Chapter 1
Introduction

The goal of this book is to describe a method for handling certain large dense matrices efficiently. The fundamental idea of the $\mathcal{H}^2$-matrix approach is to reduce the storage requirements by using an alternative multilevel representation of a dense matrix instead of the standard representation by a two-dimensional array.

1.1 Origins of $\mathcal{H}^2$-matrix methods

The need for efficient algorithms for handling dense matrices arises from several fields of applied mathematics: in the simulation of many-particle systems governed by the laws of gravitation or electrostatics, a fast method for computing the forces acting on the individual particles is required, and these forces can be expressed by large dense matrices.

Certain homogeneous partial differential equations can be reformulated as boundary integral equations, and compared to the standard approach, these formulations have the advantage that they reduce the spatial dimension, improve the convergence and can even simplify the handling of complicated geometries. The discretization of the boundary integral equations leads again to large dense matrices.

A number of models used in the fields of population dynamics or machine learning also lead to integral equations that, after discretization, yield large dense matrices.

Several approaches for handling these kinds of problems have been investigated: for special integral operators and special geometries, the corresponding dense matrices are of Toeplitz or circulant form, and the fast Fourier transform [37] can be used to compute the matrix-vector multiplication in $\mathcal{O}(n \log n)$ operations, where n is the matrix dimension. The restriction to special geometries limits the range of applications that can be treated by this approach.

The panel clustering method [71], [72], [91], [45] follows a different approach to handle arbitrary geometries: the matrix is not represented exactly, but approximated by a data-sparse matrix, i.e., by a matrix that is still dense, but can be represented in a compact form. This approximation is derived by splitting the domain of integration into a partition of subdomains and replacing the kernel function by local separable approximations. The resulting algorithms have a complexity of $\mathcal{O}(n m^\alpha \log^\beta n)$ for problem-dependent small exponents $\alpha, \beta > 0$ and a parameter m controlling the accuracy of the approximation.

The well-known multipole method [58], [60] is closely related and takes advantage of the special properties of certain kernel functions to improve the efficiency. It has originally been introduced for the simulation of many-particle systems, but can also be applied to integral equations [88], [86], [85], [57]. "Multipole methods without multipoles" [2], [82], [108] replace the original multipole approximation by more general or computationally more efficient expansions while keeping the basic structure of the corresponding algorithms. Of particular interest is a fully adaptive approach [46] that constructs approximations based on singular value decompositions of polynomial interpolants and thus can automatically find efficient approximations for relatively general kernel functions.

It should be noted that the concept of separable approximations used in both the panel clustering and the multipole method is already present in the Ewald summation technique [44] introduced far earlier to evaluate Newton potentials in crystallographical research efficiently.

Wavelet techniques use a hierarchy of nested subspaces combined with a Galerkin method in order to approximate integral operators [94], [9], [41], [39], [73], [105], [102]. This approach reaches very good compression rates, but the construction of suitable subspaces on complicated geometries is significantly more complicated than for the techniques mentioned before.

Hierarchical matrices [62], [68], [67], [49], [52], [63] and the closely related mosaic skeleton matrices [103] are the algebraic counterparts of panel-clustering and multipole methods: a partition of the matrix takes the place of the partition of the domains of integration, and low-rank submatrices take the place of local separable expansions. Due to their algebraic structure, hierarchical matrices can be applied not only to integral equations and particle systems, but also to more general problems, e.g., partial differential equations [6], [56], [55], [54], [76], [77], [78] or matrix equations from control theory [53], [51]. Efficient approximations of densely populated matrices related to integral equations can be constructed by interpolation [16] or more efficient cross approximation schemes [5], [4], [7], [17].

$\mathcal{H}^2$-matrices [70], [64] combine the advantages of hierarchical matrices, i.e., their flexibility and wide range of applications, with those of wavelet and fast multipole techniques, i.e., the high compression rates achieved by using a multilevel basis. The construction of this *cluster basis* for different applications is one of the key challenges in the area of $\mathcal{H}^2$-matrices: it has to be efficient, i.e., it has to consist of a small number of vectors, but it also has to be accurate, i.e., it has to be able to approximate the original matrix up to a given tolerance. In some situations, an $\mathcal{H}^2$-matrix approximation can reach the *optimal* order $\mathcal{O}(n)$ of complexity while keeping the approximation error consistent with the requirements of the underlying discretization scheme [91], [23].

