I : X \to X$ is bijective, i.e., that the inverse $(T - \lambda I)^{-1}$ exists. Because of Theorem A.4 we have

$$R_\lambda(T) := (T - \lambda I)^{-1} \in L(X)$$

[1] A subset $X \subset Y$ is called dense in a normed space Y when for each $y \in Y$ there exists some sequence $(x_n)_{n\in\mathbb{N}} \subset X$ with $\lim_{n\to\infty} \|x_n - y\|_Y = 0$.

where the operator $R_\lambda(T)$ is called the *resolvent*[2] of T. If it holds instead that $\lambda \in \sigma(T)$ then the spectral value λ can be assigned to one of the three distinct types (A.3) – (A.5). In the following discussion, however, we are only interested in the point spectrum $\sigma_p(T)$ where we have

$$\lambda \in \sigma_p(T) \quad \Longleftrightarrow \quad \exists\, u \in X \setminus \{0\} \text{ with } Tu = \lambda u. \tag{A.6}$$

Definition A.10 (Eigenvalue, Eigenvector) *If it holds $\lambda \in \sigma_p(T)$ then λ is called eigenvalue of T and all $u \in X \setminus \{0\}$ fulfilling $Tu = \lambda u$ are called eigenvectors. The eigenspace of λ is defined by*

$$E(\lambda) := \operatorname{Ker}(T - \lambda I)$$

and $\dim E(\lambda)$ is called the multiplicity of λ.

Basic results on the spectrum are summarised in the following remark.

Remark A.11 *Let $T \in L(X)$ be a continuous operator. Then it holds:*

i) The eigenspace $E(\lambda) \subset X$ of an eigenvalue $\lambda \in \sigma_p(T)$ is T-invariant[3].

ii) The spectrum $\sigma(T) \subset \mathbb{C}$ is compact and non-empty. In particular, it holds

$$r(T) := \sup_{\lambda \in \sigma(T)} |\lambda| = \lim_{n \to \infty} \|T^n\|_{X \leftarrow X}^{1/n} \leq \|T\|_{X \leftarrow X}$$

where $r(T)$ is called the spectral radius of T.

iii) If $\dim X < \infty$ then it holds $\sigma(T) = \sigma_p(T)$.

iv) If $\dim X = \infty$ and the operator T is compact then it holds $0 \in \sigma(T)$.

The following important theorem characterises the spectrum of compact operators.

Theorem A.12 (Riesz-Schauder I) *Let $T \in K(X)$ be a compact operator. Then the following statements are valid:*

i) It holds $\sigma(T) \setminus \{0\} \subseteq \sigma_p(T)$. More precisely, $\sigma(T) \setminus \{0\}$ consists of countably many (finitely or infinitely many) eigenvalues with zero as the only possible accumulation point.

ii) For $\lambda \in \sigma(T) \setminus \{0\}$ it holds

$$1 \leq n_\lambda := \max\Big\{ n \in \mathbb{N} \;:\; \operatorname{Ker}((T - \lambda I)^n) \neq \operatorname{Ker}((T - \lambda I)^{n-1}) \Big\} < \infty$$

where n_λ is called the index of λ.

iii) For $\lambda \in \sigma(T) \setminus \{0\}$ the space X can be decomposed into the direct sum

$$\operatorname{Ker}((T - \lambda I)^{n_\lambda}) \oplus \operatorname{Im}((T - \lambda I)^{n_\lambda})$$

where both subspaces are closed and T-invariant, and where $\dim \operatorname{Ker}((T - \lambda I)^{n_\lambda}) < \infty$.

iv) For all $\lambda, \mu \in \sigma(T) \setminus \{0\}$ it holds

$$P_\lambda P_\mu \;=\; \begin{cases} P_\lambda & \text{if } \lambda = \mu, \\ 0 & \text{if } \lambda \neq \mu \end{cases}$$

where $P_\lambda \in L(X)$ is the projection[4] onto $\operatorname{Ker}((T - \lambda I)^{n_\lambda})$ according to the decomposition

[2] In literature the definition $R_\lambda(T) := (\lambda I - T)^{-1}$ can be found as well which only changes the sign of the operator.

[3] A subset $M \subset X$ is called T-invariant if $T(x) \in M$ for all $x \in M$.

[4] Let X be a Banach space, then an operator $T \in L(X)$ is called projection if it holds $T^2 = T$.

described in iii).

Remark A.13 (Fredholm Alternative) *Statement i) of Theorem A.12 is also known as the Fredholm alternative: For any operator $T \in K(X)$ and for $\lambda \in \mathbb{C} \setminus \{0\}$ it* **either** *holds*

$$\forall\, y \in X\ \exists! x \in X \text{ with } Tx - \lambda x = y \quad \textbf{or} \quad \exists x \in X \setminus \{0\} \text{ with } Tx - \lambda x = 0.$$

The statements made in Theorem A.12 can be strengthened when compact operators are considered which are defined on Hilbert spaces and which are selfadjoint. As it can be seen in Theorem A.16, these operators can be completely described by their eigenvalues and eigenvectors.

Definition A.14 (Inner Product, Hilbert Space) *Let X be a linear space. A mapping $(\cdot,\cdot) : X \times X \to \mathbb{R}$ is called inner product on X if it holds*

$$\begin{aligned} \forall\, x \in X \setminus \{0\} : &\quad (x,x) > 0, \\ \forall\, x,y,z \in X, \alpha \in \mathbb{R} : &\quad (\alpha x + y, z) = \alpha\,(x,z) + (y,z), \\ \forall\, x,y \in X : &\quad (x,y) = (y,x). \end{aligned}$$

A Banach space X is called Hilbert space, if an inner product $(\cdot,\cdot)$ is defined on X with $\|x\| = (x,x)^{1/2}$.

Definition A.15 (Selfadjoint Operator) *Let X be a Hilbert space. An operator $T \in L(X)$ is called selfadjoint (or symmetric) if*

$$(Tx, y) = (x, Ty) \quad \forall\, x,y \in X.$$

A selfadjoint operator is called positive definite if it additionally holds

$$(x, Tx) > 0 \quad \forall\, x \in X \setminus \{0\}.$$

Theorem A.16 (Riesz-Schauder II) *Let X be a Hilbert space and let $T \in K(X)$ be selfadjoint with $T \neq 0$. Then it holds:*

i) All eigenvalues of T are real, i.e., we have $\sigma_p(T) \subset \mathbb{R}$. Furthermore, there exists an eigenvalue $\lambda \in \sigma_p(T)$ with $|\lambda| = \|T\|_{X \leftarrow X}$.

ii) For all $\lambda \in \sigma_p(T)$ it holds $n_\lambda = 1$.

iii) The eigenspaces $E(\lambda)$ and $E(\mu)$ are orthogonal for eigenvalues $\lambda, \mu \in \sigma_p(T)$ with $\lambda \neq \mu$.

iv) Let $\{e_{\lambda,i}\}_{i=1}^{d(\lambda)}$ be an orthonormal basis of eigenspace $E(\lambda)$ for eigenvalue $\lambda \in \sigma_p(T) \setminus \{0\}$ where $d(\lambda) := \dim E(\lambda)$. Then the operator T is described by

$$Tx \;=\; \sum_{\lambda \in \sigma_p(T) \setminus \{0\}} \lambda \left(\sum_{i=1}^{d(\lambda)} (x\,,\, e_{\lambda,i})\; e_{\lambda,i} \right) \qquad \text{for all } x \in X.$$

v) An orthogonal decomposition of X is given by

$$\overline{\bigoplus_{\lambda \in \sigma_p \setminus \{0\}} E(\lambda)} \;\oplus\; \operatorname{Ker}(T). \tag{A.7}$$

vi) If T is additionally positive definite then it holds $\sigma_p(T) \subset \big(0, \|T\|_{X \leftarrow X}\big]$. In particular, $\|T\|_{X \leftarrow X}$ is an eigenvalue of T.

A.3. Analysis of Abstract Variational Eigenvalue Problems

In this section it is discussed under which assumptions the variational eigenvalue problem (A.1) possesses eigensolutions. For this purpose, eigenvalue problem (A.1) is reformulated as an eigenvalue problem of an operator. Under certain assumptions on the involved bilinear forms in (A.1) it can be shown that this operator is compact. Hence, the Fredholm-Riesz-Schauder theory from the previous section can be applied to characterise the spectrum of this compact operator which finally constitutes the existence of countably many eigensolutions of problem (A.1).

Throughout the whole section it is assumed that H is a Banach space with $H \neq \{0\}$.

Definition A.17 (Bilinear Form) *A mapping $a(\cdot,\cdot) : H \times H \to \mathbb{R}$ is called bilinear form if for all $\alpha \in \mathbb{R}$ and for all $x, y, z \in H$ it holds*

$$a(x + \alpha y, z) = a(x, z) + \alpha\, a(y, z) \quad \text{and} \quad a(x, y + \alpha z) = a(x, y) + \alpha\, a(x, z).$$

Bilinear forms can have the following properties:

- *$a(\cdot,\cdot)$ is called symmetric if it holds $a(x, y) = a(y, x)$ for all $x, y \in H$.*
- *$a(\cdot,\cdot)$ is called continuous or bounded on $H \times H$ if there exists a constant $C_B > 0$ such that*
$$|a(x, y)| \leq C_B \|x\| \|y\| \quad \forall\, x, y \in H.$$
- *$a(\cdot,\cdot)$ is called H-elliptic[5], or short elliptic, if $a(\cdot,\cdot)$ is continuous on $H \times H$ and if there exists a constant $C_E > 0$ such that*
$$a(x, x) \geq C_E \|x\|^2 \quad \forall\, x \in H.$$

Theorem A.18 (Lax-Milgram) *Let $l \in H'$ be a continuous linear functional and let $a(\cdot,\cdot) : H \times H \to \mathbb{R}$ be a symmetric elliptic bilinear form. Then the functional*

$$J : H \to \mathbb{R} \quad \text{with} \quad J(v) := \tfrac{1}{2} a(v, v) - l(v) \quad \forall\, v \in H$$

has a unique minimizer $u \in H$. This minimizer is the unique solution of the variational problem

$$\begin{cases} \text{find } \; u \in H \; \text{ such that} \\ a(u, v) \; = \; l(v) \quad \forall\, v \in H. \end{cases} \tag{A.8}$$

Based on problem (A.8) the variational eigenvalue problem

$$\begin{cases} \text{find } \; (\lambda, u) \in \mathbb{C} \times H \setminus \{0\} \; \text{ such that} \\ a(u, v) \; = \; \lambda\, d(u, v) \qquad \forall\, v \in H. \end{cases} \tag{A.9}$$

is introduced where $d(\cdot,\cdot) : H \times H \to \mathbb{R}$ is a bilinear form fulfilling the following assumptions:

Precondition A.19 *i) $d(\cdot,\cdot)$ is symmetric and $d(u, u) > 0$ for all $u \in H \setminus \{0\}$.*

[5] In [23] a symmetric elliptic bilinear form is also referred to as coercive. However, in literature coercivity often describes a property which is weaker than ellipticity (see, e.g., [37]).

ii) $d(\cdot,\cdot)$ *is continuous on* $H \times H$*, i.e., it holds* $\|d\|_{\mathbb{R}\leftarrow H\times H} =: C_d < \infty$*.*

iii) For all sequences $(u_j)_{j\in\mathbb{N}} \subset H$ *with* $\|u_j\|_H \leq C$ *for all* $j \in \mathbb{N}$ *and some* $C > 0$ *there exists a subsequence* $(u_{j_k})_{k\in\mathbb{N}}$ *which is Cauchy w.r.t. the norm induced by* $d(\cdot,\cdot)^{1/2}$*.*

Theorem A.18 constitutes the existence of a unique solution of the variational problem (A.8). In the following an analogue result is derived for the variational eigenvalue problem (A.9). For this purpose the variational problem

$$\begin{cases} \text{find } u_f \in H \text{ such that} \\ a(u_f, v) \;=\; d(f, v) \quad \forall\, v \in H. \end{cases} \tag{A.10}$$

is introduced for some given $f \in H$. Since $d(f,\cdot) : H \to \mathbb{R}$ is a continuous linear functional for all $f \in H$ we conclude from Theorem A.18 that problem (A.10) possesses a unique solution $u_f \in H$ for all $f \in H$. This induces the solution operator

$$T : H \to H \quad \text{with} \quad f \mapsto u_f := Tf. \tag{A.11}$$

Lemma A.20 *Let the bilinear form* $a(\cdot,\cdot)$ *be symmetric elliptic and let the bilinear form* $d(\cdot,\cdot)$ *satisfy the assumptions made in Precondition A.19. Then the solution operator* T *in* (A.11) *is compact.*

Proof: Before it is proved that T is compact, it has to be proved that the solution operator is linear and bounded on H.

T is linear:
Let $u, f \in H$ and $u_f := Tf$, $u_g := Tg$, $u_{f+g} := T(f+g)$ then it holds

$$a(u_f + u_g, v) = a(u_f, v) + a(u_g, v) = d(f, v) + d(g, v) = d(f + g, v) \quad \forall\, v \in H.$$

Since the solution of problem (A.10) is unique we obtain $u_{f+g} = u_f + u_g$ and correspondingly $T(f+g) = Tf + Tg$. Analogical, we obtain for $\alpha \in \mathbb{R}$ and $u_{\alpha f} := T(\alpha\, f)$ because of

$$a(\alpha\, u_f, v) = \alpha\, a(u_f, v) = \alpha\, d(f, v) = d(\alpha\, f, v) \quad \forall\, v \in H$$

that $u_{\alpha f} = \alpha u_f$ is valid, i.e., it holds $T(\alpha f) = \alpha\, Tf$.

T is continuous:
For all $f \in H$ it holds

$$\|Tf\|^2 \underset{\substack{a(\cdot,\cdot) \text{ is} \\ \text{H-elliptic}}}{\leq} \frac{1}{C_E} a(Tf, Tf) \underset{\text{(A.10)}}{=} \frac{1}{C_E} d(f, Tf) \underset{\substack{d(\cdot,\cdot) \text{ is} \\ \text{continuous}}}{\leq} \frac{C_d}{C_E} \|f\| \|Tf\|.$$

Correspondingly, we have $\|Tf\| \leq \frac{C_d}{C_E}\|f\|$ for all $f \in H$, i.e., it holds $T \in L(H)$.

