Max-Planck-Institut
für Mathematik
in den Naturwissenschaften
Leipzig

Approximation of integral operators by $\mathcal{H}^2$-matrices with adaptive bases

(revised version: December 2004)

by

Steffen Börm

Preprint no.: 18 2004

Approximation of integral operators by $\mathcal{H}^2$-matrices with adaptive bases

Steffen Börm, Leipzig

December 10, 2004

$\mathcal{H}^2$-matrices can be used to construct efficient approximations of discretized integral operators. The $\mathcal{H}^2$-matrix approximation can be constructed efficiently by interpolation, Taylor or multipole expansion of the integral kernel function, but the resulting representation requires a large amount of storage.

In order to improve the efficiency, local Schur decompositions can be used to eliminate redundant functions from an original approximation, which leads to a significant reduction of storage requirements and algorithmic complexity.

1 Introduction

We consider an integral operator

$$\mathcal{K}[u](x) = \int_\Omega \kappa(x,y)u(y)\,dy \tag{1}$$

defined by a domain or manifold $\Omega \subseteq \mathbb{R}^d$ and a kernel function $\kappa : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. Discretizing $\mathcal{K}$ by Galerkin's method with finite element basis functions $(\varphi_i)_{i\in\mathcal{I}}$ leads to a matrix $K \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}$ given by

$$K_{ij} = \langle \varphi_i, \mathcal{K}[\varphi_j]\rangle_{L^2} = \int_\Omega \varphi_i(x) \int_\Omega \kappa(x,y)\varphi_j(y)\,dy\,dx.$$

In many applications the support of κ is a superset of $\Omega\times\Omega$, so the matrix K will be dense. Treating K directly leads to a dense matrix that requires $\mathcal{O}(n^2)$ units of storage, where $n := \#\mathcal{I}$ is the number of degrees of freedom.

Since the quadratic complexity of a simple approach is not acceptable if the problem dimension becomes large, a variety of alternative representations have been introduced: If the kernel function is asymptotically smooth (cf. [7]), it can be approximated by panel-clustering (cf. [14, 17, 16]) or multipole expansion (cf. [15, 11, 12]) methods. A similar effect can be achieved by replacing the finite element basis functions $(\varphi_i)_{i\in\mathcal{I}}$ by wavelet-like functions (cf. [8]).

The algorithm presented here creates an approximation of the kernel function by means of a panel-clustering algorithm based on interpolation [9, 3]. The expansion systems corresponding to interpolation contain a certain degree of redundancy that has to be eliminated in order to improve the efficiency. A simple way of doing this is to orthogonalize the discrete expansion systems [4]. These techniques take into account the effect of the discretization, but not the influence of the discrete operator that is being approximated.

An alternative approach is the algebraic approximation algorithm presented in [2], which constructs locally optimal discrete expansion systems for arbitrary operators. If the operator is already approximated by a suitable panel-clustering approach, the algorithm can take advantage of this more efficient representation in order to reach linear complexity in time and storage requirements.

2 $\mathcal{H}^2$-matrix approximation by interpolation

We will now introduce the basic notations for $\mathcal{H}^2$-matrices and demonstrate how interpolation can be used to construct an approximation of an integral operator (cf. [3]).

2.1 Local interpolation of the kernel function

In order to be able to approximate the kernel function κ by interpolation, we have to ensure that κ restricted to the domain of interpolation is sufficiently smooth. For asymptotically smooth kernel functions, i.e., if there are $c_0 \in \mathbb{R}_{>0}, g \in \mathbb{N}_0$ such that

$$|\partial_x^\alpha \partial_y^\beta \kappa(x,y)| \lesssim \frac{c_0^{|\alpha+\beta|}(\alpha+\beta)!}{|x-y|^{g+|\alpha|+|\beta|}} \tag{2}$$

holds for all $\alpha, \beta \in \mathbb{N}_0^d$, a simple criterion can be found: Let $B^t, B^s \subseteq \mathbb{R}^d$ be axis-parallel bounding boxes and let $\mathcal{I}_m^t$ and $\mathcal{I}_m^s$ be stable m-th order interpolation operators on B^t or B^s, respectively. If the *admissibility condition*

$$\operatorname{diam}(B^t \times B^s) \leq \eta \operatorname{dist}(B^t, B^s) \tag{3}$$

holds, then an error estimate of the form

$$\|\kappa - (\mathcal{I}^t \otimes \mathcal{I}^s)[\kappa]\|_{\infty, B^t \times B^s} \lesssim \frac{1}{\operatorname{dist}(B^t, B^s)^g} \left(\frac{c_2\eta}{c_2\eta + c_1}\right)^m \tag{4}$$

can be proven for stable interpolation schemes (like Chebyshev interpolation, cf. [5]), so the interpolation converges exponentially if the order m is increased.

Remark 1 *In many applications, derivatives of the kernel function κ are used instead of κ, e.g., when treating the double layer potential operator. Using techniques similar to those used in [6, Lemma 6.1], we can prove that the derivatives of κ and its interpolant also satisfy an estimate similar to (4) and therefore will also converge exponentially.*

The idea of most panel-clustering and multipole techniques is to split the domain $\Omega \times \Omega$ into a hierarchy of subdomains that satisfy the admissibility condition (3) and a small remainder of subdomains that still contain the singularity, but can be handled by a sparse matrix. On the admissible domains, a separable approximation is used to derive a more efficient representation.

In order to construct the admissible subdomains efficiently, we introduce a hierarchy of subdomains corresponding to subsets of the index set $\mathcal{I}$: Let $T_{\mathcal{I}}$ be a labeled tree with root r, i.e., a tree in which each node $t \in T_{\mathcal{I}}$ is associated with a label $\hat{t}$. $T_{\mathcal{I}}$ is a *cluster tree* if the following conditions hold:

- The root of $T_{\mathcal{I}}$ is $\mathcal{I}$, i.e., $\hat{r} = \mathcal{I}$.
- If $t \in T_{\mathcal{I}}$ has sons, then the labels of the sons form a partition of the label of the father, i.e., $\hat{t} = \dot{\bigcup}\{\hat{s} \ : \ s \in \operatorname{sons}(t)\}$.
- For all $t \in T_{\mathcal{I}}$, we have $\#\operatorname{sons}(t) \neq 1$ and $\#\hat{t} > 0$.

The nodes $t \in T_{\mathcal{I}}$ of a cluster tree $T_{\mathcal{I}}$ are called *clusters*, and the *level* of a cluster $t \in T_{\mathcal{I}}$ is defined inductively by $\operatorname{level}(r) = 0$ and $\operatorname{level}(t') = \operatorname{level}(t) + 1$ for $t' \in \operatorname{sons}(t)$. The number of all clusters is denoted by $c := \#T_{\mathcal{I}}$ and satisfies $c \leq 2n - 1$.

Typically, we will assume that the leaves of $T_{\mathcal{I}}$ correspond to small subsets of $\mathcal{I}$, i.e., that there is a constant $C_{\text{leaf}} \in \mathbb{N}$ such that $\#\hat{t} \leq C_{\text{leaf}}$ holds for all leaves of $T_{\mathcal{I}}$.

Using the cluster tree, we can construct the desired partition: For each cluster t, we fix a *bounding box* $B^t \subseteq \mathbb{R}^d$, i.e., an axis-parallel box satisfying $\operatorname{supp} \varphi_i \subseteq B^t$ for all $i \in \hat{t}$. For a given pair $(t, s) \in T_{\mathcal{I}} \times T_{\mathcal{I}}$, we proceed as follows:

- If B^t and B^s satisfy the admissibility condition (3), we add (t, s) to the set P_{far}.
- If t and s are leaves, we add (t, s) to the set P_{near}.
- Otherwise, we repeat the procedure for pairs formed by the sons of t and s (if only one of the clusters has no sons, we use the cluster itself instead).

Starting with $P_{\text{far}} = P_{\text{near}} = \emptyset$ and $(t, s) = (r, r)$, we get a partition $P = P_{\text{far}} \dot{\cup} P_{\text{near}}$ of $\mathcal{I} \times \mathcal{I}$ such that all pairs in P_{far} correspond to admissible subdomains and all pairs in P_{near} correspond to leaf clusters.

In standard situations, it can be shown that $\#P \in \mathcal{O}(n)$ holds (cf. [10]).

2.2 Approximation of the matrix

The approximation of the matrix K can now be constructed block by block: For each $(t, s) \in P_{\text{far}}$, we interpolate κ on $B^t \times B^s$ and set

$$\tilde{K}_{ij}^{t,s} := \begin{cases} \int_\Omega \varphi_i(x) \int_\Omega (\mathcal{I}_m^t \otimes \mathcal{I}_m^s)[\kappa](x, y) \varphi_j(y) \, dy \, dx & \text{if } i \in \hat{t}, j \in \hat{s} \\ 0 & \text{otherwise.} \end{cases}$$

for all $i, j \in \mathcal{I}$. Since $(t, s) \in P_{\text{far}}$, we know that $\tilde{K}_{ij}^{t,s}$ will converge rapidly to K_{ij} if the interpolation order m is increased.