Obviously, we cannot hope to be able to approximate all dense matrices in this way: if a matrix contains only independent random values, the standard representation is already optimal and no compression scheme will be able to reduce the storage requirements. Therefore we have first to address the question "Which kinds of matrices can be compressed by $\mathcal{H}^2$-matrix methods?"

It is not sufficient to know that a matrix can be compressed, we also have to be able to find the compressed representation and to use it in applications, e.g., to perform matrix-vector multiplications or solve systems of linear equations. Of course, we do not want to convert the $\mathcal{H}^2$-matrices back to the less efficient standard format, therefore we have to consider the question "Which kinds of operations can be performed efficiently with compressed matrices?"

Once these two theoretical questions have been answered, we can consider practical applications of the $\mathcal{H}^2$-matrix technique, i.e., try to answer the question "Which problems can be solved efficiently by $\mathcal{H}^2$-matrices?"

1.2 Which kinds of matrices can be compressed?

There are two answers to this question: in the introductory Chapter 2, a very simple one-dimensional integral equation is discussed, and it is demonstrated that its discrete counterpart can be handled by $\mathcal{H}^2$-matrices: if we replace the kernel function by a *separable approximation*, the resulting matrix will be an $\mathcal{H}^2$-matrix and can be treated efficiently. Chapter 4 generalizes this result to the more general setting of integral operators with asymptotically smooth kernel functions.

In Chapter 6, on the other hand, a relatively general characterization of $\mathcal{H}^2$-matrices is introduced. Using this characterization, we can determine whether arbitrary matrices can be approximated by $\mathcal{H}^2$-matrices. In this framework, the approximation of integral operators can be treated as a special case, but it is also possible to investigate more general applications, e.g., the approximation of solution operators of ordinary [59], [96] and elliptic partial differential equations by $\mathcal{H}^2$-matrices [6], [15]. The latter very important case is treated in Chapter 9.

Separable approximations

Constructing an $\mathcal{H}^2$-matrix based on separable approximations has the advantage that the problem is split into two relatively independent parts: the first task is to approximate the kernel function in suitable subdomains by separable kernel functions. This task can be handled by Taylor expansions [72], [100] or interpolation [45], [65], [23] if the kernel function is locally analytic. Both of these approaches are discussed in Chapter 4.

For special kernel functions, special approximations like the multipole expansion [58], [60] or its counterparts for the Helmholtz kernel [1], [3] can be used. The special techniques required by these methods are not covered here.

Once a good separable approximation of the kernel function has been found, we face the second task: the construction of an $\mathcal{H}^2$-matrix. This is accomplished by splitting the integral operator into a sum of local operators on suitably defined subsets and then replacing the original kernel function by its separable approximations. Discretizing the resulting perturbed integral operator by a standard scheme (e.g., Galerkin methods, collocation or Nystrøm techniques) then yields an $\mathcal{H}^2$-matrix approximation of the original matrix.

The challenge in this task is to ensure that the number of local operators is as small as possible: using one local operator for each matrix entry will not lead to a good compression ratio, therefore we are looking for methods that ensure that only a small number of local operators are required.

The standard approach in this context is to use *cluster trees*, i.e., to split the domains defining the integral operator into a hierarchy of subdomains and use an efficient recursive scheme to find an almost optimal decomposition of the original integral operator into local operators which can be approximated.

The efficiency of this technique depends on the properties of the discretization scheme. If the supports of the basis functions are local, i.e., if a neighborhood of the support of a basis function intersects only a small number of supports of other basis functions, it can be proven that the cluster trees will lead to efficient approximations of the matrix [52]. For complicated anisotropic meshes or higher-order basis functions, the situation becomes more complicated and special techniques have to be employed.

General characterization

Basing the construction of an $\mathcal{H}^2$-matrix on the general theory presented in Chapter 6 has the advantage that it allows us to treat arbitrary dense matrices. Whether a matrix can be approximated by an $\mathcal{H}^2$-matrix or not can be decided by investigating the effective ranks of two families of submatrices, the *total cluster bases*. If all of these submatrices can be approximated using low ranks, the matrix itself can be approximated by an $\mathcal{H}^2$-matrix.

Since this characterization relies only on low-rank approximations, but requires no additional properties, it can be applied in relatively general situations, e.g., to prove that solution operators of strongly elliptic partial differential operators with L^∞ coefficients can be approximated by $\mathcal{H}^2$-matrices. Chapter 9 gives the details of this result.