T is compact:
It remains to prove that the subset

$$B := \Big\{ Tf \;:\; f \in H \text{ with } \|f\|_H \leq 1 \Big\}$$

is precompact in H. Consider for this purpose a sequence $(Tf_i)_{i\in\mathbb{N}} \subset B$, then it holds

$$\begin{aligned}
\|Tf_i - Tf_j\|_H^2 &\underset{\substack{T \text{ is}\\ \text{linear}}}{=} \|T(f_i - f_j)\|_H^2 \underset{\substack{a(\cdot,\cdot) \text{ is}\\ \text{H-elliptic}}}{\leq} \frac{1}{C_E}\, a\big(T(f_i - f_j), T(f_i - f_j)\big) \\
&\underset{\text{(A.10)}}{=} \frac{1}{C_E}\, d\big(f_i - f_j, T(f_i - f_j)\big) \\
&\underset{\substack{\text{Precondition}\\ \text{A.19 i)}}}{\leq} \frac{1}{C_E}\, d\big(f_i - f_j, f_i - f_j\big)^{1/2}\, d\big(T(f_i - f_j), T(f_i - f_j)\big)^{1/2} \\
&\underset{\substack{\text{Precondition}\\ \text{A.19 ii)}}}{\leq} \frac{(C_d)^{1/2}}{C_E}\, d\big(f_i - f_j, f_i - f_j\big)^{1/2}\, \|T(f_i - f_j)\|_H \\
&\underset{\substack{T \text{ is}\\ \text{continuous}}}{\leq} \frac{(C_d)^{3/2}}{(C_E)^2}\, d\big(f_i - f_j, f_i - f_j\big)^{1/2}\, \|f_i - f_j\|_H \\
&\leq \frac{(C_d)^{3/2}}{(C_E)^2}\, d\big(f_i - f_j, f_i - f_j\big)^{1/2}\, \big(\|f_i\|_H + \|f_j\|_H\big) \\
&\underset{\substack{\|f_i\|_H \leq 1,\\ \|f_j\|_H \leq 1}}{\leq} 2\frac{(C_d)^{3/2}}{(C_E)^2}\, d\big(f_i - f_j, f_i - f_j\big)^{1/2}.
\end{aligned}$$

Since $\|f_i\|_H \leq 1$ for all $i \in \mathbb{N}$ it follows from Precondition A.19 iii) that there exists a subsequence $(f_{i_k})_{k\in\mathbb{N}}$ which is Cauchy w.r.t. $d(\cdot,\cdot)^{1/2}$. Altogether we obtain that

$$\|Tf_{i_k} - Tf_{i_l}\|_H^2 \quad \leq \quad 2\frac{(C_d)^{3/2}}{(C_E)^2}\, d\big(f_{i_k} - f_{i_l}, f_{i_k} - f_{i_l}\big)^{1/2} \xrightarrow{k,l\to\infty} 0,$$

i.e., each sequence $(Tf_i)_{i\in\mathbb{N}} \subset B$ has a subsequence which is convergent in H. Hence, the subset B is precompact and it follows that operator T is compact. ∎

Remark A.21 *The assumption made in Lemma A.20 on the bilinear form $d(\cdot,\cdot)$ can be replaced by the following stronger assumption: To prove that the solution operator T is compact it is sufficient to assume that $d : U \times U \to \mathbb{R}$ is a bilinear form operating on a larger Banach space U, i.e., $H \subset U$ where*

i) $d(\cdot,\cdot)$ is symmetric and $d(u,u) > 0$ for all $u \in U \setminus \{0\}$.

ii) $d(\cdot,\cdot)$ is continuous on $U \times U$, i.e., it holds $\|d\|_{\mathbb{R}\leftarrow U\times U} =: C_d < \infty$.

iii) The embedding of H in U is compact.

Since the solution operator T in (A.11) is compact, Theorem A.12 can be applied and we obtain that the operator eigenvalue problem

$$\begin{cases} \text{find } (\mu, u) \in \mathbb{C} \times H \setminus \{0\} \text{ such that} \\ Tu = \mu\, u \end{cases} \tag{A.12}$$

has countably many eigensolutions. Note that the variational eigenvalue problem (A.9) is equivalent to problem (A.12) in such a way that for $\lambda \neq 0$ it holds

$$\begin{aligned} &(\lambda, u) \in \mathbb{C} \times H \setminus \{0\} \text{ is an eigenpair of (A.9)} \\ &\qquad\qquad \Longleftrightarrow \\ &\left(\frac{1}{\lambda}, u\right) \in \mathbb{C} \times H \setminus \{0\} \text{ is an eigenpair of (A.12).} \end{aligned} \tag{A.13}$$

Since it is assumed that the bilinear form $a(\cdot,\cdot) : H \times H \to \mathbb{R}$ is symmetric elliptic, an inner product is induced on H by

$$(\cdot,\cdot)_{a(\cdot,\cdot)} : H \times H \to \mathbb{R} \quad \text{with} \quad (u,v)_{a(\cdot,\cdot)} := a(u,v) \quad \forall\, u, v \in H.$$

Furthermore, the associated norm $\|u\|_{a(\cdot,\cdot)} := a(u,u)^{1/2}$, which is also called *energy norm* in literature, is equivalent to the norm $\|\cdot\|_H$ because the bilinear form $a(\cdot,\cdot)$ is elliptic and continuous. Hence, the Banach space H forms together with the inner product $(\cdot,\cdot)_{a(\cdot,\cdot)}$ a Hilbert space. Using the inner product induced by $a(\cdot,\cdot)$ for the underlying space one can derive additional properties for the solution operator T.

Lemma A.22 *Let the assumptions of Lemma A.20 be fulfilled. Then the solution operator T in* (A.11) *is selfadjoint and positive definite regarding to the inner product $(\cdot,\cdot)_{a(\cdot,\cdot)}$.*

Proof: From the symmetry of $a(\cdot,\cdot)$ and $d(\cdot,\cdot)$ it follows for all $u, v \in H$ that

$$(Tu, v)_{a(\cdot,\cdot)} = a(Tu, v) = d(u, v) = d(v, u) = a(Tv, u) = a(u, Tv) = (u, Tv)_{a(\cdot,\cdot)},$$

i.e., T is selfadjoint. Because of

$$(u, Tu)_{a(\cdot,\cdot)} = a(u, Tu) = a(Tu, u) = d(u, u) > 0 \quad \forall\, u \in H \setminus \{0\} \tag{A.14}$$

it follows that T is positive definite. ∎

Altogether the following result is obtained for the variational eigenvalue problem (A.9).

Corollary A.23 *Consider the variational eigenvalue problem* (A.9)*. Let $a(\cdot,\cdot)$ be symmetric elliptic and let $d(\cdot,\cdot)$ satisfy the assumptions made in Precondition A.19 or the assumptions made in Remark A.21. Then problem* (A.9) *possesses a countable family of eigensolutions*

$$\big(\lambda_j, u_j\big)_{j=1}^{N} \subset \mathbb{R}_{>0} \times H \setminus \{0\} \tag{A.15}$$

with $N \in \mathbb{N} \cup \{\infty\}$ and eigenvalues λ_j ordered[6] *such that $\lambda_j \leq \lambda_{j+1}$. Furthermore, it holds:*

i) All eigenvalues are real and positive. Furthermore, it holds $\#N = \dim H$ and in the case of $\dim H = \infty$ we have $\lambda_j \xrightarrow{j\to\infty} \infty$.

ii) The eigenspace $E(\lambda_j) \subset H$ of the eigenvalue λ_j, which is defined by

$$E(\lambda_j) := \operatorname{span}\Big\{ u \in H \;:\; a(u,v) = \lambda_j d(u,v) \quad \forall\, v \in H \Big\},$$

is finite-dimensional.

[6] Eigenvalues are repeated in (A.15) according to their geometric multiplicity.

iii) If it holds $\lambda_j \neq \lambda_k$ then the corresponding eigenfunctions u_j and u_k are orthogonal with respect to $a(\cdot,\cdot)$ and $d(\cdot,\cdot)$, i.e., we have $a(u_j, u_k) = 0$ and $d(u_j, u_k) = 0$.

iv) The eigenfunctions $\left(u_j\right)_{j=1}^{N}$ form a basis of H and without loss of generality it can be assumed that all eigenfunctions are orthonormal with respect to $a(\cdot,\cdot)$ or $d(\cdot,\cdot)$.

Proof: According to Lemma A.20 and Lemma A.22 the solution operator T associated to eigenvalue problem (A.9) is compact, selfadjoint and positive definite. Correspondingly, Theorem A.12 and Theorem A.16 can be applied to characterise the spectrum and the eigensolution of the operator T. Equivalence (A.13) proves the remaining statements. In particular, for eigenvalues $\lambda_k \neq \lambda_l$ it holds because of the symmetry of $a(\cdot,\cdot)$ and $d(\cdot,\cdot)$ that

$$\lambda_k \, d(u_k, u_l) = a(u_k, u_l) = a(u_l, u_k) = \lambda_l \, d(u_l, u_k) = \lambda_l \, d(u_k, u_l).$$

Since $\lambda_k \neq \lambda_l$ it follows that $d(u_k, u_l) = 0$, and from $a(u_k, u_l) = \lambda_k \, d(u_k, u_l)$ it follows that $a(u_k, u_l) = 0$. ■

B. Theory of Sobolev Spaces

In this section basic definitions and results from the theory of Sobolev spaces are recalled which are used for the analysis of elliptic PDE eigenvalue problems. To introduce this topic, basic function spaces are recalled (Section B.1) and the smoothness of the boundary of a domain $\Omega \subset \mathbb{R}^d$ is discussed (Section B.2). Afterwards the weak derivatives is introduced and basic results from the theory of the associated Sobolev spaces are discussed (Section B.3). More information on Sobolev spaces can be found in [37] or any textbook on this topic.

Throughout the whole section it is assumed that Ω is an open subset of $\mathbb{R}^d$.

B.1. Basic Function Spaces

First of all, different differentiability classes of functions $f : \Omega \to \mathbb{R}$ are listed.

Definition B.1 ($C^k(\Omega)$-functions)

- *Let M be a subset of $\mathbb{R}^d$, then the space of functions $f : M \to \mathbb{R}$ which are continuous on M is denoted by $C^0(M)$.*
- *For $k \in \mathbb{N}$ we denote by $C^k(\Omega)$ the space of functions $f : \Omega \to \mathbb{R}$ for which all (partial) derivatives $D^\alpha f$ of order $|\alpha| \leq k$ exist, with $\alpha \in \mathbb{N}_0^d$, and where all these derivatives are continuous on Ω. Note that $\alpha = (\alpha_1, \ldots, \alpha_d)$ is a multi-index and that the notation*

$$D^\alpha f := \frac{\partial^{|\alpha|} f}{\partial x_1^{\alpha_1} \ldots \partial x_d^{\alpha_d}} \quad \text{with} \quad |\alpha| := \alpha_1 + \ldots + \alpha_d.$$

  is used.

  For the closure of the open set $\Omega \subset \mathbb{R}^d$ the space $C^k(\overline{\Omega})$ is introduced which is defined by the set of all functions $f \in C^k(\Omega) \cap C^0(\overline{\Omega})$ where all (partial) derivatives $D^\alpha f$ of order $|\alpha| \leq k$ can be continuously extended to $\overline{\Omega}$.
- *The space of functions $f : \Omega \to \mathbb{R}$ which have (partial) derivatives of any order is denoted by $C^\infty(\Omega)$. Functions $f \in C^\infty(\Omega)$ are called smooth.*
- *A function $f \in C^\infty(\Omega)$ is called analytic, if f is equal to its Taylor series expansion around any point $x \in \Omega$. The space of all these functions is denoted by $f \in C^\omega(\Omega)$ and is a proper subset of $C^\infty(\Omega)$.*

Definition B.2 ($C_0^\infty(\Omega)$-function) *The support of a function $f : \Omega \to \mathbb{R}$ is defined by*

$$\operatorname{supp}(f) := \overline{\{x \in \Omega : f(x) \neq 0\}}.$$

The so-called space of test (or bump) functions is then defined by

$$C_0^\infty(\Omega) := \Big\{ f \in C^\infty(\Omega) \ : \ \operatorname{supp}(f) \text{ is compact}, \operatorname{supp}(f) \subset \Omega, \operatorname{supp}(f) \cap \partial\Omega = \emptyset \Big\}.$$

The Sobolev spaces, which are introduced in Section B.3, are based on the Lebesgue space $L^2(\Omega)$ which is defined as follows:

Definition B.3 ($L^2(\Omega)$-function) *The set of all functions $f : \Omega \to \mathbb{R}$ where $|f|^2$ is Lebesgue integrable on Ω is denoted by $L^2(\Omega)$. Two functions $f, g \in L^2(\Omega)$ are considered as equal ($f = g$), if it holds $f(x) = g(x)$ for almost all $x \in \Omega$, i.e., the equality holds for all $x \in \Omega \setminus M$ where M is a suitable measure-zero set.*

Theorem B.4 • *$L^2(\Omega)$ is a Hilbert space together with the inner product and the norm*

$$(f, g)_0 := (f, g)_{L^2(\Omega)} := \int_\Omega f(x)g(x)\,\mathrm{d}x \qquad \text{with } f, g \in L^2(\Omega),$$
$$\|f\|_0 := \|f\|_{L^2(\Omega)} := \left\{ \int_\Omega |f(x)|^2\,\mathrm{d}x \right\}^{1/2} \qquad \text{with } f \in L^2(\Omega).$$

- *The spaces $C^\infty(\Omega) \cap L^2(\Omega)$ and $C_0^\infty(\Omega)$ are densein $L^2(\Omega)$.*

Definition B.5 ($L^\infty(\Omega)$-function) *The set of all functions $f : \Omega \to \mathbb{R}$ that are bounded almost everywhere and locally Lebesgue integrable on Ω is denoted by $L^\infty(\Omega)$. It is not distinguished between functions which are equal almost everywhere on Ω. The supremum norm is defined by*

$$\|f\|_{L^\infty(\Omega)} := \inf\left\{ \sup_{x \in \Omega \setminus M} |f(x)| \ : \ M \text{ is a measure-zero set} \right\}$$

and with this norm the linear space $L^\infty(\Omega)$ forms a Banach space.

B.2. Classification of the Boundary

In the analysis of partial differential equations and in the theory related to Sobolev functions $f : \Omega \to \mathbb{R}$ it is often required that the subset $\Omega \subset \mathbb{R}^d$ has a sufficiently smooth boundary.

Definition B.6 (Boundary) *The boundary $\partial\Omega$ of a subset $\Omega \subset \mathbb{R}^d$ is defined by*

$$\partial\Omega := \overline{\Omega} \cap \overline{(\mathbb{R}^d \setminus \Omega)}.$$

To classify the smoothness of the boundary $\partial\Omega$ the so-called Hölder spaces are introduced.

Definition B.7 (Hölder Spaces) *Let M be an open and bounded subset of $\mathbb{R}^d$. Then the Hölder spaces $C^{0,\lambda}(\overline{M})$ and $C^{k,\lambda}(\overline{M})$ are defined as follows:*

- *A function $f \in C^0(\overline{M})$ is called (uniformly) Hölder continuous on $\overline{M}$ with exponent $\lambda \in (0, 1]$, if there exists a constant $C_f > 0$ such that*

$$|f(x) - f(y)| \le C_f \|x - y\|^\lambda \quad \text{for all } x, y \in \overline{M}.$$

The set of all Hölder continuous functions forms a linear space which is denoted by $C^{0,\lambda}(\overline{M})$. In the special case $\lambda = 1$ we call a function $f \in C^{0,1}(\overline{M})$ Lipschitz continuous.

- *For $k \in \mathbb{N}$ and $\lambda \in (0,1]$ the space $C^{k,\lambda}(\overline{M})$ is defined by*

$$C^{k,\lambda}(\overline{M}) := \left\{ f \in C^k(\overline{M}) \ : \ D^\alpha f \in C^{0,\lambda}(\overline{M}) \text{ for all } \alpha \in \mathbb{N}_0^d \text{ with } |\alpha| \leq k \right\}.$$

A function $f \in C^{k,\lambda}(\overline{M})$ is called k-fold Hölder continuously differentiable.