All nearfield entries are collected in a matrix $K^* \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}$

$$K^*_{ij} := \begin{cases} K_{ij} & \text{if there is a pair } (t,s) \in P_{\text{near}} \text{ with } i \in \hat{t}, j \in \hat{s} \\ 0 & \text{otherwise.} \end{cases}$$

for all $i, j \in \mathcal{I}$. Since $P = P_{\text{far}} \dot\cup P_{\text{near}}$ describes a partition of $\mathcal{I} \times \mathcal{I}$, the matrix

$$\tilde{K} := K^* + \sum_{(t,s)\in P_{\text{far}}} \tilde{K}^{t,s} \tag{5}$$

is an approximation of K.

We can handle K^* efficiently since it is a sparse matrix. The farfield matrices $\tilde{K}^{t,s}$ require a different storage format that we will introduce now. For all $t \in T_{\mathcal{I}}$, we denote the interpolation points in B^t by $(x^t_\nu)_{\nu=1}^k$ and the corresponding Lagrange polynomials by $(\mathcal{L}^t_\nu)_{\nu=1}^k$. For $i \in \hat{t}$, $j \in \hat{s}$, we find

$$\begin{aligned} \tilde{K}^{t,s}_{ij} &= \int_\Omega \varphi_i(x) \int_\Omega \left(\sum_{\nu=1}^k \sum_{\mu=1}^k \kappa(x^t_\nu, x^s_\mu) \mathcal{L}^t_\nu(x) \mathcal{L}^s_\mu(y) \right) \varphi_j(y)\, dy\, dx \\ &= \sum_{\nu=1}^k \sum_{\mu=1}^k \kappa(x^t_\nu, x^s_\mu) \int_\Omega \varphi_i(x) \mathcal{L}^t_\nu(x)\, dx \int_\Omega \varphi_j(y) \mathcal{L}^s_\mu(y)\, dy, \end{aligned}$$

so we can store $\tilde{K}^{t,s}$ in the factorized form

$$\tilde{K}^{t,s} = V^t S^{t,s} (W^s)^\top \tag{6}$$

with the *row cluster basis matrix* $V^t \in \mathbb{R}^{\mathcal{I}\times k}$, the *column cluster basis matrix* $W^s \in \mathbb{R}^{\mathcal{I}\times k}$ and the *coefficient matrix* $S^{t,s} \in \mathbb{R}^{k\times k}$ defined by

$$V^t_{i\nu} = W^t_{i\nu} := \begin{cases} \int_\Omega \varphi_i(x) \mathcal{L}^t_\nu(x)\, dx & \text{if } i \in \hat{t} \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

$$S^{t,s}_{\nu\mu} := \kappa(x^t_\nu, x^s_\mu). \tag{8}$$

In the case of integral operators of the form (1), the row and column cluster basis matrices V^t and W^t are identical. In more general applications they may differ, e.g., if the operator is based on derivatives of κ.

The factorized representation of $\tilde{K}^{t,s}$ requires only $(\#\hat{t} + \#\hat{s} + k)k$ units of storage.

2.3 Nested cluster bases

We call the families $V = (V^t)_{t\in T_{\mathcal{I}}}$ and $W = (W^s)_{s\in T_{\mathcal{I}}}$ of row and column cluster basis matrices *row* and *column cluster bases*. Storing all V^t as dense matrices leads to a storage complexity of $\mathcal{O}(nkp)$, where p is the depth of the cluster tree. Usually, $p \sim \log n$ will hold, so the storage requirements will not scale linearly in n. We will now introduce an alternative representation of V^t that allows us to reach the optimal order of complexity.

Let $t \in T_{\mathcal{I}}$ be a cluster with $\operatorname{sons}(t) \neq \emptyset$, and let $t' \in \operatorname{sons}(t)$. The interpolation operator $\mathcal{I}_m^{t'}$ is a projection into the space of m-th order polynomials. Since any Lagrange polynomial $\mathcal{L}_\mu^t$ for the cluster t is in this space, we have

$$\mathcal{L}_\mu^t = \mathcal{I}_m^{t'}[\mathcal{L}_\mu^t] = \sum_{\nu=1}^{k} \mathcal{L}_\mu^t(x_\nu^{t'})\mathcal{L}_\nu^{t'},$$

i.e., we can express each Lagrange polynomial of the father cluster t in terms of the Lagrange polynomials of the son cluster t'. We introduce the *transfer matrix* $T^{t'} \in \mathbb{R}^{k\times k}$ by setting

$$T_{\nu\mu}^{t'} = \mathcal{L}_\mu^t(x_\nu^{t'}) \tag{9}$$

and notice that

$$\mathcal{L}_\mu^t = \sum_{\nu=1}^{k} T_{\nu\mu}^{t'}\mathcal{L}_\nu^{t'}$$

implies

$$V_{i\nu}^t = \sum_{\mu=1}^{k} V_{i\mu}^{t'}T_{\mu\nu}^{t'} = (V^{t'}T^{t'})_{i\nu}$$

for all $i \in \hat{t}' \subseteq \hat{t}$. Since the index sets corresponding to the sons of t are a disjoint partition of $\hat{t}$, we can sum over all sons in order to get

$$V^t = \sum_{t'\in\operatorname{sons}(t)} V^{t'}T^{t'}. \tag{10}$$

This relation between father and son clusters implies that we have to store the cluster bases V^t only for leaves $t \in T_{\mathcal{I}}$ of the cluster tree and can use the small $k \times k$ transfer matrices T^t to describe the bases for all other clusters.

This alternative representation of the matrix $\tilde{K}$ requires $\mathcal{O}(ck^2 + nk)$ units of storage. By truncating the cluster tree, we can ensure $c \sim n/k$ and reach a complexity of $\mathcal{O}(nk)$.

An approximation of the form (5) with blocks defined by (6) is called a *uniform $\mathcal{H}$-matrix*. If the cluster bases are nested, it is called an *$\mathcal{H}^2$-matrix* [13, 2, 3].

Since we are using d-dimensional interpolation of order m, we require a rank of $k \sim m^d$. In many applications, this is not optimal: If the kernel function is the Newton kernel $\kappa(x, y) = 1/(4\pi\|x - y\|)$ in $\mathbb{R}^3$, a multipole expansion [15, 12] requires only $k \sim m^2$ spherical harmonics instead of $\mathcal{O}(m^3)$ polynomials.

In order to improve the efficiency, we will now remove redundant functions from the expansion system, i.e., we will use interpolation only as an "initial guess" and reduce k by algebraic methods while keeping the good approximation properties and general applicability.

3 Orthogonalization

A simple approach to reducing redundancy is to compare the dimension of the range of the cluster basis matrices V^t with the number of columns of V^t, i.e., the number of

expansion functions involved. If the dimension is lower than the number of columns, the expansion system obviously contains superfluous functions that can be eliminated without changing the quality of the approximation.

Seen from this point of view, an *orthogonal* cluster basis matrix, i.e., a matrix satisfying $(V^t)^\top V^t = I$, is optimal: Since the columns are pairwise perpendicular, no column can be eliminated without changing the range of V^t.

3.1 Leaf clusters

A viable strategy for eliminating redundant functions from the expansion system is to perform a Gram-Schmidt orthogonalization: We are looking for a matrix $Z^t \in \mathbb{R}^{k \times k^t}$ such that the new cluster basis matrix

$$\tilde{V}^t := V^t Z^t$$

is orthogonal, i.e., satisfies the equation

$$(\tilde{V}^t)^\top \tilde{V}^t = (V^t Z^t)^\top V^t Z^t = I \in \mathbb{R}^{k^t \times k^t}.$$

Using $G^t = (V^t)^\top V^t$, this equation takes the form

$$(Z^t)^\top G^t Z^t = I. \tag{11}$$

Different methods can be used to find a suitable Z^t: One possibility is to compute a rank-revealing Cholesky decomposition of G^t, which would be equivalent to the classical Gram-Schmidt procedure. In order to be able to detect redundant functions in a more direct fashion, we will use the Schur decomposition $G^t = PDP^\top$ of G^t instead, i.e., we will compute an orthogonal matrix P containing the eigenvectors of G^t and a diagonal matrix $D = \operatorname{diag}\{\lambda_1, \ldots, \lambda_k\}$ containing the corresponding eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_k \geq 0$.

We fix a rank $k^t \in \{1, \ldots, k\}$ such that $\lambda_i > 0$ holds for all $i \in \{1, \ldots, k^t\}$, define the matrix $\tilde{D} \in \mathbb{R}^{k \times k^t}$ by $\tilde{D}_{ij} := \frac{\delta_{ij}}{\sqrt{\lambda_i}}$ and set $Z^t := P\tilde{D}$. Since

$$(Z^t)^\top G^t Z^t = \tilde{D}^\top P^\top P D P^\top P \tilde{D} = \tilde{D}^\top D \tilde{D}^\top = I$$

holds, we have found a suitable solution Z^t of problem (11).