1.3 Which kinds of operations can be performed efficiently?

In this book, we consider three types of operations: first the construction of an approximation of the system matrix, then arithmetic operations like matrix-vector and matrix-matrix multiplications, and finally more complicated operations like matrix factorizations or matrix inversion, which can be constructed based on the elementary arithmetic operations.

Construction

An $\mathcal{H}^2$-matrix can be constructed in several ways: if it is the approximation of an explicitly given integral operator, we can proceed as described above and compute the

$\mathcal{H}^2$-matrix by discretizing a number of local separable approximations. For integral operators with locally smooth kernel functions, the implementation of this method is relatively straightforward and it performs well. This approach is described in Chapter 4.

If we want to approximate a given matrix, we can use the compression algorithms introduced in Chapter 6. These algorithms have the advantage that they construct quasi-optimal approximations, i.e., they will find an approximation that is almost as good as the best possible $\mathcal{H}^2$-matrix approximation. This property is very useful, since it allows us to use $\mathcal{H}^2$-matrices as a "black box" method.

It is even possible to combine both techniques: if we want to handle an integral operator, we can construct an initial approximation by using the general and simple interpolation scheme, and then improve this approximation by applying the appropriate compression algorithm. The experimental results in Chapter 6 indicate that this technique can reduce the storage requirements by large factors.

Arithmetic operations

If we want to solve a system of linear equations with a system matrix in $\mathcal{H}^2$-representation, we at least have to be able to evaluate the product of the matrix with a vector. This and related operations, like the product with the transposed matrix or forward and backward substitution steps for solving triangular systems, can be accomplished in optimal complexity for $\mathcal{H}^2$-matrices: not more than two operations are required per unit of storage.

Using Krylov subspace methods, it is even possible to construct solvers based exclusively on matrix-vector multiplications and a number of simple vector operations. This is the reason why most of today's schemes for solving dense systems of equations (e.g., based on panel clustering [72], [91] or multipole expansions [58], [60]) provide only efficient algorithms for matrix-vector multiplications, but not for more complicated operations.

Hierarchical matrices and $\mathcal{H}^2$-matrices, on the other hand, are purely algebraic objects, and since we have efficient compression algorithms at our disposal, we are able to approximate the results of complex operations like the matrix-matrix multiplication. In Chapters 7 and 8, two techniques for performing this fundamental computation are presented. The first one reaches the optimal order of complexity, but requires a priori knowledge of the structure of the result. The second one is slightly less efficient, but has the advantage that it is fully adaptive, i.e., that it is possible to guarantee a prescribed accuracy of the result.

Inversion and preconditioners

Using the matrix-matrix multiplication algorithms, we can perform more complicated arithmetic operations like the inversion or the LU factorization. The derivation of the corresponding algorithms is straightforward: if we express the result in terms of block matrices, we see that it can be computed by a sequence of matrix-matrix multiplications. We replace each of these products by its $\mathcal{H}^2$-matrix approximation and combine all of the $\mathcal{H}^2$-submatrices to get an $\mathcal{H}^2$-matrix approximation of the result (cf. Section 6.7 and Chapter 10).

If we perform all operations with high accuracy, the resulting inverse or factorization can be used as a direct solver for the original system, although it may require a large amount of storage. If we use only a low accuracy, we can still expect to get a good preconditioner which can be used in an efficient iterative or semi-iterative scheme, e.g., a conjugate gradient or GMRES method.

1.4 Which problems can be solved efficiently?

In this book, we focus on dense matrices arising from the discretization of integral equations, especially those connected to solving homogeneous elliptic partial differential equations with the boundary integral method. For numerical experiments, these matrices offer the advantage that they are discretizations of a continuous problem, therefore we have a scale of discretizations of differing resolution at our disposal and can investigate the behavior of the methods for very large matrices and high condition numbers. The underlying continuous problem is relatively simple, so we can easily construct test cases and verify the correctness of an implementation.

We also consider the construction of approximate inverses for the stiffness matrices arising from finite element discretizations of elliptic partial differential operators. In the paper [6], it has been proven that these inverses can be approximated by hierarchical matrices [62], [52], [63], but the proof is based on a global approximation argument that does not carry over directly to the case of $\mathcal{H}^2$-matrices. Chapter 9 uses the localized approach presented in [15] to construct low-rank approximations of the total cluster bases, and applying the general results of Chapter 6 and [13] yields the existence of $\mathcal{H}^2$-matrix approximations.