For the Hölder spaces defined above the following inclusions are valid:

Remark B.8 *For all $k \in \mathbb{N}_0$ and all $\lambda, \mu \in (0,1]$ with $\lambda < \mu$ we have*

$$C^{k,\lambda}(\overline{M}) \subset C^k(\overline{M}) \qquad \text{and} \qquad C^{k,\mu}(\overline{M}) \subset C^{k,\lambda}(\overline{M}).$$

The definition of the Hölder spaces can be generalised to functions $f : \overline{M} \to U$ where U is a subset of $\mathbb{R}^d$. Note that a function $f : \overline{M} \to U$ can be entirely described by its component functions f_i for $i = 1, \ldots, d$ with

$$f_i : \overline{M} \to \mathbb{R} \quad \text{and} \quad f(x) = (\, f_1(x), \ldots, f_d(x)\,)^T \quad \text{for all } x \in \overline{M}. \tag{B.1}$$

Definition B.9 *Let $M \subset \mathbb{R}^d$ be open and bounded, and $U \subset \mathbb{R}^n$ for some $n \in \mathbb{N}$. For $k \in \mathbb{N}_0$ and $\lambda \in (0,1]$ we define the space $C^{k,\lambda}(\overline{M}, U)$ by all functions $f : \overline{M} \to U$ which possess component functions $f_i : \overline{M} \to \mathbb{R}$ with $f_i \in C^{k,\lambda}(\overline{M})$ for all $i = 1, \ldots, n$.*

Finally, the smoothness of the boundary of $\Omega \subset \mathbb{R}^d$ is characterised as follows:

Definition B.10 (Boundary Classification) *Let $k \in \mathbb{N}_0$ and $\lambda \in (0,1]$. Furthermore, we denote by $B_r(x_0) := \left\{ x \in \mathbb{R}^d \ : \ \|x - x_0\| < r \right\}$ the d-dimensional open ball of radius $r > 0$ and centre $x_0 \in \mathbb{R}^d$.*

- *We say the boundary $\partial\Omega$ is of class $C^{k,\lambda}$ and write $\partial\Omega \in C^{k,\lambda}$ if for every point $x_0 \in \partial\Omega$ there exists a radius $r > 0$ and a bijection $\psi : \overline{B_r(x_0)} \to \overline{B_1(0)}$ such that*

$$\psi(\, B_r(x_0) \cap \Omega\,) = B_1^+(0) := \left\{ (x_1, \ldots, x_d)^T \in B_1(0) \ : \ x_d > 0 \right\}, \tag{B.2}$$

$$\psi(\, B_r(x_0) \cap \partial\Omega\,) = B_1^0(0) := \left\{ (x_1, \ldots, x_d)^T \in B_1(0) \ : \ x_d = 0 \right\}, \tag{B.3}$$

$$\psi(\, B_r(x_0) \cap \mathbb{R}^d \setminus \overline{\Omega}\,) = B_1^-(0) := \left\{ (x_1, \ldots, x_d)^T \in B_1(0) \ : \ x_d < 0 \right\}, \tag{B.4}$$

and

$$\psi \in C^{k,\lambda}(\, \overline{B_r(x_0)}, \overline{B_1(0)}\,), \qquad \psi^{-1} \in C^{k,\lambda}(\, \overline{B_1(0)}, \overline{B_r(x_0)}\,). \tag{B.5}$$

- *Furthermore, a boundary $\partial\Omega \in C^{k,\lambda}$ is called analytic if in addition to (B.2)—(B.5) it holds that all component functions $\psi_i : \overline{B_r(x_0)} \to \mathbb{R}$ of the bijection $\psi : \overline{B_r(x_0)} \to \overline{B_1(0)}$ are analytic. In particular, it is assumed that there exist constants $C_\psi, \gamma_\psi \in \mathbb{R}_{>0}$ such that for all $n \in \mathbb{N}_0$ it holds*

$$|\psi|_{n,\infty} := \left\| \left\{ \sum_{|\alpha|=n} \frac{n!}{\alpha!} |D^\alpha \psi|^2 \right\}^{1/2} \right\|_{L^\infty(\, B_r(x_0)\,)} \leq C_\psi \, n! \, (\gamma_\psi)^n$$

where $|D^\alpha \psi(x)|^2 := \sum_{i=1}^d |D^\alpha \psi_i(x)|^2$.

In the analysis of partial differential equations and the theory of Sobolev spaces it is often sufficient to have a boundary $\partial\Omega$ of class $C^{0,1}$, i.e., the boundary can be represented locally by the graph of Lipschitz-continuous functions. Beside this, it is often needed that Ω is open, connected[1] and bounded:

Definition B.11 (Lipschitz Domain) *A set $\Omega \subset \mathbb{R}^d$ which is open and connected is called domain. If it additionally holds $\partial\Omega \in C^{0,1}$ then Ω is said to have a Lipschitz boundary, and Ω is called Lipschitz domain.*

The assumption that a partial differential equation is defined on a Lipschitz domain is not very restrictive, since these properties are usually fulfilled by the domains used in practice.

Remark B.12 *Let $\Omega \subset \mathbb{R}^d$ be a domain with Lipschitz boundary $\partial\Omega$. Then, there exists an exterior normal field $\vec{n}$ almost everywhere on $\partial\Omega$. That means $\vec{n}(x) \in \mathbb{R}^d$ is a unit vector, orthogonal to the tangential hyperplane in $x \in \partial\Omega$, and directed to the outside. The normal derivative of u in $x \in \partial\Omega$ is defined by*

$$\frac{\partial u(x)}{\partial n} := (\vec{n}(x))^T \nabla u(x).$$

B.3. The Weak Derivative and the Sobolev Space

The theory for the solution of elliptic PDE eigenvalue problems is based on the concept of weak derivatives and the corresponding Sobolev spaces. Important definitions and results which are related to this topic are summarised in the following.

Definition B.13 (Weak Derivative) *Consider $u \in L^2(\Omega)$ and the multi-index $\alpha = (\alpha_1, \ldots, \alpha_d) \in \mathbb{N}_0^d$ with $|\alpha| > 0$. We say, the function u has the α-th weak derivative $D^\alpha u$, if there exists a function $D^\alpha u := w \in L^2(\Omega)$ with[2]*

$$(v, w)_0 = (-1)^{|\alpha|} (\, D^\alpha v, u\,)_0 \quad \text{for all } v \in C_0^\infty(\Omega) \tag{B.6}$$

where $D^\alpha v$ is the α-th (partial) derivative of v with

$$D^\alpha v = \frac{\partial^{|\alpha|} v}{\partial x_1^{\alpha_1} \ldots \partial x_d^{\alpha_d}} \quad \text{and} \quad |\alpha| := \alpha_1 + \ldots + \alpha_d.$$

Lemma B.14 *If the weak derivative $D^\alpha u \in L^2(\Omega)$ exists for a function $u \in L^2(\Omega)$ and $\alpha \in \mathbb{N}_0^d$, then it is unique in $L^2(\Omega)$. If it additionally holds that $u \in C^k(\Omega)$ for some $k \in \mathbb{N}$ and if $0 < |\alpha| \leq k$, then the α-th weak derivative and the α-th classical derivative are equal almost everywhere in Ω.*

Remark B.15 *As noted above, the weak derivative and the classical derivative — if both exist — are equal in the $L^2(\Omega)$ sense. However, the existence of the classical derivative does not imply the existence of the corresponding weak derivative, and also not the other way around. For example, even the condition $u \in C^\infty(\Omega) \cap C^0(\overline{\Omega})$ does not guarantee the existence of the first weak derivative.*

[1] A set $\Omega \subset \mathbb{R}^d$ is called connected if for all $x, y \in \Omega$ exists a function $\varphi \in C^0([0,1])$ with $\varphi : t \in [0,1] \mapsto \varphi(t) \in \Omega$ and $\varphi(0) = x$, $\varphi(1) = y$.

[2] The requirement (B.6) on the weak derivative is adapted to Green's formula: For given $k \in \mathbb{N}$ and $\alpha \in \mathbb{N}_0^d$ with $|\alpha| \leq k$ it holds $\int_\Omega D^\alpha u(x) v(x)\, \mathrm{d}x = (-1)^{|\alpha|} \int_\Omega u(x) D^\alpha v(x)\, \mathrm{d}x$ for all $u \in C^k(\Omega)$ and $v \in C_0^\infty(\Omega)$.

Definition B.16 (Sobolev Space I) *For $k \in \mathbb{N}_0$ the Sobolev space $H^k(\Omega)$ is given by*

$$H^k(\Omega) := \left\{ f \in L^2(\Omega) \ : \ D^\alpha f \in L^2(\Omega) \text{ exists for all } \alpha \in \mathbb{N}_0^d \text{ with } |\alpha| \leq k \right\}$$

where $D^\alpha f$ denotes the α-th (partial) weak derivative of f.

Theorem B.17 *For $\Omega \subset \mathbb{R}^d$ it holds:*

i) $H^k(\Omega)$ is a Hilbert space with the inner product and norm

$$(f,g)_k := (f,g)_{H^k(\Omega)} := \sum_{|\alpha| \leq k} \int_\Omega D^\alpha f(x) D^\alpha g(x) \, \mathrm{d}x \qquad \text{with } f,g \in H^k(\Omega),$$

$$\|f\|_k := \|f\|_{H^k(\Omega)} := \left\{ \sum_{|\alpha| \leq k} \|D^\alpha f\|^2_{L^2(\Omega)} \right\}^{1/2} \qquad \text{with } f \in H^k(\Omega).$$

ii) A seminorm[3] on $H^k(\Omega)$ is given by

$$|f|_k := |f|_{H^k(\Omega)} := \left\{ \sum_{|\alpha| = k} \|D^\alpha f\|^2_{L^2(\Omega)} \right\}^{1/2}.$$

iii) If Ω is a bounded Lipschitz domain, then $|\cdot|_k$ and $\|\cdot\|_k$ are equivalent norms on $H_0^k(\Omega)$.

iv) $C^\infty(\Omega) \cap H^k(\Omega)$ is dense in $H^k(\Omega)$.

In contrast to $C^\infty(\Omega)$ the smaller function space $C_0^\infty(\Omega)$ is in general not dense in $H^k(\Omega)$. Since $C_0^\infty(\Omega)$-functions vanish close to the boundary $\partial\Omega$, the closure of $C_0^\infty(\Omega)$ with respect to the $H^k(\Omega)$-norm is used to define the Sobolev space with a *zero-boundary* in a "weak" sense.

Definition B.18 (Sobolev Space II) *The space $H_0^k(\Omega)$ is defined as the closure of $C_0^\infty(\Omega)$ with respect to the norm $\|\cdot\|_k$ of $H^k(\Omega)$.*

For the investigation of partial differential equations and their weak formulations, it is necessary to evaluate boundary values of Sobolev functions. For a function $u \in C^0(\overline{\Omega})$ the boundary values are described by the restriction $u_{|\partial\Omega}$. However, for a function $u \in H^1(\Omega)$ it is not clear how to describe its values on the boundary $\partial\Omega$, since $\partial\Omega$ is a measure-zero set. This problem, however, is resolved by the following theorem.

Theorem B.19 (Trace Operator) *Let $\Omega \subset \mathbb{R}^d$ be a bounded domain with Lipschitz boundary $\Gamma := \partial\Omega$. Then there exists a unique bounded linear operator $\gamma : H^1(\Omega) \to L^2(\Gamma)$ with*

$$\begin{aligned} &\forall\, u \in C^1(\overline{\Omega}) : \quad \gamma(u) = u_{|\Gamma}, \\ &\forall\, u \in H^1(\Omega) : \quad \|\gamma(u)\|_{L^2(\Gamma)} \leq C_\Omega \, \|u\|_{H^1(\Omega)}, \end{aligned}$$

where $C_\Omega > 0$ is a constant depending only on Ω. The operator γ is called the trace operator, and for a given $u \in H^1(\Omega)$ the function $\gamma(u) \in L^2(\Gamma)$ is called the trace of u.

[3] A seminorm on a linear space X is a mapping $|\cdot| : X \to [0,\infty)$ which has the same properties as a norm except that it has not to hold that $|\cdot|$ is positive definite.

Remark B.20 *i) The function $\gamma(u)$ represents the "values" of the function $u \in H^1(\Omega)$ on the boundary $\Gamma := \partial\Omega$. Correspondingly, the notation $u_{|\Gamma}$ is often used for $\gamma(u)$. For example, for a given $u \in H^1(\Omega)$, the equation $u_{|\Gamma} = 0$ has to be interpreted as $\gamma(u) = 0$ in $L^2(\Gamma)$.*

ii) In general the trace operator is not surjective. It can be only shown that $\mathrm{Im}(\gamma)$ is dense in $L^2(\Gamma)$. However, $\mathrm{Im}(\gamma)$ can be characterised by the so-called Sobolev-Slobodeckij space $H^{1/2}(\Gamma)$ via

$$H^{1/2}(\Gamma) = \mathrm{Im}(\gamma) = \Big\{ v \in L^2(\Gamma) \ : \ \exists u \in H^1(\Omega) \text{ with } v = \gamma(u) \Big\}.$$

The Sobolev-Slobodeckij spaces $H^s(\Gamma)$ are Hilbert spaces with inner product $(\cdot,\cdot)_{H^s(\Gamma)}$ and defined for indices $s \in \mathbb{R}_{\geq 0}$ (cf. [37, Theorem 6.2.39]). These spaces can be seen as a generalisation of the usual Sobolev spaces with quite similar properties as $H^k(\Gamma)$ with $k \in \mathbb{N}$. In particular, if Ω is a bounded Lipschitz domain then the trace operator $\gamma : H^1(\Omega) \to H^{1/2}(\Gamma)$ is continuous (cf. [37, Theorem 6.2.40]), i.e., there exists a constant $C > 0$ such that $\|\gamma(u)\|_{H^{1/2}(\Gamma)} \leq C\|u\|_{H^1(\Omega)}$ where $\|\tau\|_{H^{1/2}(\Gamma)} := (\tau,\tau)^{1/2}_{H^{1/2}(\Gamma)}$ is the norm of $\tau \in H^{1/2}(\Gamma)$. Vice versa, there exists a bounded linear (extension) operator $\Phi : H^{1/2}(\Gamma) \to H^1(\Omega)$ such that for all $\tau \in H^{1/2}(\Gamma)$ it holds $\gamma(\widetilde{\tau}) = \tau$ with $\widetilde{\tau} := \Phi(\tau)$. In particular, there exists a constant $C' > 0$ such that $\|\Phi(\tau)\|_{H^1(\Omega)} \leq C'\|\tau\|_{H^{1/2}(\Gamma)}$.

Using the trace operator from Theorem B.19 the space $H^k_0(\Omega)$ can be characterised for the special case $k = 1$ in a more natural way:

Theorem B.21 *If $\Omega \subset \mathbb{R}^d$ is a Lipschitz domain, then it holds*

$$H^1_0(\Omega) = \Big\{ u \in H^1(\Omega) : u_{|\partial\Omega} = 0 \Big\}$$

where $u_{|\partial\Omega}$ is the value of u on the boundary $\partial\Omega$ according to Remark B.20 i).

In Lemma B.14 and Remark B.15 the connection between the weak and the classical derivative has been discussed. The following theorem partially treats this issue. More precisely, in Theorem B.22 the smoothness of Sobolev functions in the sense of weak derivatives is related to the classical smoothness in the sense of $C^m(\overline{\Omega})$. For example, in the one-dimensional case with $\Omega \subset \mathbb{R}$ it can be shown that all $H^1(\Omega)$-functions are continuous on $\overline{\Omega}$, however, in the two-dimensional case this is in general not true.