Creating the matrix G^t requires $\mathcal{O}(k^2 \#\hat{t})$ operations, solving the symmetric eigenvalue problem takes $\mathcal{O}(k^3)$ operations, and Z^t can be constructed in $\mathcal{O}(k^2 k^t)$ operations, so the total complexity is bounded by $\mathcal{O}(k^2(k + k^t + \#\hat{t}))$.

Any choice of $k^t \in \{0, \ldots, \operatorname{rank}(V^t)\}$ will give us a matrix Z^t satisfying (11). Using $k^t < \operatorname{rank}(V^t)$ implies that the range of $\tilde{V}^t = V^t Z^t$ will be a proper subspace of the range of V^t, so the quality of the approximation may be reduced. Since we are using the Schur decomposition to construct the approximation, we have all eigenvalues of G^t at our disposal and can derive the following error estimate:

Lemma 2 (Local truncation error) *We have*

$$\inf\{\|V^t - \tilde{V}^t R\|_F^2 \ : \ R \in \mathbb{R}^{k^t \times k}\} = \|V^t - \tilde{V}^t(\tilde{V}^t)^\top V^t\|_F^2 = \epsilon_F^t := \sum_{i=k^t+1}^{k} \lambda_i \quad \textit{and}$$

$$\inf\{\|V^t - \tilde{V}^t R\|_2^2 \ : \ R \in \mathbb{R}^{k^t \times k}\} = \|V^t - \tilde{V}^t(\tilde{V}^t)^\top V^t\|_2^2 = \epsilon_2^t := \lambda_{k^t+1}.$$

Proof. Since $\tilde{V}^t$ is orthogonal, $\tilde{V}^t(\tilde{V}^t)^\top$ is an orthogonal projection in both the Euclidean and the Frobenius norm, so the best approximation of V^t is given by $\tilde{V}^t(\tilde{V}^t)^\top V^t$.

By definition of Z^t, we have

$$\begin{aligned} V^t - \tilde{V}^t(\tilde{V}^t)^\top V^t &= V^t(I - Z^t(Z^t)^\top (V^t)^\top V^t) = V^t(I - Z^t(Z^t)^\top G^t) \\ &= V^t(I - P\tilde{D}\tilde{D}^\top P^\top PDP^\top) = V^t P(I - \tilde{D}\tilde{D}^\top D)P^\top. \end{aligned}$$

A simple computation reveals

$$(\tilde{D}\tilde{D}^\top D)_{ij} = \begin{cases} \delta_{ij} & \text{if } i \leq k^t \\ 0 & \text{otherwise,} \end{cases}$$

so we find

$$\begin{aligned} \|V^t - \tilde{V}^t(\tilde{V}^t)^\top V^t\|_F^2 &= \|V^t P(I - \tilde{D}\tilde{D}^\top D)P^\top\|_F^2 \\ &= \operatorname{trace}\Big((I - \tilde{D}\tilde{D}^\top D)^\top P^\top (V^t)^\top V^t P(I - \tilde{D}\tilde{D}^\top D)\Big) \\ &= \operatorname{trace}\left(\operatorname{diag}\{0, \ldots, 0, \lambda_{k^t+1}, \ldots, \lambda_k\}\right) = \sum_{i=k^t+1}^{k} \lambda_i. \end{aligned}$$

The estimate for the operator norm can be derived by replacing the trace operator by the spectral radius, since $\varrho(X^\top X) = \|X\|_2^2$. ∎

In practical applications, it is sometimes a good idea to choose the original rank k to be larger than $\#\hat{t}$. In this case, the rank of G^t would be bounded by $m := \min\{k, \#\hat{t}\}$ and computing the Schur decomposition of G^t would give us at least $k - m > 0$ vanishing eigenvalues that are irrelevant to the orthogonalization process.

In order to avoid this, we can use Householder transformations to compute a decomposition $V^t|_{\hat{t}\times k} = LQ$ of $V^t|_{\hat{t}\times k}$ into a lower triangular (with respect to an arbitrary, but fixed, ordering of $\hat{t}$) matrix $L \in \mathbb{R}^{\hat{t}\times k}$ and a unitary matrix $Q \in \mathbb{R}^{k\times k}$ that can be represented by a product of m elementary reflectors.

Since L is lower triangular, only its first m columns will differ from zero, so only the upper left $m \times m$ block of the transformed Gram matrix $QG^tQ^\top = L^\top L$ will contain non-zero entries. In order to find the relevant eigenvalues and eigenvectors of G^t, we can therefore transform by Q, compute the Schur decomposition of the upper $m \times m$ block and reverse the transformation.

Computing the LQ decomposition requires $\mathcal{O}(km^2)$ operations, solving the reduced eigenvalue problem requires $\mathcal{O}(m^3)$ operations, transforming the k^t relevant eigenvectors requires $\mathcal{O}(mkk^t)$ operations and creating Z^t requires $\mathcal{O}(mkk^t)$ operations, so we can construct the matrix Z^t by a total of $\mathcal{O}(m^2k + m^3 + mkk^t) \subseteq \mathcal{O}(m^2(k+m))$ operations, which is significantly better than the original $\mathcal{O}(k^2\#\hat{t} + k^3 + k^2k^t)$ if k is larger than $\#\hat{t}$.

3.2 Non-leaf clusters

Applying the Gram-Schmidt procedure directly to all clusters V^t will yield orthogonal cluster bases $\tilde{V}^t$, but these bases will no longer be nested. Since this property is very important for the efficiency of the algorithm, we will now modify the construction in order to ensure that the orthogonalized cluster bases are again nested.

We assume that orthogonal cluster bases have already been constructed for all $t' \in \operatorname{sons}(t)$ (e.g., by a recursive procedure). A nested cluster basis is characterized by (10), so we have to be able to find transfer matrices $\tilde{T}^{t'}$ for all $t' \in \operatorname{sons}(t)$ such that

$$\tilde{V}^t = \sum_{t' \in \operatorname{sons}(t)} \tilde{V}^{t'} \tilde{T}^{t'}$$

holds. This implies that the range of $\tilde{V}^t$ is a subspace of

$$\widehat{\mathcal{V}}^t := \bigoplus_{t' \in \operatorname{sons}(t)} \operatorname{range} \tilde{V}^{t'} \subseteq \mathbb{R}^{\mathcal{I}}.$$

Therefore, we replace V^t by the best approximation satisfying this condition, i.e., by its orthogonal projection

$$\widehat{V}^t := \left(\sum_{t' \in \operatorname{sons}(t)} \tilde{V}^{t'} (\tilde{V}^{t'})^\top \right) V^t. \tag{12}$$

Since V^t is already nested, we can use (10) in order to get

$$\widehat{V}^t = \left(\sum_{t' \in \operatorname{sons}(t)} \tilde{V}^{t'} (\tilde{V}^{t'})^\top \right) V^t = \sum_{t' \in \operatorname{sons}(t)} \tilde{V}^{t'} (\tilde{V}^{t'})^\top V^{t'} T^{t'} = \sum_{t' \in \operatorname{sons}(t)} \tilde{V}^{t'} P^{t'} T^{t'},$$

where the matrices $P^{t'} := (\tilde{V}^{t'})^\top V^{t'}$ describe the transformation from the original to the new bases.

For any matrix $Z^t \in \mathbb{R}^{k \times k^t}$, a cluster basis defined by $\tilde{V}^t = \widehat{V}^t Z^t$ will still satisfy

$$\tilde{V}^t = \widehat{V}^t Z^t = \sum_{t' \in \operatorname{sons}(t)} \tilde{V}^{t'} P^{t'} T^{t'} Z^t = \sum_{t' \in \operatorname{sons}(t)} \tilde{V}^{t'} \tilde{T}^{t'}$$

with

$$\tilde{T}^{t'} := P^{t'} T^{t'} Z^t. \tag{13}$$

This means that we can proceed as in the case of leaf clusters and choose a matrix Z^t satisfying

$$(Z^t)^\top \widehat{G}^t Z^t = I \tag{14}$$

for the modified Gram matrix

$$\widehat{G}^t = \sum_{t' \in \operatorname{sons}(t)} (\tilde{V}^{t'} P^{t'} T^{t'})^\top (\tilde{V}^{t'} P^{t'} T^{t'}) = \sum_{t' \in \operatorname{sons}(t)} (P^{t'} T^{t'})^\top (\tilde{V}^{t'})^\top \tilde{V}^{t'} (P^{t'} T^{t'})$$

$$= \sum_{t' \in \operatorname{sons}(t)} (P^{t'} T^{t'})^\top (P^{t'} T^{t'}).$$

This matrix Z^t defines the transfer matrices of the orthogonal nested cluster basis we are looking for by equation (13).