$\mathcal{H}^2$-matrices have also been successfully applied to problems from the field of electromagnetism [24], heat radiation, and machine learning.

1.5 Organization of the book

In the following, I try to give an overview of the current state of the field of $\mathcal{H}^2$-matrices. The presentation is organized in nine chapters covering basic definitions, algorithms with corresponding complexity analysis, approximation schemes with corresponding error analysis, and a number of numerical experiments.

Chapter 2: Model problem This chapter introduces the basic concepts of $\mathcal{H}^2$-matrices for a one-dimensional model problem. In this simple setting, the construction of an $\mathcal{H}^2$-matrix and the analysis of its complexity and approximation properties is fairly straightforward.

Chapter 3: Hierarchical matrices This chapter considers the generalization of the definition of $\mathcal{H}^2$-matrices to the multi-dimensional setting. $\mathcal{H}^2$-matrices are defined based on a *block cluster tree* describing a partition of the matrix into a hierarchy of submatrices and *cluster bases* describing the form of these submatrices. If a number of relatively general conditions for the block cluster tree and the cluster bases are fulfilled, it is possible to derive optimal-order estimates for the storage requirements and the time needed to compute the matrix-vector multiplication

Chapter 4: Integral operators A typical application of $\mathcal{H}^2$-matrices is the approximation of matrices resulting from the finite element (or boundary element) discretization of integral operators. This chapter describes simple approximation schemes based on Taylor expansion and constant-order interpolation, but also more advanced approaches based on variable-order interpolation. The error of the resulting $\mathcal{H}^2$-matrices is estimated by using error bounds for the local approximants of the kernel function.

Chapter 5: Orthogonal cluster bases This chapter describes techniques for finding the optimal $\mathcal{H}^2$-matrix approximation of a given arbitrary matrix under the assumption that a suitable block cluster tree and good cluster bases are already known. If the cluster bases are *orthogonal*, the construction of the optimal approximation is straightforward, therefore this chapter contains two algorithms for converting arbitrary cluster bases into orthogonal cluster bases: the first algorithm yields an orthogonal cluster basis that is equivalent to the original one, the second algorithm constructs an approximation of lower complexity.

Chapter 6: Compression This chapter introduces the *total cluster bases* that allow us to give an alternative characterization of $\mathcal{H}^2$-matrices and to develop algorithms for approximating arbitrary matrices. The analysis of these algorithms relies on the results of Chapter 5 in order to establish quasi-optimal error estimates.

Chapter 7: A priori matrix arithmetic Once a matrix has been approximated by an $\mathcal{H}^2$-matrix, the question of solving corresponding systems of linear equations has to be answered. For dense matrices, the usual solution strategies require factorizations of the matrix or sometimes even its inverse. Since applying these techniques directly to $\mathcal{H}^2$-matrices would be very inefficient, this chapter introduces an alternative: factorization and inversion can be performed using matrix-matrix products, therefore finding an efficient algorithm for approximating these products is an important step towards solving linear systems. By using the orthogonal projections introduced in Chapter 5 and preparing suitable quantities in advance, the best approximation of a matrix-matrix product in a given $\mathcal{H}^2$-matrix space can be computed very efficiently.

Chapter 8: A posteriori matrix arithmetic The algorithms introduced in Chapter 7 compute the best approximation of the matrix-matrix product in a *given* matrix space, but if this space is not chosen correctly, the resulting error can be quite large. This chapter describes an alternative algorithm that constructs an $\mathcal{H}^2$-matrix approximation of the matrix-matrix product and chooses the cluster bases in such a way that a given precision can be guaranteed.

Chapter 9: Elliptic partial differential equations Based on the a posteriori arithmetic algorithms of Chapter 8, it is possible to compute approximate inverses of $\mathcal{H}^2$-matrices, but it is not clear whether these inverses can be represented efficiently by an $\mathcal{H}^2$-matrix. This chapter proves that the inverse of the stiffness matrix of an elliptic partial differential equation can indeed be approximated well in the compressed format, and due to the best-approximation property of the compression algorithm, this means that the computation can be carried out efficiently.

Chapter 10: Applications The final chapter considers a number of practical applications of $\mathcal{H}^2$-matrices. Most of the applications are related to boundary integral formulations for Laplace's equation, but there are also some examples related to more general elliptic partial differential equations.

In some chapters, I have collected technical lemmas in a separate section in the hope of focusing the attention on the important results, not on often rather technical proofs of auxiliary statements.