Theorem B.22 (Sobolev's Embedding Theorem) *Let $\Omega \subset \mathbb{R}^d$ be a bounded Lipschitz domain and $m \in \mathbb{N}_0$. Furthermore, the notations $H^0(\Omega) := L^2(\Omega)$ and $H^0_0(\Omega) := L^2(\Omega)$ are used. Then, depending on the index $k \in \mathbb{N}_0$ of the Sobolev space and the spatial dimension $d \in \mathbb{N}$, the following embeddings are continuous and compact:*

$$\left.\begin{array}{l} H^{k+1}(\Omega) \hookrightarrow H^k(\Omega), \\ H^{k+1}_0(\Omega) \hookrightarrow H^k_0(\Omega) \end{array}\right\} \text{ for } k \geq 0, \qquad \text{and} \qquad \left.\begin{array}{l} H^k(\Omega) \hookrightarrow C^m(\overline{\Omega}), \\ H^k_0(\Omega) \hookrightarrow C^m(\overline{\Omega}) \end{array}\right\} \text{ for } k - \tfrac{d}{2} > m.$$

Especially, the embeddings

$$H^k(\Omega) \hookrightarrow L^2(\Omega) \text{ for } k \in \mathbb{N} \qquad \text{and} \qquad H^k_0(\Omega) \hookrightarrow L^2(\Omega) \text{ for } k \in \mathbb{N}$$

are continuous, compact and dense.

C. Asymptotic Distribution of the Eigenvalues

In this section fundamental results on the asymptotic distribution of the eigenvalues of the elliptic PDE eigenvalue problem (1.1) are presented. These result derived here will be used in Appendix D to refine the error estimates of the finite element discretisation.

Since the numerical computation of eigenvalues λ_j becomes very difficult for large j (and only for some problems the eigenvalues can be computed analytically), in literature the asymptotic behaviour of the eigenvalues has been investigated as $j \to \infty$. However, instead of deriving the asymptotics for λ_j it has been shown that it is more convenient to calculate the asymptotic distribution of the eigenvalues. The first important results in this field have been derived by Weyl in 1911 (cf. [66]) where he described the asymptotic behaviour[1] of the eigenvalues of the Laplace eigenvalue problem on bounded Lipschitz domains. From then on this field has been investigated by many (see, for example, [1], [4], [25, Chapter VI, Section 4], [58]). The main result reads as follows.

Theorem C.1 (Weyl's Law) *Consider the variational eigenvalue problem* (2.12) *and assume that a) and b) of Precondition 3.12 are fulfilled. Furthermore, define the eigenvalue counting functions* $N_1, N_2 : \mathbb{R} \to \mathbb{N}_0$ *by*

$$N_1(t) := \operatorname{card}\big\{\, j \,:\, \lambda_j < t \,\big\} \qquad \text{and} \qquad N_2(t) := \operatorname{card}\big\{\, j \,:\, \lambda_j \le t \,\big\}.$$

Then the asymptotic distribution of the eigenvalues is described by

$$\lim_{t\to\infty} \frac{N_1(t)}{t^{d/2}} \;=\; C_{\text{Weyl}} \qquad \text{and} \qquad \lim_{t\to\infty} \frac{N_2(t)}{t^{d/2}} \;=\; C_{\text{Weyl}} \tag{C.1}$$

where $C_{\text{Weyl}} > 0$ *is a constant which only depends on the underlying domain* Ω *and on the coefficients of the underlying partial differential operator* L.

Proof: In [58, Theorem 13.1] it is proven that for all $s \in (0, \frac{1}{3})$ it holds

$$N_1(t) \;=\; C_{\text{Weyl}}\, t^{d/2} + \mathcal{O}\big(t^{(d-s)/2}\big) \quad \text{as } t \to \infty. \tag{C.2}$$

It follows that statement (C.1) is valid for counting function N_1. Statement (C.1) for counting function N_2 is derived as follows: Since eigenvalue problem (2.12) has a discrete spectrum there exist for all $t \in \mathbb{R}$ some $\delta_1(t) > 0$ such that $N_2(t) = N_1(t+\delta)$ for all $0 < \delta < \delta_1(t)$. Furthermore, for all $\varepsilon > 0$ and all $t > 0$ there exists an $\delta_2(t, \varepsilon) > 0$ such that

$$\Big| \frac{N_2(t)}{t^{d/2}} - \frac{N_2(t)}{(t+\delta)^{d/2}} \Big| \;<\; \varepsilon \qquad \text{for all } 0 < \delta < \delta_2(t, \varepsilon).$$

[1] The behaviour of the asymptotic distribution of the eigenvalues for the Laplace problem became known as *Weyl's Law*.

We conclude that

$$\frac{N_1(t+\delta)}{(t+\delta)^{d/2}} - \varepsilon \;<\; \frac{N_2(t)}{t^{d/2}} \;<\; \frac{N_1(t+\delta)}{(t+\delta)^{d/2}} + \varepsilon \qquad \forall\, 0 < \delta < \min\bigl\{\delta_1(t), \delta_2(t,\varepsilon)\bigr\}. \tag{C.3}$$

From (C.3) we obtain for $t \to \infty$, by using the already proven statement (C.1) for N_1, that

$$C_{\text{Weyl}} - \varepsilon \;\leq\; \lim_{t\to\infty} \frac{N_2(t)}{t^{d/2}} \;\leq\; C_{\text{Weyl}} + \varepsilon$$

which finally leads to statement (C.1) for counting function N_2 when $\varepsilon \to 0$. ■

As seen in Theorem C.1 counting function N_1 and N_2 can be equivalently used to express the asymptotic statement (C.1). In literature counting function N_2 is more commonly used. Note that the counting functions take the multiplicities of the eigenvalues into account. Using statement (C.1) the following bounds for the eigenvalues of problem (2.12) can be derived.

Theorem C.2 (Eigenvalue Bounds) *Let the assumptions of Theorem C.1 be fulfilled. Then there exist constants $C_\mathrm{b}, c_\mathrm{b} > 0$ independent of j (with $c_\mathrm{b} < C_{\text{Weyl}} < C_\mathrm{b}$), such that the eigenvalues of problem* (2.12) *can be bounded by*

$$c_\mathrm{b}\, \lambda_j^{d/2} \;\leq\; j \;\leq\; C_\mathrm{b}\, \lambda_j^{d/2} \quad \text{and} \quad \left(\frac{j}{C_\mathrm{b}}\right)^{2/d} \leq\; \lambda_j \;\leq \left(\frac{j}{c_\mathrm{b}}\right)^{2/d} \qquad \text{for all } j \in \mathbb{N}. \tag{C.4}$$

Proof: From the definition of the counting functions it follows

$$N_1(\lambda_j) \;<\; j \qquad \text{and} \qquad j \;\leq\; N_2(\lambda_j) \qquad \text{for all } j \in \mathbb{N}. \tag{C.5}$$

Furthermore, since the sequence $(\lambda_j)_{j\in\mathbb{N}}$ is monotonically increasing with $\lambda_j \to \infty$ for $j \to \infty$ it follows from (C.1) that

$$\lim_{j\to\infty} \frac{N_1(\lambda_j)}{\lambda_j^{d/2}} \;=\; C_{\text{Weyl}} \qquad \text{and} \qquad \lim_{j\to\infty} \frac{N_2(\lambda_j)}{\lambda_j^{d/2}} \;=\; C_{\text{Weyl}}. \tag{C.6}$$

Hence, there exist constants $C_\mathrm{b}, c_\mathrm{b} > 0$ independent of j (with $c_\mathrm{b} < C_{\text{Weyl}} < C_\mathrm{b}$) such that

$$c_\mathrm{b} \;\leq\; \frac{N_1(\lambda_j)}{\lambda_j^{d/2}} \qquad \text{and} \qquad \frac{N_2(\lambda_j)}{\lambda_j^{d/2}} \;\leq\; C_\mathrm{b} \qquad \text{for all } j \in \mathbb{N}. \tag{C.7}$$

Combining (C.5) with (C.7) we obtain that

$$c_\mathrm{b}\, \lambda_j^{d/2} \;\leq\; N_1(\lambda_j) \;<\; j \quad \text{and} \quad j \;\leq\; N_2(\lambda_j) \;\leq\; C_\mathrm{b}\, \lambda_j^{d/2} \qquad \text{for all } j \in \mathbb{N}$$

which finally leads to statement (C.4). ■

In the following the asymptotic behaviour of the spectral gap is analysed. In particular, it is discussed when a lower bound for the relative spectral gap

$$\Delta_1(\lambda_j) \;:=\; \frac{\operatorname{dist}(\, \lambda_j \,,\, \sigma(L) \setminus \{\lambda_j\}\,)}{\lambda_j}$$

can be found where $\sigma(L) \subset \mathbb{R}$ is defined as the spectrum of eigenvalue problem (2.12). To the best of the author's knowledge in literature asymptotic sharp lower bounds for the spectral gap are not available. Only in [64] it has been motivated that under quite technical assumptions on the right-hand side of (C.2), and provided that λ_j has multiplicity 1 and λ_{j-1} is sufficiently large, it holds

$$\Delta_1(\lambda_j) \;=\; \lambda_j^{-d/2}\Big(\, C \,\pm\, \mathcal{O}\Big(\, \lambda_{j-1}^{-d/2} \, \Big) \, \Big) \quad \text{as } j \to \infty \tag{C.8}$$

with for some constant $C > 0$.

For the analysis of the spectral gap an additional notation and ordering of the eigenvalues of problem (2.12) is introduced in this particular section: For $j \in \mathbb{N}$ we denote with $\tilde{\lambda}_j$ the eigenvalues of (2.12) which are ordered by the size but where — in contrast to the ordering (3.13) — the multiplicity of the eigenvalues is not taken into account, i.e., we have

$$0 < \tilde{\lambda}_1 < \tilde{\lambda}_2 < \tilde{\lambda}_3 < \ldots .$$

For $j \in \mathbb{N}$ we define

$$\tilde{k}_j \;:=\; \dim E(\,\tilde{\lambda}_j\,) \qquad \text{and} \qquad \tilde{m}_j \;:=\; \sum_{l=1}^{j} \dim E(\,\tilde{\lambda}_l\,)$$

and obtain the following relation

$$\underbrace{\lambda_{\tilde{m}_j}}_{=\,\tilde{\lambda}_j} \;<\; \underbrace{\lambda_{\tilde{m}_j+1} \,=\, \ldots \,=\, \lambda_{\tilde{m}_j+\tilde{k}_{j+1}}}_{=\,\tilde{\lambda}_{j+1}} \;<\; \underbrace{\lambda_{\tilde{m}_{j+1}+1}}_{=\,\tilde{\lambda}_{j+2}} \qquad \text{for all } j \in \mathbb{N}. \tag{C.9}$$

Lemma C.3 *Let the assumptions of Theorem C.1 be fulfilled, then it holds*

$$\lim_{j\to\infty} \frac{\tilde{k}_{j+1}}{\tilde{m}_j} \;=\; 0 \qquad \text{and} \qquad \lim_{j\to\infty} \frac{\tilde{k}_j}{\tilde{m}_j} \;=\; 0 \tag{C.10}$$

Proof: Analogically to the proof of the estimates (C.4) it can be shown that there exists for all $\varepsilon > 0$ some $j_0(\varepsilon) \in \mathbb{N}$ such that

$$\left(\frac{j}{C_{\text{Weyl}} + \varepsilon} \right)^{2/d} \leq \; \lambda_j \; \leq \left(\frac{j}{C_{\text{Weyl}} - \varepsilon} \right)^{2/d} \qquad \text{for all } j \geq j_0(\varepsilon). \tag{C.11}$$

From this and (C.9) it follows that for all $j \in \mathbb{N}$ with $\tilde{m}_j + 1 \geq j_0(\varepsilon)$ it holds

$$\left(\frac{\tilde{m}_j + \tilde{k}_{j+1}}{C_{\text{Weyl}} + \varepsilon} \right)^{2/d} \leq \; \lambda_{\tilde{m}_j+\tilde{k}_{j+1}} = \; \lambda_{\tilde{m}_j+1} \; \leq \left(\frac{\tilde{m}_j + 1}{C_{\text{Weyl}} - \varepsilon} \right)^{2/d}. \tag{C.12}$$

If we assume that the left-hand side of (C.10) is not valid, then there exists some constant $\varrho > 0$ and a strictly increasing sequence $(j_n)_{n\in\mathbb{N}} \subset \mathbb{N}$ such that

$$\tilde{k}_{j_n+1} \;>\; \varrho \, \tilde{m}_{j_n}. \tag{C.13}$$

According to (C.12) it has to hold for all j_n with $\tilde{m}_{j_n} + 1 \geq j_0(\varepsilon)$ that

$$\frac{\tilde{m}_{j_n} + \tilde{k}_{j_n+1}}{C_{\text{Weyl}} + \varepsilon} \;\leq\; \frac{\tilde{m}_{j_n} + 1}{C_{\text{Weyl}} - \varepsilon}$$

which is equivalent to

$$\tilde{k}_{j_n+1} \;\leq\; \frac{2\varepsilon}{C_{\text{Weyl}} - \varepsilon}\, \tilde{m}_{j_n} \;+\; \frac{C_{\text{Weyl}} + \varepsilon}{C_{\text{Weyl}} - \varepsilon}. \tag{C.14}$$

However, if we choose ε such that $\varrho \geq (4\varepsilon)/(C_{\text{Weyl}} - \varepsilon)$ and j_n such that

$$\tilde{m}_{j_n} \;\geq\; \max\Big(j_0(\varepsilon) - 1,\, \frac{2}{\varrho}\frac{C_{\text{Weyl}} + \varepsilon}{C_{\text{Weyl}} - \varepsilon} \Big)$$

it follows from (C.13) that

$$\tilde{k}_{j_n+1} \;>\; \varrho\, \tilde{m}_{j_n} \;=\; \frac{\varrho}{2}\, \tilde{m}_{j_n} \;+\; \frac{\varrho}{2}\, \tilde{m}_{j_n} \;\geq\; \frac{2\varepsilon}{C_{\text{Weyl}} - \varepsilon}\, \tilde{m}_{j_n} \;+\; \frac{C_{\text{Weyl}} + \varepsilon}{C_{\text{Weyl}} - \varepsilon}$$

which stands in contradiction to (C.14), i.e., it has to hold $\tilde{k}_{j+1}/\tilde{m}_j \to 0$ for $j \to \infty$. Finally, the right-hand side of (C.10) follows from

$$\frac{\tilde{k}_j}{\tilde{m}_j} \;\leq\; \frac{\tilde{k}_j}{\tilde{m}_{j-1}} \xrightarrow{j\to\infty} 0.$$

■

From the right-hand side of (C.6) it follows that

$$\lim_{j\to\infty} \frac{\tilde{\lambda}_j}{\big(\, N_2(\tilde{\lambda}_j)\,\big)^{2/d}} \;=\; C'_{\text{Weyl}} \qquad \text{with } C'_{\text{Weyl}} := \big(C_{\text{Weyl}}\big)^{-2/d}. \tag{C.15}$$

From the definition of the counting function N_2 we conclude that $N_2(\tilde{\lambda}_j) = \tilde{m}_j$ and we obtain from (C.15) the identity

$$\tilde{\lambda}_j \;=\; C'_{\text{Weyl}}\, \tilde{m}_j^{2/d} \;+\; \tilde{r}(j) \tag{C.16}$$

with some suitable remainder term $\tilde{r}(j) \in \mathrm{o}\big(\tilde{m}_j^{2/d}\big)$ as $j \to \infty$.