In order to form $\widehat{G}^t$, we need the matrices $P^{t'} = (\tilde{V}^{t'})^\top V^{t'}$. For the leaf clusters, they can be constructed directly. For non-leaf clusters, we can use the nested structure of both original and new cluster bases to derive the following update equation:

$$P^t = (\tilde{V}^t)^\top V^t = \left(\sum_{t' \in \operatorname{sons}(t)} \tilde{V}^{t'} \tilde{T}^{t'} \right)^\top \left(\sum_{t' \in \operatorname{sons}(t)} V^{t'} T^{t'} \right) = \sum_{t' \in \operatorname{sons}(t)} (\tilde{T}^{t'})^\top P^{t'} T^{t'}. \quad (15)$$

For leaf clusters, we applied an LQ-decomposition in the case $\#t < k$ in order to reduce the complexity. We can do the same in the case of non-leaf clusters: Let $s := \#\operatorname{sons}(t)$ and let $\operatorname{sons}(t) = \{t_1, \ldots, t_s\}$. Then $\widehat{G}^t$ can be written in the form

$$\widehat{G}^t = (X^t)^\top X^t$$

with

$$X^t := \begin{pmatrix} P^{t_1} T^{t_1} \\ \vdots \\ P^{t_s} T^{t_s} \end{pmatrix}. \quad (16)$$

The auxiliary matrix X^t has $q := k^{t_1} + \ldots + k^{t_s}$ rows and k columns. If $q < k$ holds, we can again use Householder transformations to compute an LQ-decomposition $X^t = LQ$ of X^t and solve an m-dimensional eigenvalue problem with $m := \min\{q, k\}$ instead of a k-dimensional one.

We can compute $\widehat{G}^t$ in $\mathcal{O}(qk^2)$, find the matrix Z^t in $\mathcal{O}(m^2k + m^3 + mkk^t) \subseteq \mathcal{O}(m^2(k+m))$, and construct P^t in $\mathcal{O}(q(k+k^t)k) \subseteq \mathcal{O}(qk^2)$ operations, so the total complexity for orthogonalizing V^t is $\mathcal{O}(qk^2 + m^2(k+m))$.

Lemma 3 (Local truncation error) *Let $\lambda_1 \geq \ldots \geq \lambda_k \geq 0$ be the eigenvalues of $\widehat{G}$. Then we have*

$$\inf\{\|\widehat{V}^t - \tilde{V}^t R\|_F^2 \ : \ R \in \mathbb{R}^{k^t \times k}\} = \|\widehat{V}^t - \tilde{V}^t (\tilde{V}^t)^\top \widehat{V}^t\|_F^2 = \epsilon_F^t := \sum_{i=k^t+1}^{k} \lambda_i \quad \textit{and}$$

$$\inf\{\|\widehat{V}^t - \tilde{V}^t R\|_2^2 \ : \ R \in \mathbb{R}^{k^t \times k}\} = \|V^t - \tilde{V}^t (\tilde{V}^t)^\top \widehat{V}^t\|_2^2 = \epsilon_2^t := \lambda_{k^t+1}.$$

Proof. Apply Lemma 2 to $\widehat{V}^t$. ∎

3.3 Global properties

We define the set of descendants of a cluster $t \in T_{\mathcal{I}}$ by

$$\operatorname{sons}^*(t) := \{s \in T_{\mathcal{I}} \ : \ \hat{s} \subseteq \hat{t}\}$$

and corresponding transfer matrices by

$$T^{s,t} := \begin{cases} I & \text{if } \operatorname{sons}(t) = \emptyset \\ T^{s,t'} T^{t'} & \text{if } t' \in \operatorname{sons}(t) \text{ with } \hat{s} \subseteq \hat{t}' \end{cases}$$

for all $s \in \operatorname{sons}^*(t)$. Due to the definition of $\operatorname{sons}^*(t)$ and the cluster tree $T_{\mathcal{I}}$, $T^{s,t}$ is well-defined.

We can combine the local estimates to get a global bound for the truncation error:

Theorem 4 (Truncation error) *We have*

$$\|V^t - \tilde{V}^t (\tilde{V}^t)^\top V^t\|_F^2 \leq \sum_{s \in \operatorname{sons}^*(t)} \epsilon_F^s \|T^{s,t}\|_2^2 \quad \text{and} \tag{17}$$

$$\|V^t - \tilde{V}^t (\tilde{V}^t)^\top V^t\|_2^2 \leq \sum_{s \in \operatorname{sons}^*(t)} \epsilon_2^s \|T^{s,t}\|_2^2 \tag{18}$$

for all $t \in T_{\mathcal{I}}$. Here, ϵ_F^s, ϵ_2^s are defined as in Lemma 2 for leaf clusters and as in Lemma 3 for non-leaf clusters.

Proof. The proof is split into three parts: We start by proving that the local error in a cluster is perpendicular with respect to all of its ancestors, then we represent the global error in terms of the local errors, and we conclude by demonstrating that we can apply Pythagoras' equality in order to get a bound for the global error.

Step 1: We denote the local errors by

$$E^t := \begin{cases} V^t - \tilde{V}^t (\tilde{V}^t)^\top V^t & \text{if } \operatorname{sons}(t) = \emptyset \\ \widehat{V}^t - \tilde{V}^t (\tilde{V}^t)^\top \widehat{V}^t & \text{otherwise} \end{cases}$$

for all $t \in T_{\mathcal{I}}$. We will start by proving that

$$\langle \tilde{V}^t x, E^s y \rangle = 0 \tag{19}$$

holds for all $s, t \in T_{\mathcal{I}}$ with $\hat{s} \subseteq \hat{t}$ and all $x \in \mathbb{R}^{k^t}$, $y \in \mathbb{R}^k$. We do this by induction over $\operatorname{level}(s) - \operatorname{level}(t) = n \in \mathbb{N}_0$. The case $\operatorname{level}(s) = \operatorname{level}(t)$ is trivial, since it implies $s = t$.

Let $n \in \mathbb{N}_0$. We assume that (19) holds for all $s, t \in T_{\mathcal{I}}$ with $\operatorname{level}(s) - \operatorname{level}(t) = n$ and $\hat{s} \subseteq \hat{t}$. Let $s, t \in T_{\mathcal{I}}$ with $\operatorname{level}(s) - \operatorname{level}(t) = n + 1$ and $\hat{s} \subseteq \hat{t}$. The assumption implies $\operatorname{sons}(t) \neq \emptyset$, so there is a $t' \in \operatorname{sons}(t)$ with $\hat{s} \subseteq \hat{t}'$, and $\operatorname{level}(s) - \operatorname{level}(t') = n$. We apply the induction assumption to get

$$\langle \tilde{V}^t x, E^s y \rangle = \sum_{t'' \in \operatorname{sons}(t)} \langle \tilde{V}^{t''} \tilde{T}^{t''} x, E^s y \rangle = \langle \tilde{V}^{t'} \tilde{T}^{t'} x, E^s y \rangle = 0,$$

which completes the induction.

Step 2: We denote the global errors by

$$\widehat{E}^t := V^t - \tilde{V}^t (\tilde{V}^t)^\top V^t.$$

By definition, we have $\widehat{E}^t = E^t$ for all $t \in T_{\mathcal{I}}$ with $\operatorname{sons}(t) = \emptyset$, i.e., for all leaf clusters.

For non-leaf clusters, we can use

$$\begin{aligned}(\tilde{V}^t)^\top V^t &= (Z^t)^\top (\widehat{V}^t)^\top V^t = (Z^t)^\top \sum_{t' \in \operatorname{sons}(t)} (P^{t'} T^{t'})^\top (\tilde{V}^{t'})^\top V^{t'} T^{t'} \\ &= (Z^t)^\top \sum_{t' \in \operatorname{sons}(t)} (P^{t'} T^{t'})^\top P^{t'} T^{t'} = (\tilde{V}^t)^\top \widehat{V}^t\end{aligned}$$

in order to prove

$$\begin{aligned}\widehat{E}^t &= V^t - \widehat{V}^t + \widehat{V}^t - \tilde{V}^t (\tilde{V}^t)^\top V^t \\ &= \sum_{t' \in \operatorname{sons}(t)} (V^{t'} - \tilde{V}^{t'} (\tilde{V}^{t'})^\top V^{t'}) T^{t'} + \widehat{V}^t - \tilde{V}^t (\tilde{V}^t)^\top \widehat{V}^t \\ &= \sum_{t' \in \operatorname{sons}(t)} \widehat{E}^{t'} T^{t'} + E^t,\end{aligned}$$

so the recurrence relation

$$\widehat{E}^t = \begin{cases} E^t & \text{if } \operatorname{sons}(t) = \emptyset \\ E^t + \sum_{t' \in \operatorname{sons}(t)} \widehat{E}^{t'} T^{t'} & \text{otherwise} \end{cases}$$

holds for all $t \in T_{\mathcal{I}}$. A simple induction using the definitions of $T^{s,t}$ and $\widehat{E}^t$ yields

$$\widehat{E}^t = \sum_{s \in \operatorname{sons}^*(t)} E^s T^{s,t} \tag{20}$$

for all $t \in T_{\mathcal{I}}$.