Theorem C.4 (Spectral Gap Bounds) *Let the assumptions of Theorem C.1 be fulfilled. If the remainder term $\tilde{r}(j)$ in* (C.16) *fulfils one of the following conditions*

$$i)\quad \tilde{d}(j) \;:=\; \tilde{r}(j) - \tilde{r}(j-1) \;\in\; \mathrm{o}\big(\,\tilde{k}_j\, \tilde{m}_j^{2/d-1}\,\big) \quad \text{as } j \to \infty, \tag{C.17}$$

$$ii)\quad \tilde{r}(j) \;\geq\; \tilde{r}(j-1) \quad \forall\, j \geq j_0 \qquad \text{for some } j_0 \in \mathbb{N}, \tag{C.18}$$

then there exist constants $c_{\mathrm{g}1}, c_{\mathrm{g}2}, c_{\mathrm{g}3} > 0$ independent of j such that the spectral gap can be bounded from below by

$$\Delta_1(\tilde{\lambda}_j) := \min\left\{ \tfrac{\tilde{\lambda}_j - \tilde{\lambda}_{j-1}}{\tilde{\lambda}_j}, \tfrac{\tilde{\lambda}_{j+1} - \tilde{\lambda}_j}{\tilde{\lambda}_j} \right\} \;\geq\; c_{\mathrm{g}1}\, \frac{\min\big\{\dim E(\tilde{\lambda}_j), \dim E(\tilde{\lambda}_{j+1})\big\}}{\tilde{\lambda}_j^{d/2}} \qquad \forall\, j > 1, \tag{C.19}$$

$$\Delta_2(\tilde{\lambda}_j) := \min\left\{ \tfrac{\tilde{\lambda}_j - \tilde{\lambda}_{j-1}}{\tilde{\lambda}_j}, \tfrac{\tilde{\lambda}_{j+1} - \tilde{\lambda}_j}{\tilde{\lambda}_{j+1}} \right\} \;\geq\; c_{\mathrm{g}2}\, \frac{\min\big\{\dim E(\tilde{\lambda}_j), \dim E(\tilde{\lambda}_{j+1})\big\}}{\tilde{\lambda}_j^{d/2}} \qquad \forall\, j > 1, \tag{C.20}$$

$$\Delta_3(\tilde{\lambda}_j) := \tfrac{\tilde{\lambda}_j - \tilde{\lambda}_{j-1}}{\tilde{\lambda}_j} \;\geq\; c_{\mathrm{g}3}\, \frac{\dim E(\tilde{\lambda}_j)}{\tilde{\lambda}_j^{d/2}} \qquad \forall\, j > 1. \tag{C.21}$$

Proof: According to (C.9) we have $\tilde{\lambda}_j = \lambda_{\tilde{m}_j}$ and it follows from Corollary C.2 that

$$c_{\mathrm{b}}\,\tilde{\lambda}_j^{d/2} \,\le\, \tilde{m}_j \,\le\, C_{\mathrm{b}}\,\tilde{\lambda}_j^{d/2} \quad \text{and} \quad \left(\frac{\tilde{m}_j}{C_{\mathrm{b}}}\right)^{2/d} \le\, \tilde{\lambda}_j \,\le \left(\frac{\tilde{m}_j}{c_{\mathrm{b}}}\right)^{2/d} \qquad \text{for all } j \in \mathbb{N}. \tag{C.22}$$

Let be $j > 1$ in the following. Using identity (C.16) and that $\tilde{m}_{j-1} = \tilde{m}_j - \tilde{k}_j$ we obtain

$$\begin{aligned}\frac{\tilde{\lambda}_j - \tilde{\lambda}_{j-1}}{\tilde{\lambda}_j} \underset{(C.22)}{\ge} c_{\mathrm{b}}^{2/d}\, \frac{\tilde{\lambda}_j - \tilde{\lambda}_{j-1}}{\tilde{m}_j^{2/d}} \,&=\, c_{\mathrm{b}}^{2/d}\, C'_{\mathrm{Weyl}}\, \frac{\tilde{m}_j^{2/d} - \tilde{m}_{j-1}^{2/d}}{\tilde{m}_j^{2/d}} \,+\, c_{\mathrm{b}}^{2/d}\, \frac{\tilde{r}(j) - \tilde{r}(j-1)}{\tilde{m}_j^{2/d}} \\ &=\, c_{\mathrm{b}}^{2/d}\, C'_{\mathrm{Weyl}}\, \frac{\tilde{m}_j^{2/d} - (\tilde{m}_j - \tilde{k}_j)^{2/d}}{\tilde{m}_j^{2/d}} \,+\, c_{\mathrm{b}}^{2/d}\, \frac{\tilde{d}(j)}{\tilde{m}_j^{2/d}}.\end{aligned}$$

For $d \in \mathbb{N}$ a Taylor argument leads to

$$(\tilde{m}_j - \tilde{k}_j)^{2/d} \,=\, \tilde{m}_j^{2/d} \,-\, \tfrac{2}{d}\,\tilde{k}_j\,\tilde{m}_j^{2/d-1} \,+\, \tfrac{2-d}{d^2}\,\tilde{k}_j^2\,(\tilde{m}_j - \xi_j)^{2/d-2} \quad \text{for some } \xi_j \in (0, \tilde{k}_j).$$

It follows that

$$\begin{aligned}\frac{\tilde{\lambda}_j - \tilde{\lambda}_{j-1}}{\tilde{\lambda}_j} \,&\ge\, c_{\mathrm{b}}^{2/d} \left(\tfrac{2}{d} C'_{\mathrm{Weyl}}\, \frac{\tilde{k}_j\,\tilde{m}_j^{2/d-1}}{\tilde{m}_j^{2/d}} \,-\, \tfrac{2-d}{d^2} C'_{\mathrm{Weyl}}\, \frac{\tilde{k}_j^2\,(\tilde{m}_j - \xi_j)^{2/d-2}}{\tilde{m}_j^{2/d}} \,+\, \frac{\tilde{d}(j)}{\tilde{m}_j^{2/d}} \right) \\ &=\, c_{\mathrm{b}}^{2/d}\, \frac{\tilde{k}_j}{\tilde{m}_j} \left(\tfrac{2}{d} C'_{\mathrm{Weyl}} \,-\, \tfrac{2-d}{d^2} C'_{\mathrm{Weyl}}\, \frac{\tilde{k}_j\,(\tilde{m}_j - \xi_j)^{2/d-2}}{\tilde{m}_j^{2/d-1}} \,+\, \frac{\tilde{d}(j)}{\tilde{k}_j\,\tilde{m}_j^{2/d-1}} \right)\end{aligned} \tag{C.23}$$

Because of Lemma C.3 there exists some $j_0 \in \mathbb{N}$ such that $\tilde{k}_j \le \tilde{m}_j/2$ for all $j \ge j_0$. Hence, for all $j \ge j_0$ and all $d \in \mathbb{N}$ it follows that $\xi_j < \tilde{k}_j \le \tilde{m}_j/2$ and that

$$\left| \frac{\tilde{k}_j (\tilde{m}_j - \xi_j)^{2/d-2}}{\tilde{m}_j^{2/d-1}} \right| \,\le\, \frac{\tilde{k}_j (\tfrac{1}{2}\tilde{m}_j)^{2/d-2}}{\tilde{m}_j^{2/d-1}} \,=\, \frac{\tilde{k}_j\, 2^{2-2/d}}{\tilde{m}_j} \,<\, 4\frac{\tilde{k}_j}{\tilde{m}_j} \xrightarrow{j\to\infty} 0. \tag{C.24}$$

Correspondingly, if we assume that (C.17) or (C.18) is valid then we obtain from (C.23) by using (C.24) that for sufficiently large j it holds

$$\frac{\tilde{\lambda}_j - \tilde{\lambda}_{j-1}}{\tilde{\lambda}_j} \,\ge\, c_{\mathrm{b}}^{2/d}\, \frac{\tilde{k}_j}{\tilde{m}_j}\, \tfrac{1}{d} C'_{\mathrm{Weyl}} \underset{(C.22)}{\ge} \frac{\tilde{k}_j}{\tilde{\lambda}_j^{d/2}}\, \frac{c_{\mathrm{b}}^{2/d} C'_{\mathrm{Weyl}}}{d\, C_{\mathrm{b}}}.$$

We conclude that there exists some constant $c_{\mathrm{g}3} > 0$ independent of j such that

$$\Delta_3(\tilde{\lambda}_j) \,=\, \frac{\tilde{\lambda}_j - \tilde{\lambda}_{j-1}}{\tilde{\lambda}_j} \,\ge\, c_{\mathrm{g}3}\, \frac{\tilde{k}_j}{\tilde{\lambda}_j^{d/2}} \qquad \text{for all } j > 1$$

and (C.21) is proven. Furthermore, for $j > 0$ it holds

$$\Delta_2(\tilde{\lambda}_j) = \min\Big\{ \Delta_3(\tilde{\lambda}_j), \Delta_3(\tilde{\lambda}_{j+1}) \Big\} \quad \text{and} \quad \Delta_1(\tilde{\lambda}_j) = \min\Big\{ \Delta_3(\tilde{\lambda}_j), \Delta_3(\tilde{\lambda}_{j+1}) \tfrac{\tilde{\lambda}_{j+1}}{\tilde{\lambda}_j} \Big\}. \tag{C.25}$$

Because of Lemma C.3 there exists some $j_0 \in \mathbb{N}$ such that $\tilde{k}_{j+1} \leq \tilde{m}_j$ for all $j \geq j_0$ which leads to

$$\begin{aligned}\Delta_3(\tilde{\lambda}_{j+1}) &\underset{\text{(C.21)}}{\geq} c_{\mathrm{g3}} \frac{\tilde{k}_{j+1}}{\tilde{\lambda}_{j+1}^{d/2}} \underset{\text{(C.22)}}{\geq} c_{\mathrm{g3}} \frac{c_{\mathrm{b}}\, \tilde{k}_{j+1}}{\tilde{m}_{j+1}} = c_{\mathrm{g3}} \frac{c_{\mathrm{b}}\, \tilde{k}_{j+1}}{\tilde{m}_j + \tilde{k}_{j+1}} \\ &\underset{\text{(if } j \geq j_0\text{)}}{\geq} c_{\mathrm{g3}} \frac{c_{\mathrm{b}}\, \tilde{k}_{j+1}}{\tilde{m}_j + \tilde{m}_j} = \frac{c_{\mathrm{g3}}\, c_{\mathrm{b}}}{2} \frac{\tilde{k}_{j+1}}{\tilde{m}_j} \underset{\text{(C.22)}}{\geq} \frac{c_{\mathrm{g3}}\, c_{\mathrm{b}}}{2\, C_{\mathrm{b}}} \frac{\tilde{k}_{j+1}}{\tilde{\lambda}_j^{d/2}}\end{aligned}$$

and

$$\Delta_3(\tilde{\lambda}_{j+1}) \frac{\tilde{\lambda}_{j+1}}{\tilde{\lambda}_j} \;\geq\; \Delta_3(\tilde{\lambda}_{j+1}) \underset{\text{(if } j \geq j_0\text{)}}{\geq} \frac{c_{\mathrm{g3}}\, c_{\mathrm{b}}}{2\, C_{\mathrm{b}}} \frac{\tilde{k}_{j+1}}{\tilde{\lambda}_j^{d/2}}.$$

These estimates combined with (C.21) and (C.25) lead to the statement that there exist constants $c_{\mathrm{g1}}, c_{\mathrm{g2}} > 0$ independent of j such that

$$\Delta_2(\tilde{\lambda}_j) \;\geq\; c_{\mathrm{g2}} \frac{\min\{\, \tilde{k}_j,\, \tilde{k}_{j+1} \,\}}{\tilde{\lambda}_j^{d/2}} \quad \text{and} \quad \Delta_1(\tilde{\lambda}_j) \;\geq\; c_{\mathrm{g1}} \frac{\min\{\, \tilde{k}_j,\, \tilde{k}_{j+1} \,\}}{\tilde{\lambda}_j^{d/2}}.$$

Furthermore, it is noted that — if condition (C.17) is valid — one can proof in an analogical way that there exist constants $C_{\mathrm{g1}}, C_{\mathrm{g2}}, C_{\mathrm{g3}} > 0$ independent of j such that

$$\Delta_3(\tilde{\lambda}_j) \leq C_{\mathrm{g3}} \frac{\tilde{k}_j}{\tilde{\lambda}_j^{d/2}}, \quad \Delta_2(\tilde{\lambda}_j) \leq C_{\mathrm{g2}} \frac{\min\{\tilde{k}_j, \tilde{k}_{j+1}\}}{\tilde{\lambda}_j^{d/2}}, \quad \Delta_1(\tilde{\lambda}_j) \leq C_{\mathrm{g1}} \frac{\min\{\tilde{k}_j, \tilde{k}_{j+1}\}}{\tilde{\lambda}_j^{d/2}}.$$

■

A very easy example where this condition is fulfilled is the Laplace eigenvalue problem (2.17) for $d = 1$. All eigenvalues of this problem are simple and given by $\tilde{\lambda}_j = \lambda_j = j^2\pi^2$. Correspondingly, we have $\tilde{r}(j) = 0$ and hence the spectral gap can be estimated by

$$c_{\mathrm{g}}\, \lambda_j^{-1/2} \;\leq\; \Delta_k(\lambda_j) \;\leq\; C_{\mathrm{g}}\, \lambda_j^{-1/2} \quad \text{for all } j \in \mathbb{N} \qquad \text{(with k=1,2,3).}$$

However, it is quite challenging to prove for a general elliptic PDE eigenvalue problem that the remainder term $\tilde{r}(j) \in o(\tilde{m}_j^{2/d})$ from (C.16) fulfils condition (C.17) or (C.18). In literature typically only asymptotic statements for the counting functions N_1 or N_2 of the form (C.2) can be found. These statements are not sufficient to derive for the difference $\tilde{d}(j) = \tilde{r}(j) - \tilde{r}(j-1)$, for example, a bound of the form $\tilde{d}(j) \in \mathrm{o}\big(\, \tilde{k}_j\, \tilde{m}_j^{2/d-1} \,\big)$. Hence, the asymptotic analysis has to be extended to the difference term $\tilde{d}(j)$ and further research is needed, so far it is only known that $\tilde{d}(j) \in o(\, \tilde{m}_j\, \tilde{m}_j^{2/d-1})$. However, Corollary C.4 shows that the remainder term $\tilde{r}(j)$ has to fulfil certain properties [such as (C.17) or (C.18)] in order to derive sharp lower bounds for the spectral gap. Furthermore, Corollary C.4 gives an orientation what we can possibly expect from the spectral gap.

D. Preliminary Work for Results on the FEM Approximation

This section forms the basis of Section 3.4 where approximation results for the finite element discretisation of eigenvalue problem (2.12) are derived. First of all, the error estimates presented in [64] are summarised and slightly adjusted. These results are combined with the results from Appendix C on the asymptotic behaviour of the continuous eigenvalues and their spectral separation. The resulting approximation properties are finally summarised in Section 3.4, however, the corresponding proofs can be found here.

Starting point of the following discussion is the setting described in Precondition 3.12 from Section 3.4. To derive error estimates for the discretised eigenvalues and eigenfunctions the approximation quality of the finite element space $V_h = \mathbb{X}^p_{h,0}$ has to be measured, more precisely, it has to be measured how good the space V_h is approximating the continuous eigenfunctions. For this purpose the spaces U_j and quantities $\tilde{d}^2(U_j, V_h)$ are introduced for $j = 1, \ldots, N_h$ by

$$U_j := \operatorname{span}\Big\{ u_i \ : \ 1 \leq i \leq j \Big\} \quad \text{and} \quad \tilde{d}^2(U_j, V_h) := \sum_{i=1}^{j} \left(\frac{\|(I - Q_h)u_i\|_1}{\|u_i\|_1} \right)^2 \tag{D.1}$$

where $Q_h : H^1_0(\Omega) \to V_h$ is the $H^1_0(\Omega)$-orthogonal projection onto V_h.

The connection between the finite element discretisation of eigenvalue problem (2.12) and the approximation properties of the underlying finite element space V_h is described by the following theorem.

Theorem D.1 *Consider the variational eigenvalue problem* (2.12) *and its Ritz-Galerkin discretisation* (3.1)*. Let Precondition 3.12 be satisfied and $j \in \{1, \ldots, N_h\}$. Then the relative error of the continuous eigenvalue λ_j and its finite element approximation $\lambda_j^{(h)}$ can be estimated from above by*

$$0 \ \leq \ \frac{\lambda_j^{(h)} - \lambda_j}{\lambda_j^{(h)}} \ \leq \ \tilde{d}^2(U_j, V_h).$$

In the case that it holds $\tilde{d}^2(U_j, V_h) < 1$ for some $j \in \{1, \ldots, N_h\}$ the discretisation error can be estimated by

$$0 \ \leq \ \frac{\lambda_j^{(h)} - \lambda_j}{\lambda_j} \ \leq \ \frac{\tilde{d}^2(U_j, V_h)}{1 - \tilde{d}^2(U_j, V_h)}. \tag{D.2}$$

Proof: The result can be found in [64, Theorem 4.1]. ∎

Estimates for the quantities $\tilde{d}^2(U_j, V_h)$ have been derived in [64] by combining the regularity result of Theorem 2.10 with suitable approximation properties of the underlying finite element space V_h:

Theorem D.2 *Consider the variational eigenvalue problem* (2.12) *and assume that Precondition 3.12 is fulfilled. Let* $C_1 > 0$ *be some given (arbitrary) constant. If the discretisation parameters* h *and* p *of the finite element space* V_h *are chosen such that the condition*

$$\frac{\sqrt{\lambda_j}h}{p} \;\leq\; C_1 \tag{D.3}$$

is fulfilled then it holds

$$\tilde{d}^{\,2}(U_j, V_h) \;\leq\; \frac{C_3}{(\,\min\{\lambda_1, 1\}\,)^2} \sum_{i=1}^{j} \left(\frac{1}{\lambda_i} \left(\frac{C_2 h}{h+\sigma} \right)^{2p} + \left(\frac{\sqrt{\lambda_i} h}{\sigma p} \right)^{2p} \right) \tag{D.4}$$

where $C_2, C_3, \sigma > 0$ *are constants independent*[1] *of* j, h, p.

Proof: The result and the corresponding proof can be found in [64, Theorem 3.2 and Corollary 4.2]. To avoid misunderstandings, it is noted that the constant C_3 differs from the constant C_3 used in [64]. ∎

The combination of Theorem D.1 and Theorem D.2 leads to the following error estimates between the continuous and discrete eigenvalues. It is noted that in [64, Corollary 4.2] quite similar error estimates have been derived already.

Corollary D.3 *Let the assumptions of Theorem D.2 be fulfilled, then it holds:*

i) The quantities $\tilde{d}^{\,2}(U_j, V_h)$ *can be estimated from above by*

$$\tilde{d}^{\,2}(U_j, V_h) \leq C_4\, j \left(\left(\frac{C_2 h}{h+\sigma} \right)^{2p} + \left(\frac{\sqrt{\lambda_j} h}{\sigma p} \right)^{2p} \right) \quad \text{with } C_4 := \frac{C_3}{(\,\min\{\lambda_1, 1\}\,)^3}. \tag{D.5}$$

ii) If the discretisation parameters h *and* p *of the space* V_h *fulfil additionally the condition*

$$h^{2p}\, j\, \lambda_j^p \;\leq\; \frac{1}{2C_5} \qquad \text{with } C_5 := \frac{C_4}{\sigma^{2p}} \left(\frac{C_2^{2p}}{\lambda_1^p} + 1 \right) \tag{D.6}$$

then it holds $\tilde{d}^{\,2}(U_j, V_h) \leq 1/2$ *and the discretisation error of the eigenvalue approximation can be estimated from above by*

$$0 \;\leq\; \frac{\lambda_j^{(h)} - \lambda_j}{\lambda_j} \;\leq\; 2C_4\, j \left(\left(\frac{C_2 h}{h+\sigma} \right)^{2p} + \left(\frac{\sqrt{\lambda_j} h}{\sigma p} \right)^{2p} \right). \tag{D.7}$$

Proof: Estimate (D.5) directly follows from (D.4). Furthermore, we obtain from (D.5) for $p \in \mathbb{N}$

[1] The constants C_2, C_3 depend on the concrete choice of C_1.

and $h > 0$ that

$$\begin{aligned}
\tilde{d}^2(U_j, V_h) \;&\leq\; C_4\, j \left(\left(\frac{C_2 h}{h+\sigma} \right)^{2p} + \left(\frac{\sqrt{\lambda_j} h}{\sigma p} \right)^{2p} \right) \;\leq\; C_4\, j \left(\left(\frac{C_2 h}{\sigma} \right)^{2p} + \left(\frac{\sqrt{\lambda_j} h}{\sigma} \right)^{2p} \right) \\
&=\; C_4\, j \left(\frac{h}{\sigma} \right)^{2p} \lambda_j^p \left(\frac{C_2^{2p}}{\lambda_j^p} + 1 \right) \;\leq\; C_4\, j \left(\frac{h}{\sigma} \right)^{2p} \lambda_j^p \left(\frac{C_2^{2p}}{\lambda_1^p} + 1 \right) \\
&=\; C_5\, h^{2p}\, j\, \lambda_j^p \qquad\qquad \text{with constant } C_5 = \frac{C_4}{\sigma^{2p}} \left(\frac{C_2^{2p}}{\lambda_1^p} + 1 \right).
\end{aligned}$$

We conclude that $\tilde{d}^2(U_j, V_h) \leq 1/2$ if condition (D.6) is valid. Combining this result with (D.2) and (D.5) estimate (D.7) is obtained. It is noted that the constant C_5 depends on λ_1, p and the concrete choice of C_1, apart from that, the constant C_5 is independent of the underlying mesh width h and the eigenvalues $(\lambda_j)_{j=2}^{\infty}$. ■

As noted in Section 3.2 the error analysis of the eigenfunction approximation is more challenging in the case of multiple eigenvalues. The error estimates for the eigenfunction approximation which are presented in the following have been derived in [64]. These estimates are restricted to the special case that all eigenvalues of the continuous problem (2.12) have multiplicity 1, i.e., it is assumed that it holds

$$\lambda_1 < \lambda_2 < \lambda_3 < \dots . \tag{D.8}$$

Theorem D.4 *Consider the variational eigenvalue problem* (2.12) *and its Ritz-Galerkin discretisation* (3.1). *In the following it is assumed that Precondition 3.12 and* (D.8) *are fulfilled. Let $C_1 > 0$ be some given (arbitrary) constant and $j \in \{1, \dots, N_h\}$. Furthermore, let the discretisation parameters h and p of the space V_h be chosen such that the condition*

$$\frac{\sqrt{\lambda_j} h}{p} \;\leq\; C_1 \tag{D.9}$$

is fulfilled, and let in the case that $j > 1$ the parameters h and p be chosen such that it holds

$$\tilde{d}^2(U_{j-1}, V_h) \;\leq\; \frac{1}{2} \frac{\lambda_j - \lambda_{j-1}}{\lambda_j}. \tag{D.10}$$

Then there exists a discrete eigenfunction $\widetilde{u}_j^{(h)} \in E_h\big(\lambda_j^{(h)}\big)$ such that it holds

$$\frac{\|u_j - \widetilde{u}_j^{(h)}\|_1}{\|u_j\|_1} \;\leq\; \frac{\widetilde{C}_3}{\min\{1, \lambda_1\}} \left(1 + \frac{\widetilde{C}_4\, h^{\min\{p,2\}}}{\delta_j} \right) \left(\frac{1}{\sqrt{\lambda_j}} \left(\frac{C_2 h}{h+\sigma} \right)^p + \left(\frac{\sqrt{\lambda_j} h}{\sigma p} \right)^p \right) \tag{D.11}$$

where $C_2, \widetilde{C}_3, \widetilde{C}_4, \sigma > 0$ are constants independent[2] *of j, h, p and where*

$$\delta_j := \min_{i \in \{j_+, j+1\}} \frac{\lambda_i - \lambda_{i-1}}{2 \lambda_i \lambda_{i-1}} \qquad \text{with } j_+ := \max\{j, 2\}.$$

[2]The constants $C_2, \widetilde{C}_3$ depend on the concrete choice of C_1. Furthermore, it is noted that the constants C_2 and σ are the same as in Theorem D.2.

Proof: This result and the corresponding proof can be found in [64, Corollary 5.2]. ■

In Appendix C results on the asymptotic behaviour of the eigenvalues and the spectral gaps have been derived. Using these results the analysis of the finite element discretisation is refined as follows.

Corollary D.5 *Let all assumptions of Theorem D.4 be fulfilled except for condition* (D.10). *Instead it is assumed for $j \in \mathbb{N}$, in view of Theorem C.4, that the spectral gap satisfies*

$$\min_{i\in\{j_+,j+1\}} \frac{\lambda_i - \lambda_{i-1}}{2\lambda_i} \;\geq\; c_6 \lambda_j^{-\frac{d}{2}} \qquad \text{with } j_+ := \max\{j,2\}, \tag{D.12}$$

for some constant $c_6 > 0$ independent of j, and it is assumed that the discretisation parameters h and p of the finite element space V_h are chosen such that the condition

$$h^{2p}\, j\, \lambda_j^{p+\frac{d}{2}} \;\leq\; \frac{c_6}{C_5} \tag{D.13}$$

is fulfilled where the constant C_5 is chosen as in (D.6). *Then the relative error of the eigenfunction approximation can be estimated from above by*

$$\frac{\|u_j - \widetilde{u}_j^{(h)}\|_1}{\|u_j\|_1} \;\leq\; \frac{\widetilde{C}_3}{\min\{1,\lambda_1\}} \left(1 + \frac{\widetilde{C}_4}{c_6}\, \lambda_j^{1+\frac{d}{2}}\, h^{\min\{p,2\}}\right) \left(\frac{1}{\sqrt{\lambda_j}} \left(\frac{C_2 h}{h+\sigma}\right)^p + \left(\frac{\sqrt{\lambda_j} h}{\sigma p}\right)^p\right). \tag{D.14}$$

Proof: In the proof of Corollary D.3 it is shown that

$$\tilde{d}^{\,2}(U_j, V_h) \;\leq\; C_5\, h^{2p}\, j\, \lambda_j^p \;=\; C_5\, \lambda_j^{-\frac{d}{2}}\; h^{2p}\, j\, \lambda_j^{p+\frac{d}{2}}$$

where the constant C_5 is chosen as in (D.6). Correspondingly, if condition (D.12) and (D.13) are fulfilled we obtain

$$\tilde{d}^{\,2}(U_j, V_h) \underset{\text{(D.13)}}{\leq} C_5\, \lambda_j^{-\frac{d}{2}}\; \frac{c_6}{C_5} \underset{\text{(D.12)}}{\leq} \min_{i\in\{j_+,j+1\}} \frac{\lambda_i - \lambda_{i-1}}{2\lambda_i} \;\leq\; \frac{1}{2}\frac{\lambda_j - \lambda_{j-1}}{\lambda_j}$$

with $j_+ = \max\{j,2\}$. From the definition (D.1) we conclude that

$$\tilde{d}^{\,2}(U_{j-1}, V_h) \leq \tilde{d}^{\,2}(U_j, V_h)$$

and it follows that condition (D.10) is fulfilled. From Theorem D.4 it follows that there exist a discrete function $\widetilde{u}_j^{(h)} \in E_h\big(\lambda_j^{(h)}\big)$ with

$$\frac{\|u_j - \widetilde{u}_j^{(h)}\|_1}{\|u_j\|_1} \;\leq\; \frac{\widetilde{C}_3}{\min\{1,\lambda_1\}} \left(1 + \frac{\widetilde{C}_4\, h^{\min\{p,2\}}}{\delta_j}\right) \left(\frac{1}{\sqrt{\lambda_j}} \left(\frac{C_2 h}{h+\sigma}\right)^p + \left(\frac{\sqrt{\lambda_j} h}{\sigma p}\right)^p\right).$$

From the assumption (D.12) on the spectral gap we obtain for δ_j that

$$\delta_j = \min_{i\in\{j_+,j+1\}} \frac{\lambda_i - \lambda_{i-1}}{2\lambda_i\lambda_{i-1}} \;\geq\; \frac{1}{\lambda_j} \min_{i\in\{j_+,j+1\}} \frac{\lambda_i - \lambda_{i-1}}{2\lambda_i} \underset{\text{(D.12)}}{\geq} \frac{1}{\lambda_j}\, c_6\, \lambda_j^{-\frac{d}{2}} \;=\; c_6\, \lambda_j^{-1-\frac{d}{2}}$$

which finally leads to the error estimate (D.14). ■

The results of Corollary D.3 and Corollary D.5 show that various conditions on the discretisation parameters h and p of the finite element space have to be satisfied in order that the corresponding error estimates for the eigenvalue and eigenfunction approximation become valid. For the eigenvalue approximation condition (D.3) and (D.6) are needed, for the eigenfunction approximation condition (D.9) and (D.13). In Corollary 3.14 from Section 3.4 it is shown that these conditions can be summarised. The corresponding proof is as follows.