Step 3: In order to be able to apply Pythagoras' equality to the norm of (20), we have to establish that the ranges of the terms $E^s T^{s,t}$ appearing in this sum are orthogonal, i.e., that

$$\langle E^{s_1} x, E^{s_2} y \rangle = 0 \tag{21}$$

holds for all $s_1, s_2 \in \operatorname{sons}^*(t)$ with $s_1 \neq s_2$ and all $x, y \in \mathbb{R}^k$.

If $\hat{s}_1 \cap \hat{s}_2 = \emptyset$ holds, this follows directly from the definition of V^{s_1}, $\tilde{V}^{s_1}$, V^{s_2} and $\tilde{V}^{s_2}$: For all $i \in \hat{s}_1$, we have $i \notin \hat{s}_2$, so the i-th row of V^{s_2} and $\tilde{V}^{s_2}$ vanishes, and therefore the entire inner product also vanishes.

If $\hat{s}_1 \cap \hat{s}_2 \neq \emptyset$ holds, we have either $s_1 \in \operatorname{sons}^*(s_2)$ or $s_2 \in \operatorname{sons}^*(s_1)$. Since both cases are similar, we consider only the latter one. Our assumptions imply $\operatorname{sons}(s_1) \neq \emptyset$, i.e.,

$$E^{s_1} x = \widehat{V}^{s_1} x - \tilde{V}^{s_1} (\tilde{V}^{s_1})^\top \widehat{V}^{s_1} x = \widehat{V}^{s_1} x - \widehat{V}^{s_1} Z^{s_1} (\tilde{V}^{s_1})^\top \widehat{V}^{s_1} x = \widehat{V}^{s_1} (I - Z^{s_1} (\tilde{V}^{s_1})^\top \widehat{V}^{s_1}) x.$$

Now we can apply (19) to prove (21), and Pythagoras' equality yields

$$\|\widehat{E}^s\|_F^2 = \sum_{s \in \operatorname{sons}^*(t)} \|E^s T^{s,t}\|_F^2 \quad \text{and} \quad \|\widehat{E}^s\|_2^2 = \sum_{s \in \operatorname{sons}^*(t)} \|E^s T^{s,t}\|_2^2.$$

Using the submultiplicativity of the norms and the error bounds from Lemma 2 and Lemma 3 concludes the proof. ∎

In order to apply this general error estimate to our case, we need a bound for the operator norm of the matrices $T^{s,t}$. By a simple induction based on the definition (9), we find

$$T^{s,t}_{\nu\mu} = \mathcal{L}^t_\mu(x^s_\nu).$$

If the interpolation operator is stable with stability constant Λ, we have $\|\mathcal{L}^t_\nu\|_\infty \leq \Lambda$, i.e., $|T^{s,t}_{ij}| \leq \Lambda$ for all $i, j \in \{1, \ldots, k\}$. This implies $\|T^{s,t}\|_2^2 \leq k^2\Lambda^2$.

Theorem 5 (Matrix error) *Let $V = (V^t)_{t\in T_\mathcal{I}}$ and $W = (W^s)_{s\in T_\mathcal{I}}$ be cluster bases and let $\tilde{V} = (\tilde{V}^t)_{t\in T_\mathcal{I}}$ and $\tilde{W} = (\tilde{W}^s)_{s\in T_\mathcal{I}}$ be orthogonal cluster bases satisfying*

$$\|V^t - \tilde{V}^t(\tilde{V}^t)^\top V^t\|_F^2 \leq \epsilon_t \quad \text{and} \quad \|W^s - \tilde{W}^s(\tilde{W}^s)^\top W^s\|_F^2 \leq \epsilon_s$$

for all $t, s \in T_\mathcal{I}$. For each $(t, s) \in P_{\text{far}}$, we define

$$\tilde{S}^{t,s} := (\tilde{V}^t)^\top V^t S^{t,s} (W^s)^\top \tilde{W}^s.$$

Then we get the following error bound:

$$\|V^t S^{t,s}(W^s)^\top - \tilde{V}^t \tilde{S}^{t,s}(\tilde{W}^s)^\top\|_F^2 \leq \epsilon_t \|S^{t,s}(W^s)^\top\|_2^2 + \epsilon_s \|V^t S^{t,s}\|_2^2.$$

Proof. By Pythagoras' equality, we have

$$\begin{aligned}
&\|V^t S^{t,s}(W^s)^\top - \tilde{V}^t \tilde{S}^{t,s}(\tilde{W}^s)^\top\|_F^2 \\
&= \|V^t S^{t,s}(W^s)^\top - \tilde{V}^t((\tilde{V}^t)^\top V^t S^{t,s}(W^s)^\top \tilde{W}^s)(\tilde{W}^s)^\top\|_F^2 \\
&= \|V^t S^{t,s}(W^s)^\top - \tilde{V}^t(\tilde{V}^t)^\top V^t S^{t,s}(W^s)^\top\|_F^2 \\
&\quad + \|\tilde{V}^t(\tilde{V}^t)^\top V^t S^{t,s}(W^s)^\top - \tilde{V}^t(\tilde{V}^t)^\top V^t S^{t,s}(W^s)^\top \tilde{W}^s(\tilde{W}^s)^\top\|_F^2 \\
&\leq \|V^t - \tilde{V}^t(\tilde{V}^t)^\top V^t\|_F^2 \|S^{t,s}(W^s)^\top\|_2^2 \\
&\quad + \|\tilde{V}^t(\tilde{V}^t)^\top V^t S^{t,s}\|_2^2 \|(W^s)^\top - (W^s)^\top \tilde{W}^s(\tilde{W}^s)^\top\|_F^2 \\
&\leq \epsilon_t \|S^{t,s}(W^s)^\top\|_2^2 + \|V^t S^{t,s}\|_2^2 \epsilon_s.
\end{aligned}$$

∎

This general error estimate implies that we need bounds for $\|S^{t,s}(W^s)^\top\|_2^2$ and $\|V^t S^{t,s}\|_2^2$ in order to find error bounds for the approximated matrix. We will consider only the second case, since the first follows directly due to symmetry. For each of the matrix entries of $V^t S^{t,s}$ with $(t, s) \in P_{\text{far}}$, we get

$$\begin{aligned}
|(V^t S^{t,s})_{i\mu}| &= \left|\sum_{\nu=1}^k V^t_{i\nu} S^{t,s}_{\nu\mu}\right| = \left|\sum_{\nu=1}^k \int_\Omega \varphi_i(x)\mathcal{L}^t_\nu(x)\,dx\ \kappa(x^t_\nu, x^s_\mu)\right| \\
&= \left|\int_\Omega \varphi_i(x)\mathcal{I}^t[\kappa(\cdot, x^s_\mu)](x)\,dx\right| \leq \|\varphi_i\|_{L^1} \|\mathcal{I}^t[\kappa(\cdot, x^s_\mu)]\|_{L^\infty(B^t)}
\end{aligned}$$

$$\leq \Lambda \|\varphi_i\|_{L^1} \|\kappa(\cdot, x_\mu^s)\|_{L^\infty(B^t)}.$$

Since κ is asymptotically smooth (cf. (2)) and since (t,s) is admissible, we find

$$\|\kappa(\cdot, x_\mu^s)\|_{L^\infty(B^t)} \lesssim \operatorname{dist}(B^t, B^s)^{-g} \lesssim \operatorname{diam}(B^t)^{-g}.$$

For quasi-regular grids with grid parameter h, we can expect $h \lesssim \operatorname{diam}(B^t)$ and $\|\varphi_i\|_{L^1} \sim h^{d_\Omega}$, where d_Ω is the dimension of the sub-manifold Ω, so combining the estimates gives us

$$|(V^t S^{t,s})_{i\mu}| \lesssim \Lambda h^{d_\Omega - g}$$

and therefore

$$\|V^t S^{t,s}\|_2^2 \lesssim \Lambda^2 h^{2(d_\Omega - g)} k \# \hat{t}.$$

Lemma 6 (Complexity) *For a general cluster tree with c clusters, the orthogonalization algorithm requires $\mathcal{O}(ck^3) \subseteq \mathcal{O}(nk^3)$ operations.*

Proof. Using the LQ decomposition, we can perform all operations for a leaf cluster in $\mathcal{O}(m^2(k+m))$ and for a non-leaf cluster in $\mathcal{O}\left(k^2 \sum_{t' \in \operatorname{sons}(t)} k^{t'} + m^2(k+m)\right)$ operations.