Proof of Corollary 3.14:

Proof for the Eigenvalue Approximation:

From Corollary D.3 follows that error estimate (3.18) is valid if the discretisation parameters h and p of the space V_h are chosen such that it holds

$$\frac{\sqrt{\lambda_j}h}{p} \;\leq\; C_1 \qquad \text{and} \qquad h^{2p}\, j\, \lambda_j^p \;\leq\; \frac{1}{2C_5} \tag{D.15}$$

where $C_1 > 0$ is an arbitrary chosen constant, and where $C_5 > 0$ is a constant chosen as in (D.6) which depends on λ_1, p and C_1, but which is independent from the mesh width h and the eigenvalues $(\lambda_j)_{j=2}^{\infty}$. We introduce the following new condition

$$h^{2p}\, j\, \lambda_j^p \;\leq\; C_{\mathrm{EV}} \qquad \text{with} \quad C_{\mathrm{EV}} := \min\Big\{C_1^{2p},\, \frac{1}{2C_5}\Big\}. \tag{D.16}$$

If condition (D.16) is fulfilled then it follows that also the right inequality in (D.15) is fulfilled. Furthermore, condition (D.16) can be reformulated as follows

$$\text{(D.16)} \iff h^{2p}\, j\, \lambda_j^p \leq C_{\mathrm{EV}} \iff h\sqrt{\lambda_j} \leq \big(C_{\mathrm{EV}}\, j^{-1}\big)^{\frac{1}{2p}} \iff \frac{h\sqrt{\lambda_j}}{p} \leq \frac{\big(C_{\mathrm{EV}}\, j^{-1}\big)^{\frac{1}{2p}}}{p}.$$

Hence, if (D.16) is valid it holds for $p \in \mathbb{N}$ that

$$\frac{h\sqrt{\lambda_j}}{p} \;\leq\; \frac{\big(C_{\mathrm{EV}}\, j^{-1}\big)^{\frac{1}{2p}}}{p} \;\leq\; \big(C_{\mathrm{EV}}\, j^{-1}\big)^{\frac{1}{2p}} \;\leq\; \big(C_{\mathrm{EV}}\big)^{\frac{1}{2p}} \underset{\text{(D.16)}}{\leq} \big(C_1^{2p}\big)^{\frac{1}{2p}} \;=\; C_1,$$

i.e., the left inequality in (D.15) is fulfilled as well. We conclude if condition (D.16) is satisfied then it follows from Corollary D.3 that error estimate (3.18) is valid. Furthermore, condition (D.16) can be replaced by stronger conditions as follows: According to Corollary C.2 there exist constants $C_{\mathrm{b}}, c_{\mathrm{b}} > 0$ independent of j such that

$$j \;\leq\; C_{\mathrm{b}}\, \lambda_j^{d/2} \qquad \text{and} \qquad \lambda_j \;\leq\; (j/c_{\mathrm{b}})^{2/d} \qquad \text{for all } j \in \mathbb{N}. \tag{D.17}$$

Using these inequalities condition (D.16) can be replaced by the following stronger conditions

$$h^{2p}\, j^{1+\frac{2p}{d}} \;\leq\; C_{\mathrm{idx}}^{\mathrm{EV}} \qquad \text{with} \quad C_{\mathrm{idx}}^{\mathrm{EV}} \;:=\; c_{\mathrm{b}}^{2p/d}\, C_{\mathrm{EV}}, \tag{D.18}$$

$$h^{2p}\, \lambda_j^{p+\frac{d}{2}} \;\leq\; C_{\mathrm{size}}^{\mathrm{EV}} \qquad \text{with} \quad C_{\mathrm{size}}^{\mathrm{EV}} \;:=\; C_{\mathrm{EV}}/C_{\mathrm{b}} \tag{D.19}$$

since it holds

$$h^{2p}\, j\, \lambda_j^p \underset{\text{(D.17)}}{\leq} h^{2p}\, c_{\mathrm{b}}^{-\frac{2p}{d}}\, j^{1+\frac{2p}{d}} \underset{\text{(D.18)}}{\leq} C_{\mathrm{EV}},$$

$$h^{2p}\, j\, \lambda_j^p \underset{\text{(D.17)}}{\leq} h^{2p}\, C_{\mathrm{b}}\, \lambda_j^{p+\frac{d}{2}} \underset{\text{(D.19)}}{\leq} C_{\mathrm{EV}}.$$

Altogether, we conclude that if the mesh width fulfils condition (3.16) or condition (3.17) where the constants $C_{\mathrm{idx}}^{\mathrm{EV}}$ and $C_{\mathrm{size}}^{\mathrm{EV}}$ are chosen sufficiently small [e.g., the constants are chosen as in (D.18) and (D.19)] then it follows that error estimate (3.18) is valid.

Proof for Eigenfunction Approximation:

From Corollary D.5 follows that estimate (3.23) is valid if the spectral gap satisfies condition (3.20), if all continuous eigenvalues have multiplicity 1, and if the discretisation parameters h and p of V_h are chosen such that it holds

$$\frac{\sqrt{\lambda_j}h}{p} \;\leq\; C_1 \qquad \text{and} \qquad h^{2p}\, j\, \lambda_j^{p+\frac{d}{2}} \;\leq\; \frac{c_6}{C_5} \tag{D.20}$$

where $C_1 > 0$ is an arbitrary chosen constant and C_5 is a constant chosen as in (D.6). We introduce the following new condition

$$h^{2p}\, j\, \lambda_j^{p+\frac{d}{2}} \;\leq\; C_{\mathrm{EF}} \qquad \text{with} \quad C_{\mathrm{EF}} := \min\Big\{ C_1^{2p} \lambda_1^{d/2}\,,\, \frac{c_6}{C_5} \Big\}. \tag{D.21}$$

If condition (D.21) is fulfilled then it follows that also the right inequality in (D.20) is fulfilled. Furthermore, (D.21) can be reformulated as

$$\begin{aligned} \text{(D.21)} &\iff h^{2p}\, \lambda_j^p\, j\, \lambda_j^{d/2} \leq C_{\mathrm{EF}} \iff h\sqrt{\lambda_j} \leq \big(C_{\mathrm{EF}}\, j^{-1} \lambda_j^{-d/2}\big)^{\frac{1}{2p}} \\ &\iff \frac{h\sqrt{\lambda_j}}{p} \leq \frac{\big(C_{\mathrm{EF}}\, j^{-1} \lambda_j^{-d/2}\big)^{\frac{1}{2p}}}{p}. \end{aligned}$$

Hence, if (D.21) is valid we follow for $p \in \mathbb{N}$ that

$$\frac{h\sqrt{\lambda_j}}{p} \leq \frac{\Big(C_{\mathrm{EF}}\, j^{-1} \lambda_j^{-\frac{d}{2}}\Big)^{\frac{1}{2p}}}{p} \leq \Big(C_{\mathrm{EF}}\, j^{-1} \lambda_j^{-\frac{d}{2}}\Big)^{\frac{1}{2p}} \leq \Big(C_{\mathrm{EF}}\, \lambda_1^{-\frac{d}{2}}\Big)^{\frac{1}{2p}} \underset{\text{(D.21)}}{\leq} \Big(C_1^{2p} \lambda_1^{\frac{d}{2}}\, \lambda_1^{-\frac{d}{2}}\Big)^{\frac{1}{2p}} = C_1,$$

i.e., the left inequality in (D.20) is fulfilled as well. We conclude if condition (D.21) is satisfied then it follows from Corollary D.5 that error estimate (3.23) is valid. As in the proof for eigenvalue approximation condition the estimates (D.17), are used to replace (D.21) by the following stronger conditions

$$\begin{aligned} h^{2p}\, j^{2+\frac{2p}{d}} \;&\leq\; C_{\mathrm{idx}}^{\mathrm{EF}} \qquad \text{with} \quad C_{\mathrm{idx}}^{\mathrm{EF}} \;:=\; c_{\mathrm{b}}^{1+2p/d}\, C_{\mathrm{EF}}, \\ h^{2p}\, \lambda_j^{p+d} \;&\leq\; C_{\mathrm{size}}^{\mathrm{EF}} \qquad \text{with} \quad C_{\mathrm{size}}^{\mathrm{EF}} \;:=\; C_{\mathrm{EF}}/C_{\mathrm{b}}. \end{aligned}$$

■

Proof of Corollary 3.17: By definition the continuous eigenvalue λ_j is called well approximable by V_h if condition (3.16) or (3.17) is fulfilled. In particular it holds

$$\text{condition (3.16)} \quad \Longleftrightarrow \quad h \;\leq\; \underbrace{\big(C_{\text{idx}}^{\text{EV}}\big)^{\frac{1}{2p}}\, j^{-\frac{2p+d}{2pd}}}_{=:\; h_{\max}^{\text{EV}}(j)} \tag{D.22}$$

$$\Longleftrightarrow \quad \underbrace{\big(C_{\text{idx}}^{\text{EV}}\big)^{\frac{-d}{2p}}\, j^{\frac{2p+d}{2p}}}_{=:\; \widetilde{N}_{\min}^{\text{EV}}(j)} \;\leq\; \frac{1}{h^d} \tag{D.23}$$

$$\Longleftrightarrow \quad j \;\leq\; \underbrace{\big(C_{\text{idx}}^{\text{EV}}\big)^{\frac{d}{2p+d}}\, \left(\frac{1}{h^d}\right)^{\frac{2p}{2p+d}}}_{=:\; j_{\max}^{\text{EV}}(h)} \tag{D.24}$$

where $C_{\text{idx}}^{\text{EV}} > 0$ is some sufficiently small constant. The quantities defined in (D.22) – (D.24) can be interpreted as follows: $h_{\max}^{\text{EV}}(j)$ is the maximal possible mesh width of V_h in order to well approximate the eigenvalue λ_j; since we assume that $N_h \in \Theta(1/h^d)$ the quantity $\widetilde{N}_{\min}^{\text{EV}}(j)$ can be seen as an orientation value for the minimal dimension of V_h in order to well approximate λ_j; and the quantity $\lfloor\, j_{\max}^{\text{EV}}(h)\,\rfloor \in \mathbb{N}$ is the maximal index j such that eigenvalue λ_j is well approximable by V_h. Because of the assumption that $N_h \in \Theta(1/h^d)$, and because of (D.23) and (D.24), it can be easily seen that the asymptotic behaviour of the quantities $N_{\min}^{\text{EV}}(j)$ and $j_{\max}^{\text{EV}}(N_h)$ is described by (3.25) and (3.27). However, for the sake of completeness this result is formally proven in the following.

By definition $N_{\min}^{\text{EV}}(j)$ is the minimal dimension of the finite element space V_h such that eigenvalue λ_j is well approximable, i.e., such that condition (3.16) is fulfilled. If we consider the space V_h where the mesh width h is chosen such that the dimension of V_h fulfils $N_h \geq C\widetilde{N}_{\min}^{\text{EV}}(j)$ we obtain that

$$\widetilde{N}_{\min}^{\text{EV}}(j) \;\leq\; \frac{N_h}{C} \underset{(3.24)}{\leq} \frac{1}{h^d},$$

and from (D.23) it follows that condition (3.16) is fulfilled — i.e., λ_j is well approximable by V_h — and we conclude that $N_{\min}^{\text{EV}}(j) \leq C\widetilde{N}_{\min}^{\text{EV}}(j)$. On the other side, if h' denotes the mesh width of the finite element space $V_{h'}$ with minimal dimension $N_{h'} = N_{\min}^{\text{EV}}(j)$ it has to hold $h' \leq h_{\max}^{\text{EV}}(j)$ because otherwise condition (3.16) is not valid and λ_j would not be well approximable by $V_{h'}$. From

$$\widetilde{N}_{\min}^{\text{EV}}(j) \;=\; \frac{1}{\big(\, h_{\max}^{\text{EV}}(j)\,\big)^d} \;\leq\; \frac{1}{\big(h'\big)^d} \underset{(3.24)}{\leq} \frac{N_{h'}}{c} \;=\; \frac{N_{\min}^{\text{EV}}(j)}{c} \tag{D.25}$$

we follow that $N_{\min}^{\text{EV}}(j) \geq c\widetilde{N}_{\min}^{\text{EV}}(j)$. Using these results and the definition of $\widetilde{N}_{\min}^{\text{EV}}(j)$ we finally conclude that $N_{\min}^{\text{EV}}(j) \;\in\; \Theta\big(\, j^{(2p+d)/(2p)}\,\big)$. In particular, it is noted that a space V_h, whose dimension fulfils $N_h \geq CN_{\min}^{\text{EV}}(j)/c$, is well approximating the eigenvalue λ_j since λ_j is well

approximable when $N_h \geq C\widetilde{N}^{\mathrm{EV}}_{\min}(j)$ and because of

$$N_h \;\geq\; C\,\frac{N^{\mathrm{EV}}_{\min}(j)}{c} \underset{\text{(D.25)}}{\geq} C\,\widetilde{N}^{\mathrm{EV}}_{\min}(j).$$

To analyse the asymptotic behaviour of $j^{\mathrm{EV}}_{\max}(N_h)$ we define the auxiliary quantity

$$\widetilde{j}^{\mathrm{EV}}_{\max}(N_h) \;:=\; \big(C^{\mathrm{EV}}_{\mathrm{idx}}\big)^{\frac{d}{2p+d}}\;(N_h)^{\frac{2p}{2p+d}} \tag{D.26}$$

which can be seen as an orientation value for $j^{\mathrm{EV}}_{\max}(N_h)$. By definition $j^{\mathrm{EV}}_{\max}(N_h)$ is the index of the largest eigenvalue which is well approximable by the finite element space V_h with dimension N_h. If the index j of an eigenvalue λ_j fulfils $j \leq \widetilde{C}\,\widetilde{j}^{\mathrm{EV}}_{\max}(N_h)$ with $\widetilde{C} := C^{-2p/(2p+d)}$ we obtain

$$j \;\leq\; \widetilde{C}\,\widetilde{j}^{\mathrm{EV}}_{\max}(N_h) \underset{\text{(3.24)}}{\overset{\text{(D.26)}}{\leq}} \widetilde{C}\,\big(C^{\mathrm{EV}}_{\mathrm{idx}}\big)^{\frac{d}{2p+d}}\,\left(\frac{C}{h^d}\right)^{\frac{2p}{2p+d}} \underset{\text{(D.24)}}{=} j^{\mathrm{EV}}_{\max}(h).$$

According to (D.24) condition (3.16) is fulfilled — i.e., λ_j is well approximable by V_h — and we conclude that $j^{\mathrm{EV}}_{\max}(N_h) \geq \widetilde{C}\,\widetilde{j}^{\mathrm{EV}}_{\max}(N_h)$. On the other hand, if we choose $j \leq j^{\mathrm{EV}}_{\max}(N_h)$ then by definition eigenvalue λ_j is well approximable, and in particular inequality (D.24) has to be valid. We conclude

$$j^{\mathrm{EV}}_{\max}(N_h) \;\leq\; j^{\mathrm{EV}}_{\max}(h) \underset{\text{(3.24)}}{\overset{\text{(D.24)}}{\leq}} \big(C^{\mathrm{EV}}_{\mathrm{idx}}\big)^{\frac{d}{2p+d}}\,\left(\frac{N_h}{c}\right)^{\frac{2p}{2p+d}} \underset{\text{(D.26)}}{=} \widetilde{c}\,\widetilde{j}^{\mathrm{EV}}_{\max}(N_h)$$

with $\widetilde{c} := c^{-2p/(2p+d)}$, i.e., we have $j^{\mathrm{EV}}_{\max}(N_h) \leq \widetilde{c}\,\widetilde{j}^{\mathrm{EV}}_{\max}(N_h)$. Using these results and using the definition of $\widetilde{j}^{\mathrm{EV}}_{\max}(N_h)$ we finally obtain that $j^{\mathrm{EV}}_{\max}(N_h) \in \Theta\big(N_h^{2p/(2p+d)}\big)$.

In an analogical way we derive from

$$(3.21) \;\Leftrightarrow\; h \leq \big(C^{\mathrm{EF}}_{\mathrm{idx}}\big)^{\frac{1}{2p}}\,j^{-\frac{p+d}{pd}} \;\Leftrightarrow\; \big(C^{\mathrm{EF}}_{\mathrm{idx}}\big)^{\frac{-d}{2p}}\,j^{\frac{p+d}{p}} \leq \frac{1}{h^d} \;\Leftrightarrow\; j \leq \big(C^{\mathrm{EF}}_{\mathrm{idx}}\big)^{\frac{d}{2(p+d)}}\left(\frac{1}{h^d}\right)^{\frac{p}{p+d}},$$

$$(3.17) \;\Leftrightarrow\; h \leq \big(C^{\mathrm{EV}}_{\mathrm{size}}\big)^{\frac{1}{2p}}\,\lambda_j^{-\frac{2p+d}{4p}} \;\Leftrightarrow\; \big(C^{\mathrm{EV}}_{\mathrm{size}}\big)^{\frac{-d}{2p}}\,\lambda_j^{d\frac{2p+d}{4p}} \leq \frac{1}{h^d} \Leftrightarrow\; \lambda \leq \big(C^{\mathrm{EV}}_{\mathrm{size}}\big)^{\frac{2}{2p+d}}\left(\frac{1}{h^d}\right)^{\frac{4p}{d(2p+d)}},$$

$$(3.22) \;\Leftrightarrow\; h \leq \big(C^{\mathrm{EF}}_{\mathrm{size}}\big)^{\frac{1}{2p}}\,\lambda_j^{-\frac{p+d}{2p}} \;\Leftrightarrow\; \big(C^{\mathrm{EF}}_{\mathrm{size}}\big)^{\frac{-d}{2p}}\,\lambda_j^{d\frac{p+d}{2p}} \leq \frac{1}{h^d} \;\Leftrightarrow\; \lambda \leq \big(C^{\mathrm{EF}}_{\mathrm{size}}\big)^{\frac{1}{p+d}}\left(\frac{1}{h^d}\right)^{\frac{2p}{d(p+d)}}$$

the asymptotical behaviour of the minimal dimensions $N^{\mathrm{EF}}_{\min}(j), N^{\mathrm{EV}}_{\min}(\lambda), N^{\mathrm{EF}}_{\min}(\lambda)$; the maximal index $j^{\mathrm{EF}}_{\max}(N_h)$; and the maximal sizes $\lambda^{\mathrm{EV}}_{\max}(N_h)$, $\lambda^{\mathrm{EF}}_{\max}(N_h)$. ■

Bibliography

[1] S. Agmon. Asymptotic formulas with remainder estimates for eigenvalues of elliptic operators. *Arch. Rational Mech. Anal.*, 28:165–183, 1967/1968.

[2] G. M. Amdahl. Validity of the single processor approach to achieving large scale computing capabilities. In *Proceedings of the April 18-20, 1967, Spring Joint Computer Conference*, AFIPS '67 (Spring), pages 483–485, New York, NY, USA, 1967. ACM.

[3] E. Anderson, Z. Bai, C. Bischof, L. S. Blackford, J. Demmel, J. J. Dongarra, J. Du Croz, S. Hammarling, A. Greenbaum, A. McKenney, and D. Sorensen. *LAPACK Users' Guide (Third Ed.)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1999.

[4] W. Arendt, R. Nittka, W. Peter, and F. Steiner. *Weyl's Law: Spectral Properties of the Laplacian in Mathematics and Physics*, pages 1–71. Wiley-VCH Verlag GmbH & Co. KGaA, 2009.

[5] I. Babuška and J. Osborn. Eigenvalue problems. In *Handbook of numerical analysis, Vol. II*, Handb. Numer. Anal., II, pages 641–787. North-Holland, Amsterdam, 1991.

[6] I. Babuška and J. E. Osborn. Estimates for the errors in eigenvalue and eigenvector approximation by Galerkin methods, with particular attention to the case of multiple eigenvalues. *SIAM J. Numer. Anal.*, 24(6):1249–1276, 1987.

[7] I. Babuška and J. E. Osborn. Finite element-Galerkin approximation of the eigenvalues and eigenvectors of selfadjoint problems. *Math. Comp.*, 52(186):275–297, 1989.

[8] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, editors. *Templates for the solution of algebraic eigenvalue problems*, volume 11 of *Software, Environments, and Tools*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000. A practical guide.

[9] L. Banjai, S. Börm, and S. Sauter. FEM for elliptic eigenvalue problems: how coarse can the coarsest mesh be chosen? An experimental study. *Comput. Vis. Sci.*, 11(4-6):363–372, 2008.

[10] K.-J. Bathe. Convergence of subspace iteration. In *Formulations and computational algorithms in finite element analysis (U.S.-Germany Sympos., Mass. Inst. Tech., Cambridge, Mass., 1976)*, pages 575–598. MIT Press, Cambridge, Mass., 1977.

[11] K.-J. Bathe. The subspace iteration method revisited. *Computers & Structures*, 126:177 – 183, 2013. Uncertainty Quantification in structural analysis and design: To commemorate Professor Gerhart I. Schueller for his life-time contribution in the area of computational stochastic mechanics.

[12] M. Bebendorf. Hierarchical *LU* decomposition based preconditioners for BEM. *Computing*, 74:225–247, 2005.

[13] M. Bebendorf and W. Hackbusch. Existence of $\mathcal{H}$-matrix approximants to the inverse FE-matrix of elliptic operators with L^∞-coefficients. *Numerische Mathematik*, 95(1):1–28, 2003.

[14] J. K. Bennighof. Adaptive multi-level substructuring method for acoustic radiation and scattering from complex structures. *Computational Methods for Fluid/Structure Interaction*, 178:25–38, 1993.

[15] J. K. Bennighof, M. F. Kaplan, and M. B. Muller. Extending the frequency response capabilities of automated multi-level substructuring. *AIAA Dynamics Specialists Conference*, 2000. AIAA Paper 2000-1574.

[16] J. K. Bennighof and R. B. Lehoucq. An automated multilevel substructuring method for eigenspace computation in linear elastodynamics. *SIAM J. Sci. Comput.*, 25(6):2084–2106 (electronic), 2004.

[17] S. Börm. *Efficient numerical methods for non-local operators*, volume 14 of *EMS Tracts in Mathematics*. European Mathematical Society (EMS), Zürich, 2010. $\mathcal{H}^2$-matrix compression, algorithms and analysis.

[18] S. Börm, L. Grasedyck, and W. Hackbusch. Introduction to hierarchical matrices with applications. *Engineering Analysis with Boundary Elements*, 27(5):405 – 422, 2003.

[19] S. Börm and K. Reimer. Efficient arithmetic operations for rank-structured matrices based on hierarchical low-rank updates. *Computing and Visualization in Science*, 16(6):247–258, 2013.

[20] F. Bourquin. Analysis and comparison of several component mode synthesis methods on one-dimensional domains. *Numerische Mathematik*, 58:11–34, 1990.

[21] F. Bourquin. Component mode synthesis and eigenvalues of second order operators: Discretization and algorithm. *Mathematical Modeling and Numerical Analysis*, 26:385–423, 1992.

[22] F. Bourquin and F. d'Hennezel. Numerical study of an intrinsic component mode synthesis method. *Comput. Methods Appl. Mech. Engrg.*, 97(1):49–76, 1992.

[23] D. Braess. *Finite Elemente: Theorie, schnelle Löser und Anwendungen in der Elastizitätstheorie*. Springer, 4. überarb. aufl. edition, 2007.

[24] P. Ciarlet. Basic error estimates for elliptic problems. In *Finite Element Methods (Part 1)*, volume 2 of *Handbook of Numerical Analysis*, pages 17 – 351. Elsevier, 1991.

[25] R. Courant and D. Hilbert. *Methoden der mathematischen Physik. I.* Springer-Verlag, Berlin-New York, 1968. Dritte Auflage, Heidelberger Taschenbücher, Band 30.

[26] R. R. Craig and M. C. C. Bampton. Coupling of substructures for dynamic analysis. *AIAA Journal*, 6:1313–1319, 1968.

[27] L. Dagum and R. Menon. Openmp: An industry-standard api for shared-memory programming. *IEEE Comput. Sci. Eng.*, 5(1):46–55, Jan. 1998.

[28] K. Elssel and H. Voss. An a priori bound for automated multilevel substructuring. *SIAM J. Matrix Anal. Appl.*, 28(2):386–397 (electronic), 2006.

[29] M. Faustmann, J. Markus Melenk, and D. Praetorius. $\mathcal{H}$-matrix approximability of the inverses of FEM matrices. *ArXiv e-prints*, Aug. 2013.

[30] W. Gao, X. S. Li, C. Yang, and Z. Bai. An implementation and evaluation of the AMLS method for sparse eigenvalue problems. *ACM Trans. Math. Software*, 34(4):Art. 20, 28, 2008.

[31] P. Gerds and L. Grasedyck. Solving an elliptic pde eigenvalue problem via automated multilevel substructuring and hierarchical matrices. *Computing and Visualization in Science*, 16(6):283–302, 2015.

[32] D. Gilbarg and N. S. Trudinger. *Elliptic partial differential equations of second order*. Classics in Mathematics. Springer-Verlag, Berlin, 2001. Reprint of the 1998 edition.

[33] L. Grasedyck and W. Hackbusch. Construction and arithmetics of $\mathcal{H}$-matrices. *Computing*, 70(4):295–334, 2003.

[34] L. Grasedyck, R. Kriemann, and S. Le Borne. Parallel black box $\mathcal{H}$-LU preconditioning for elliptic boundary value problems. *Comput. Vis. Sci.*, 11(4-6):273–291, 2008.

[35] L. Grasedyck, R. Kriemann, and S. LeBorne. Domain decomposition based $\mathcal{H}$-LU preconditioning. *Numerische Mathematik*, 112(4):565–600, 2009.

[36] R. G. Grimes, J. G. Lewis, and H. D. Simon. A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems. *SIAM J. Matrix Anal. Appl.*, 15(1):228–272, 1994.

[37] W. Hackbusch. *Elliptic differential equations: theory and numerical treatment*, volume 18 of *Springer series in computational mathematics*. Springer, Berlin, 1992.

[38] W. Hackbusch. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$-matrices. *Computing*, 62(2):89–108, 1999.

[39] W. Hackbusch. *Hierarchische Matrizen : Algorithmen und Analysis*. Springer, Dordrecht, 2009.

[40] W. Hackbusch, B. N. Khoromskij, and R. Kriemann. Hierarchical matrices based on a weak admissibility criterion. *Computing*, 73(3):207–243, 2004.

[41] W. Hackbusch, B. N. Khoromskij, S. A. Sauter, and E. E. Tyrtyshnikov. Use of tensor formats in elliptic eigenvalue problems. *Numerical linear algebra with applications*, 19(1):133–151, 2012.

[42] U. Hetmaniuk and R. B. Lehoucq. Multilevel methods for eigenspace computations in structural dynamics. In *In Domain Decomposition Methods in Science and Engineering*, pages 103–114. Springer-Verlag, 2007.

[43] F. Hirsch and G. Lacombe. *Elements of functional analysis*, volume 192 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1999. Translated from the 1997 French original by Silvio Levy.

[44] W. C. Hurty. Vibrations of structural systems by component-mode synthesis. *Journal of the Engineering Mechanics Division*, 86:51–69, 1960.

[45] Intel. Intel math kernel library. `https://software.intel.com/en-us/intel-mkl/`.

[46] Intel. Threading building blocks. `https://www.threadingbuildingblocks.org/`.

[47] M. F. Kaplan. *Implementation of automated multi-level substructuring for frequency response analysis of structures*. Ph.d. thesis, Universtiy of Texas at Austin, 2001.

[48] A. V. Knyazev. New estimates for ritz vectors. *Math. Comput.*, 66(219):985–995, July 1997.

[49] L. G. . W. H. . R. Kriemann. Performance of $\mathcal{H}$-LU preconditioning for sparse matrices. *Computational methods in applied mathematics*, 8(4):336–349, 2008.

[50] R. Kriemann. HLIBpro. `http://www.hlibpro.com/`.

[51] R. Kriemann. *Parallele Algorithmen fr $\mathcal{H}$-Matrizen*. Dissertation, Universitt Kiel, 2004.

[52] R. Kriemann. Parallel $\mathcal{H}$-matrix arithmetics on shared memory systems. *Computing*, 74(3):273–297, 2005.

[53] R. Kriemann. HLIBpro C language interface. Technical report, Max Planck Institute for Mathematics in the Sciences, 2008.

[54] R. Kriemann. $\mathcal{H}$-LU factorization on many-core systems. *Computing and visualization in science*, 16(3):105–117, 2013.

[55] A. Kropp and D. Heiserer. Efficient broadband vibro-acoustic analysis of passenger car bodies using an fe-based component mode synthesis approach. *Journal of Computational Acoustics*, 11(02):139–157, 2003.

[56] C. L. Lawson, R. J. Hanson, D. R. Kincaid, and F. T. Krogh. Basic linear algebra subprograms for fortran usage. *ACM Trans. Math. Softw.*, 5(3):308–323, Sept. 1979.

[57] R. B. Lehoucq, C.-C. Yang, and D. C. Sorensen. *ARPACK users' guide : solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*. SIAM, Philadelphia, 1998.

[58] S. Levendorskiĭ. *Asymptotic distribution of eigenvalues of differential operators*, volume 53 of *Mathematics and its Applications (Soviet Series)*. Kluwer Academic Publishers Group, Dordrecht, 1990. Translated from the Russian.

[59] M. Lintner. The eigenvalue problem for the 2d laplacian in h-matrix arithmetic and application to the heat and wave equation. *Computing*, 72(3-4):293–323, May 2004.

[60] A. Quarteroni and A. Valli. *Domain decomposition methods for partial differential equations*. Numerical Mathematics and Scientific Computation. The Clarendon Press Oxford University Press, New York, 1999. Oxford Science Publications.

[61] K. Reimer. *$\mathcal{H}^2$-Matrix-Arithmetik basierend auf Niedrigrangupdates.* Dissertation, Universitt Kiel, 2015.

[62] J. Reinders. *Intel Threading Building Blocks.* O'Reilly & Associates, Inc., Sebastopol, CA, USA, first edition, 2007.

[63] H. Rutishauser. Computational aspects of F. L. Bauer's simultaneous iteration method. *Numer. Math.*, 13(1):4–13, 1969.

[64] S. Sauter. hp-finite elements for elliptic eigenvalue problems: Error estimates which are explicit with respect to lambda, h, and p. *SIAM J. Numerical Analysis*, 48(1):95–108, 2010.

[65] P. Seshu. Substructuring and component mode synthesis. *Shock and Vibration*, 4:199–210, 1997.

[66] H. Weyl. Ueber die asymptotische verteilung der eigenwerte. *Nachrichten von der Gesellschaft der Wissenschaften zu Gttingen, Mathematisch-Physikalische Klasse*, 1911:110–117, 1911.

[67] C. Yang, W. Gao, Z. Bai, X. S. Li, L.-Q. Lee, P. Husbands, and E. Ng. An algebraic substructuring method for large-scale eigenvalue calculation. *SIAM J. Sci. Comput.*, 27(3):873–892 (electronic), 2005.

[68] J. Yin, H. Voss, and P. Chen. Improving eigenpairs of automated multilevel substructuring with subspace iterations. *Computers & Structures*, 119:115 – 124, 2013.