Since $m \leq k$ and $k^t \leq k$ hold, we have

$$\sum_{t \in T_{\mathcal{I}}} \left(k^2 \sum_{t' \in \operatorname{sons}(t)} k^{t'} + m^2(k+m) \right) \leq \sum_{t' \in T_{\mathcal{I}}} k^2 k^t + \sum_{t \in T_{\mathcal{I}}} 2k^3 \leq 3ck^3$$

and get the desired bound for the total complexity. ∎

Remark 7 *For balanced cluster trees, the above complexity estimate can be improved. In order to keep the argument simple, we assume that $n = 2^p$ holds and that $T_{\mathcal{I}}$ is a balanced binary tree. This implies that $\#\hat{t} = 2^{p - \operatorname{level}(t)}$ holds for all $t \in T_{\mathcal{I}}$. Then the improved bound of $\mathcal{O}(nk^2(\log_2 k + 1))$ for the complexity can be derived as follows: We split the set of clusters into*

$$T_{\text{large}} := \{t \in T_{\mathcal{I}} \ : \ \#\hat{t} \geq k\} \quad \text{and} \quad T_{\text{small}} := \{t \in T_{\mathcal{I}} \ : \ \#\hat{t} < k\}$$

and define

$$\ell^* := \min\{\ell \ : \ 2^{p-\ell} < k\}.$$

For all $t \in T_{\text{large}}$, we have $2^{p - \operatorname{level}(t)} = \#\hat{t} \geq k$, i.e., $\operatorname{level}(t) < \ell^*$. The minimality of ℓ^* implies

$$\#T_{\text{large}} \leq 2^{\ell^*} = 2\ 2^{\ell^* - 1} = 2\frac{2^p}{2^{p-(\ell^*-1)}} \leq 2\frac{2^p}{k} = 2n/k.$$

The orthogonalization requires not more than $\mathcal{O}(k^3)$ operations for each cluster $t \in T_{\text{large}}$, so summing over all of these clusters gives us a complexity of $\mathcal{O}(nk^2)$.

Now let us consider the clusters in T_{small}. For each t in this set, we have $2^{p - \operatorname{level}(t)} = \#\hat{t} < k$, i.e., $\operatorname{level}(t) \geq \ell^*$. By definition, we have $m = \min\{k, \#\hat{t}\}$, so we need not

more than $\mathcal{O}(m^2(k+m)) \subseteq \mathcal{O}(k^2\#\hat{t})$ *operations for leaf clusters and not more than* $\mathcal{O}(m^2(k+m)+(k^{t_1}+k^{t_2})k^2) \subseteq \mathcal{O}(k^2\#\hat{t})$ *operations for non-leaf clusters. Due to*

$$\begin{aligned}\sum_{t\in T_{\mathrm{small}}} k^2\#\hat{t} &\leq k^2 \sum_{\ell=\ell^*}^{p} \sum_{\{t,\mathrm{level}(t)=\ell\}} \#\hat{t} = k^2 \sum_{\ell=\ell^*}^{p} n \\ &= k^2 n(p-\ell^*+1) \leq k^2 n(\log_2(k)+1),\end{aligned}$$

the total complexity for all clusters in T_{small} *is in* $\mathcal{O}(nk^2(\log_2(k)+1))$*. Combining the estimates for* T_{large} *and* T_{small}*, we get the desired bound for the complexity of the entire algorithm.*

Remark 8 *In [18], a similar approach is used in the context of multipole expansions and wavelet bases: The matrix* V^t *is created by discretizing harmonic polynomials and then orthogonalized by using a singular value decomposition.*

4 Recompression

The orthogonalization process uses only the cluster basis and transfer matrices. These matrices depend only on the discretization and the geometry, but not on the kernel function. This means that we cannot expect the process to reach the efficiency of multipole methods, since these use expansions that are designed for a specific type of kernel function.

If we aim for optimal cluster bases, we therefore have to take the kernel function into account. After polynomial approximation, all the information on the kernel function is contained in the coefficient matrices $S^{t,s}$ (cf. (8)), and we can use this information to construct better cluster bases.

Our approach is based on the algorithm presented in [2]. By modifying the computation of the Gram matrices used in the eigenvalue problems, we can derive a variant that requires only $\mathcal{O}(ck^3)$ operations.

4.1 Efficient computation of Gram matrices

The fundamental step in the adaptive algorithm is the efficient computation of Gram matrices describing the interaction of the basis vectors.

We apply this algorithm to an $\mathcal{H}^2$-matrix approximation $\tilde{K}$ that can be constructed by interpolation or similar techniques.

For the adaptive algorithm, we will need restrictions of matrices to subsets. We introduce the notation

$$A|^0_{\hat{t}\times\hat{s}} := \begin{cases} A_{ij} & \text{if } i\in\hat{t}, j\in\hat{s} \\ 0 & \text{otherwise} \end{cases}$$

for matrices $A\in\mathbb{R}^{\mathcal{I}\times\mathcal{I}}$ and clusters $t,s\in T_{\mathcal{I}}$. The basis construction algorithm from [2] requires the cluster Gram matrix

$$G^t := \sum_{s\in P^{t+}_{\mathrm{far}}} \tilde{K}|^0_{\hat{t}\times\hat{s}} (\tilde{K}|^0_{\hat{t}\times\hat{s}})^\top. \tag{22}$$

for

$$P_{\mathrm{far}}^{t+} := \{s \in T_{\mathcal{I}} \ : \ \text{there exists } t^+ \in T_{\mathcal{I}} \text{ with } \hat{t} \subseteq \hat{t}^+ \text{ and} (t^+, s) \in P_{\mathrm{far}}\}$$

(compare [2, eq. (5.4)]). The basic idea for building G^t efficiently is to split P_{far}^{t+} into tree levels and use the recursive representation

$$G^t = \begin{cases} \sum_{s \in P_{\mathrm{far}}^t} \tilde{K}|_{\hat{t} \times \hat{s}}^0 (\tilde{K}|_{\hat{t} \times \hat{s}}^0)^\top + G^{t^*}|_{\hat{t} \times \hat{t}}^0 & \text{if } t \text{ has a father } t^* \in T_{\mathcal{I}} \\ \sum_{s \in P_{\mathrm{far}}^t} \tilde{K}|_{\hat{t} \times \hat{s}}^0 (\tilde{K}|_{\hat{t} \times \hat{s}}^0)^\top & \text{otherwise,} \end{cases}$$

with

$$P_{\mathrm{far}}^t := \{s \in T_{\mathcal{I}} \ : \ (t, s) \in P_{\mathrm{far}}\}.$$

Using the special structure (6) of the terms in the sum, we get

$$\sum_{s \in T_{\mathcal{I}}, (t,s) \in P_{\mathrm{far}}} \tilde{K}|_{\hat{t} \times \hat{s}}^0 (\tilde{K}|_{\hat{t} \times \hat{s}}^0)^\top = V^t \left(\sum_{s \in P_{\mathrm{far}}^t} S^{t,s} (W^s)^\top W^s (S^{t,s})^\top \right) (V^t)^\top.$$

Since the cluster basis V is nested, we have

$$(V^t X (V^t)^\top)|_{\hat{t}' \times \hat{t}'}^0 = V^{t'} T^{t'} X (T^{t'})^\top (V^{t'})^\top,$$

for any matrix $X \in \mathbb{R}^{k \times k}$, so a simple induction shows that

$$G^t = V^t C^t (V^t)^\top \tag{23}$$

holds for all $t \in T_{\mathcal{I}}$, where C^t is given by

$$C^t := \begin{cases} \sum_{s \in P_{\mathrm{far}}^t} S^{t,s} (W^s)^\top W^s (S^{t,s})^\top + T^t C^{t^*} (T^t)^\top & \text{if } t \text{ has a father } t^* \in T_{\mathcal{I}} \\ \sum_{s \in P_{\mathrm{far}}^t} S^{t,s} (W^s)^\top W^s (S^{t,s})^\top & \text{otherwise.} \end{cases} \tag{24}$$

We can use the nested structure of the cluster basis W in order to prepare the auxiliary matrices $Y^s := (W^s)^\top W^s$ for all $s \in T_{\mathcal{I}}$ by a recursive procedure in $\mathcal{O}(ck^3)$. Using Y^s, the matrices C^t for all $t \in T_{\mathcal{I}}$ can be computed in $\mathcal{O}(ck^3)$ operations.

4.2 Construction of adaptive bases

According to [2], the optimal cluster basis matrix $\tilde{V}^t \in \mathbb{R}^{\hat{t} \times k^t}$ for a leaf cluster $t \in T_{\mathcal{I}}$ is orthogonal and maximizes the quantity

$$\sum_{s \in P_{\mathrm{far}}^{t+}} \|(\tilde{V}^t)^\top \tilde{K}|_{\hat{t} \times \hat{s}}^0\|_F^2,$$

and the solution of this maximization problem can be constructed by using an orthogonal basis of the eigenvectors corresponding to the k^t largest eigenvalues of the matrix G^t defined in (22).

Due to (23), G^t can be computed in $\mathcal{O}((k + \#\hat{t})k\hat{t})$ operations, and the eigenvalue problem can be solved in $\mathcal{O}(\#\hat{t}^3)$ operations. Equation (23) implies that the rank of G^t cannot exceed $m := \min\{k, \#\hat{t}\}$.

If $t \in T_{\mathcal{I}}$ is not a leaf cluster, we again have to ensure that the new cluster basis is nested. As in the case of orthogonalization, we do this by projecting V^t into the space spanned by the cluster bases of the sons of t, i.e., by replacing V^t by its orthogonal projection $\widehat{V}^t$ defined in (12). This leads to a modified Gram matrix

$$\widehat{G}^t := \widehat{V}^t C^t (\widehat{V}^t)^\top.$$

This matrix differs in one important point from the one used in the orthogonalization procedure: Since $\widehat{V}^t$ appears "outside" of the product, we would have to solve an eigenvalue problem of dimension $\#\hat{t}$. This would be too expensive for large clusters, so we have to find a way of reducing the dimension.

For $\operatorname{sons}(t) = \{t_1, \ldots, t_s\}$, we can split the projected matrix $\widehat{V}^t$ into an orthogonal rectangular matrix and a remainder with $q := k^{t_1} + \ldots + k^{t_s}$ rows:

$$\widehat{V}^t = \sum_{t' \in \operatorname{sons}(t)} \tilde{V}^{t'} P^{t'} T^{t'} = \begin{pmatrix} \tilde{V}^{t_1} & \ldots & \tilde{V}^{t_s} \end{pmatrix} \begin{pmatrix} P^{t_1} T^{t_1} \\ \vdots \\ P^{t_s} T^{t_s} \end{pmatrix}.$$

If we set

$$Q^t := \begin{pmatrix} \tilde{V}^{t_1} & \ldots & \tilde{V}^{t_s} \end{pmatrix}$$

and recall the definition of X^t in (16), we can rewrite $\widehat{G}^t$ in the form

$$\widehat{G}^t = Q^t X^t C^t (X^t)^\top (Q^t)^\top.$$

Since the matrix Q^t is orthogonal, the non-zero eigenvalues of the matrices $X^t C^t (X^t)^\top$ and $\widehat{G}^t$ are identical, and the eigenvectors of the latter can be computed from those of the former by applying Q^t. This means that we can solve the eigenvalue problem by only $\mathcal{O}(q^3)$ operations instead of the $\mathcal{O}((\#\hat{t})^3)$ operations of the direct approach.

We construct the orthogonal matrix $Z^t \in \mathbb{R}^{q \times k^t}$ from an orthogonal eigenvector basis corresponding to the k^t largest eigenvalues of $X^t C^t (X^t)^\top$. Then $\tilde{V}^t = Q^t Z^t$ contains the eigenvectors of $\widehat{G}^t$ corresponding to its k^t largest eigenvalues. Since t is not a leaf cluster, we have to compute the transfer matrices $\tilde{T}^{t'}$ satisfying

$$\tilde{V}^t = \sum_{t' \in \operatorname{sons}(t)} \tilde{V}^{t'} \tilde{T}^{t'} = \begin{pmatrix} \tilde{V}^{t_1} & \ldots & \tilde{V}^{t_s} \end{pmatrix} \begin{pmatrix} \tilde{T}^{t_1} \\ \vdots \\ \tilde{T}^{t_s} \end{pmatrix} = Q^t \begin{pmatrix} \tilde{T}^{t_1} \\ \vdots \\ \tilde{T}^{t_s} \end{pmatrix}.$$

Due to the definition of $\tilde{V}^t$ and the orthogonality of Q^t, this implies

$$\begin{pmatrix} \tilde{T}^{t_1} \\ \vdots \\ \tilde{T}^{t_s} \end{pmatrix} = (Q^t)^\top \tilde{V}^t = (Q^t)^\top Q^t Z^t = Z^t,$$

and we can extract the transfer matrices directly from Z^t.

4.3 Complete algorithm

We have seen that we can compute adaptive row cluster bases efficiently if the matrices C^t have been prepared in advance. In order to do this efficiently, we need the additional auxiliary matrices $Y^s := (W^s)^\top W^s$.

Since W is nested, we have

$$Y^s = \begin{cases} \sum_{s' \in \operatorname{sons}(t)} (T^{s'})^\top Y^{s'} T^{s'} & \text{if } \operatorname{sons}(s) \neq \emptyset \\ (W^s)^\top W^s & \text{otherwise,} \end{cases}$$

so all matrices Y^s can be computed by a recursive procedure in $\mathcal{O}(ck^3)$ operations.

Using the recursive definition (24), we can compute all matrices C^t in $\mathcal{O}(ck^3)$ operations.

Lemma 9 (Complexity) *Let $C_{\mathrm{leaf}}, C_{\mathrm{sp}}, C_{\mathrm{sons}} \in \mathbb{N}$ such that $\#\hat{t} \leq C_{\mathrm{leaf}}$ holds for all leaf clusters $t \in T_{\mathcal{I}}$, that $\operatorname{sons}(t) \leq C_{\mathrm{sons}}$ holds for all non-leaf clusters $t \in T_{\mathcal{I}}$, and that*

$$\{s \ : \ (t,s) \in P_{\mathrm{far}}\} \leq C_{\mathrm{sp}} \quad \text{and} \quad \{s \ : \ (s,t) \in P_{\mathrm{far}}\} \leq C_{\mathrm{sp}} \tag{25}$$

holds for all clusters $t \in T_{\mathcal{I}}$. Then the adaptive construction of a row or a column cluster basis requires $\mathcal{O}(ck^3)$ operations.

Proof. We start be noticing that column cluster bases can be computed by applying our algorithm to $\tilde{K}^\top$ instead of $\tilde{K}$, so we only have to consider the case of row cluster bases.

Due to (25), the construction of C^t can be accomplished in $\mathcal{O}(k^3)$ operations by using the equation (24). Computing all matrices C^t recursively requires $\mathcal{O}(ck^3)$ operations.

Forming the matrices X^t requires $\mathcal{O}(k^2 k^t)$ operations for each cluster, leading to a total of $\mathcal{O}(ck^3)$ operations.

Using C_{sons} and C_{leaf}, we can show that each eigenvalue problem can be solved in $\mathcal{O}(k^3)$ operations, giving us again a total of $\mathcal{O}(ck^3)$.

Using the recursive equation (15), the matrices P^t can also be computed in $\mathcal{O}(ck^3)$ steps, which concludes the proof. ∎

In our case, the error analysis of [2] takes the following form:

Theorem 10 (Truncation error) *For leaf or non-leaf clusters $t \in T_{\mathcal{I}}$, denote the eigenvalues of G^t or $\widehat{G}^t$, respectively, by $\lambda_1^t \geq \ldots \geq \lambda_m^t$ and set*

$$\epsilon_F^t := \sum_{i=k^t+1}^{m} \lambda_i^t.$$

The matrix $\widehat{K} \in \mathbb{R}^{\mathcal{I} \times \mathcal{I}}$ defined by

$$\widehat{K}|_{\hat{t} \times \hat{s}}^0 := \begin{cases} \tilde{V}^t \tilde{S}^{t,s} (W^s)^\top & \text{if } (t,s) \in P_{\mathrm{far}} \\ K|_{\hat{t} \times \hat{s}}^0 & \text{otherwise} \end{cases} \quad \text{with} \quad \tilde{S}^{t,s} := (\tilde{V}^t)^\top V^t S^{t,s} = P^t S^{t,s}$$

for $(t,s) \in P$ satisfies

$$\|\tilde{K} - \widehat{K}\|_F^2 \leq \sum_{t \in T_{\mathcal{I}}} \epsilon_F^t.$$

Proof. See [2, Remark 5.1, Lemma 7.1]. ∎

Of course, a similar estimate holds for the column cluster bases, so that the approximation error for orthogonalized row and column bases can be bounded by [2, Lemma 5.2].

Remark 11 (Adaptivity) *We can control the approximation error by choosing k^t appropriately. This has to be done carefully, since we have to balance three sources of errors: The discretization error of the Galerkin method, the error introduced by interpolating the kernel function, and the approximation error of the algebraic compression.*

If the discretization error is large, choosing a high-order interpolation scheme is not a good idea. And if the interpolation order is low, using a high rank in the recompression algorithm will only approximate the "numerical garbage" introduced by the inaccurate interpolation.

Remark 12 (Variable rank) *For special kernel functions, it is possible to choose the order of the interpolation for each box B^t depending on the size of B^t. If the size of the box and the order are carefully balanced, the $\mathcal{H}^2$-matrix constructed by interpolation will require only $\mathcal{O}(n)$ units of storage and matrix-vector multiplication can be accomplished in $\mathcal{O}(n)$ operations ([16, 17, 5]).*

We can apply both the orthogonalization and the recompression algorithm presented here to these modified approximations, and a careful analysis shows that they will also require only $\mathcal{O}(n)$ operations.

5 Numerical experiments

We test the algorithms by applying them to the trace of the classical double layer potential operator

$$\mathcal{K}_{\mathrm{DLP}}[u](x) := \int_\Gamma u(y) \frac{\partial}{\partial n(y)} \frac{1}{4\pi\|x-y\|}\,dy$$

for $x \in \Gamma$, discretized by piecewise constant or linear basis functions on a quasi-regular triangulation of the unit sphere $\Gamma_S := \{\|x\|_2 = 1\}$ or the unit cube $\Gamma_C := \{\|x\|_\infty = 1\}$.

The matrices of the approximation are defined by

$$V^t_{i\nu} := \int_\Gamma \varphi_i(x)\mathcal{L}^t_\nu(x)\,dx, \quad W^s_{j\mu} := \int_\Gamma \varphi_j(y)\frac{\partial}{\partial n(y)}\mathcal{L}^s_\mu(y)\,dy, \quad S^{t,s}_{\nu\mu} := \frac{1}{4\pi\|x^t_\nu - x^s_\mu\|},$$

where the Lagrange polynomials $\mathcal{L}^t_\nu$ correspond to tensor Chebyshev interpolation for the minimal axis-parallel box B^t containing the support of all piecewise constant basis functions φ_i with $i \in \hat{t}$.

We construct the cluster tree by recursive bisection stopping if a cluster contains not more than 32 degrees of freedom, and use the simple admissibility condition

$$\max\{\operatorname{diam}(B^t), \operatorname{diam}(B^s)\} \leq \eta \operatorname{dist}(B^t, B^s)$$

with $\eta = 2$ instead of (3) to create a block partition.

n	Build	Bld/n	Memory	Mem/n	MVM	Error
2048	8	4.0	34.0	17.0	0.06	6.1_{-4}
8192	35	4.3	168.6	21.1	0.31	6.0_{-4}
32768	150	4.6	703.1	22.0	1.34	6.8_{-4}
131072	773	5.9	2810.2	22.0	5.44	7.2_{-4}
524288	3262	6.2	11281.7	22.0	21.85	7.5_{-4}

Table 1: Orthogonalization of cluster bases on the sphere Γ_S with cubic interpolation

n	Build	Bld/n	Memory	Mem/n	MVM	Error
2048	11	5.4	7.5	3.7	0.01	5.9_{-4}
8192	46	5.6	34.9	4.3	0.11	6.5_{-4}
32768	186	5.7	147.4	4.5	0.47	6.9_{-4}
131072	760	5.8	607.9	4.6	2.15	7.0_{-4}
524288	3245	6.2	2685.9	5.2	9.46	7.4_{-4}

Table 2: Recompression of cluster bases on the sphere Γ_S with cubic interpolation and recompression tolerance $\epsilon = 10^{-3}$

In order to reduce the storage requirements, the original $\mathcal{H}^2$-matrix approximation constructed by interpolation is not stored, but created in temporary storage whenever the orthogonalization or recompression algorithms require it. All computations were performed by HLib [1] on UltraSPARC IIIcu processors running at 900 MHz.

In the first example, we apply the orthogonalization algorithm to an intermediate approximation constructed by cubic interpolation. The results of the experiment are reported in Table 1. The first and second column give the time for the construction of the approximation: The first column contains the total time in seconds, the second contains the time per degree of freedom in milliseconds. The third and fourth column give the amount of storage needed, again the total in MB and per degree of freedom in KB. The fifth column contains the time in seconds required for one matrix-vector multiplication, and the sixth column gives the relative approximation error in the operator norm. We can see that the relative error is bounded by 10^{-3} and that time and storage requirements grow linearly in the number of degrees of freedom.

The second example is the recompression algorithm. Comparing the results in Table 2 to those of the orthogonalization procedure, we see that the storage requirements and the time per matrix-vector multiplication are significantly reduced, while the time for building the approximation is only slightly increased.

The storage requirements can be reduced significantly by increasing the admissibility parameter η. In order to reach a sufficient accuracy despite of the resulting weaker admissibility condition, we would have to increase the interpolation order, and this implies a higher computational complexity.

There are applications that require an approximation error below the value 10^{-3} use in Table 2. In order to reach the range of 10^{-4}, we have to use quartic instead of cubic

n	Build	Bld/n	Memory	Mem/n	MVM	Error
2048	28	13.9	9.0	4.5	0.02	8.3_{-5}
8192	138	16.9	42.2	5.3	0.13	8.2_{-5}
32768	577	17.6	183.8	5.7	0.56	1.1_{-4}
131072	2450	18.7	781.3	6.1	2.50	1.2_{-4}
524288	9789	18.7	3296.5	6.4	10.68	1.2_{-4}

Table 3: Recompression of cluster bases on the sphere Γ_S with quartic interpolation and recompression tolerance $\epsilon = 10^{-4}$

n	Build	Bld/n	Memory	Mem/n	MVM	Error
3072	27	9.08	12.8	4.3	0.03	2.9_{-4}
12288	113	9.16	46.6	3.9	0.14	4.3_{-4}
49152	452	9.19	168.0	3.5	0.51	5.6_{-4}
196608	1791	9.11	609.4	3.2	1.91	6.4_{-4}
786432	7097	9.02	2245.3	2.9	6.96	7.0_{-4}

Table 4: Recompression of cluster bases on the cube Γ_C with quartic interpolation and recompression tolerance $\epsilon = 10^{-3}$

interpolation (i.e., five interpolation points per coordinate direction) and reduce the tolerance for the recompression algorithm to 10^{-4}. The results are listed in Table 3: The computation time is increased significantly, since each coefficient matrix now has $5^6 = 15625$ entries instead of $4^6 = 4096$, but the storage requirements grow only moderately. The relative approximation error is close to the desired value of 10^{-4}, increasing the interpolation order further would reduce it below this mark.

Experiments carried out on the unit sphere Γ_S do not provide us with information about the behaviour of our method for domains with edges. In order to verify that the recompression algorithm can cope with this type of domain, we consider the unit cube Γ_C. Table 4 shows that we can reach a relative error below 10^{-3} by using quartic interpolation. Compared to the unit sphere, the storage requirements are reduced, but the quartic interpolation leads to a 50% increase of the time required for building the approximation.

References

[1] S. Börm and L. Grasedyck, HLib – *a library for $\mathcal{H}$- and $\mathcal{H}^2$-matrices*, 1999. Available at `http://www.hlib.org/`.

[2] S. Börm and W. Hackbusch, *Data-sparse approximation by adaptive $\mathcal{H}^2$-matrices*, Computing, 69 (2002), pp. 1–35.

[3] ——, *$\mathcal{H}^2$-matrix approximation of integral operators by interpolation*, Applied Numerical Mathematics, 43 (2002), pp. 129–143.

[4] ——, *Approximation of boundary element operators by adaptive $\mathcal{H}^2$-matrices*, Foundations of Computational Mathematics, 312 (2004), pp. 58–75.

[5] S. Börm, M. Löhndorf, and J. M. Melenk, *Approximation of integral operators by variable-order interpolation*, Tech. Rep. 82, Max Planck Institute for Mathematics in the Sciences, 2002. To appear in Numerische Mathematik.

[6] S. Börm and S. A. Sauter, *BEM with linear complexity for the classical boundary integral operators*, Tech. Rep. 15, Institute of Mathematics, University of Zurich, 2003. To appear in Mathematics of Computation.

[7] A. Brandt and A. A. Lubrecht, *Multilevel matrix multiplication and fast solution of integral equations*, J. Comput. Phys., 90 (1990), pp. 348–370.

[8] W. Dahmen and R. Schneider, *Wavelets on manifolds I: Construction and domain decomposition*, SIAM Journal of Mathematical Analysis, 31 (1999), pp. 184–230.

[9] K. Giebermann, *Multilevel approximation of boundary integral operators*, Computing, 67 (2001), pp. 183–207.

[10] L. Grasedyck and W. Hackbusch, *Construction and arithmetics of $\mathcal{H}$-matrices*, Computing, 70 (2003), pp. 295–334.

[11] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, Journal of Computational Physics, 73 (1987), pp. 325–348.

[12] L. Greengard and V. Rokhlin, *A new version of the fast multipole method for the Laplace in three dimensions*, in Acta Numerica 1997, Cambridge University Press, 1997, pp. 229–269.

[13] W. Hackbusch, B. Khoromskij, and S. Sauter, *On $\mathcal{H}^2$-matrices*, in Lectures on Applied Mathematics, H. Bungartz, R. Hoppe, and C. Zenger, eds., Springer-Verlag, Berlin, 2000, pp. 9–29.

[14] W. Hackbusch and Z. P. Nowak, *On the fast matrix multiplication in the boundary element method by panel clustering*, Numerische Mathematik, 54 (1989), pp. 463–491.

[15] V. Rokhlin, *Rapid solution of integral equations of classical potential theory*, Journal of Computational Physics, 60 (1985), pp. 187–207.

[16] S. Sauter, *Variable order panel clustering (extended version)*, Tech. Rep. 52, Max-Planck-Institut für Mathematik, Leipzig, Germany, 1999.

[17] ——, *Variable order panel clustering*, Computing, 64 (2000), pp. 223–261.

[18] J. Tausch and J. White, *Multiscale bases for the sparse representation of boundary integral operators on complex geometries*, SIAM J. Sci. Comput., 24 (2003), pp. 1610–1629.

Steffen Börm
Max-Planck-Institut für Mathematik in den Naturwissenschaften
Inselstrasse 22–26
04103 Leipzig
Germany
sbo@mis.mpg.de