Universität des Saarlandes

Fachbereich 9 – Mathematik

Mathematischer Preprint

Low-rank Approximation of Boundary Element Matrices

Mario Bebendorf

Preprint No. 1

Saarbrücken 1999

Universität des Saarlandes

Fachbereich 9 – Mathematik

Low-rank Approximation of Boundary Element Matrices

Mario Bebendorf

Saarland University
Department of Mathematics
Postfach 15 11 50
D–66041 Saarbrücken
Germany
E-Mail: bebendorf@num.uni-sb.de

submitted: June 16, 1999

Preprint No. 1

Saarbrücken 1999

Edited by
FB 9 – Mathematik
Im Stadtwald
D–66041 Saarbrücken
Germany

Fax: + 49 681 302 4443
e-mail: preprint@math.uni-sb.de
WWW: http://www.math.uni-sb.de/

Abstract

In this article the problem of approximating a general asymptotically smooth function in two variables, typically arising in integral formulations of boundary value problems, by a sum of products of two functions in one variable is considered. From these results an iterative algorithm for the low-rank approximation of blocks of large unstructured matrices generated by asymptotically smooth functions is developed. This algorithm uses only few entries from the original block and since it has a natural stopping criterion the approximative rank is not needed in advance.

AMS Subject Classification: 41A63, 41A80, 65D05, 65D15, 65F05, 65F30
Key words: multivariate interpolation, asymptotical smoothness, low-rank approximation, matrix partitioning

Contents

1 Introduction

The approximative application $\mathcal{A}f$ of an operator $\mathcal{A}$ coming from integral formulations on a function f has been investigated in many publications. Typically, the kernel function of the integral operator $\mathcal{A}$ is approximated by a degenerate kernel, i.e. a finite sum of separable functions (functional skeletons). In the case of multipole methods [17], [9], [10], [13], [14] these functions have to be known explicitly for each kernel. In contrast, algebraic methods approximate blocks of the discrete operator by low-rank matrices. Both approaches are designed to save operations and memory and both have basically the same idea behind them which can easily be explained in the case of low-rank approximants. Assume we have a matrix $T \in \mathbb{C}^{m\times n}$ of a small rank r. Because of the representation of low-rank matrices

$$T = \sum_{i=1}^{r} u_i v_i^*, \quad u_i \in \mathbb{C}^m,\ v_i \in \mathbb{C}^n$$

only $r(n+m)$ units of memory are needed to store them and a matrix-vector multiplication Tx (which is the basis for iterative solution techniques) takes $O(r(n+m))$ operations:

$$s_i = v_i^* x, \quad i = 1, \ldots, r$$

$$Tx = \sum_{i=1}^{r} s_i u_i.$$

Blocks of matrices arising in integral equations are usually dense and unstructured. Though possibly having full rank they may be well approximated by low-rank matrices. Thus for the approximation not the usual but the ε-rank is important.

Definition 1 (ε-rank). *The ε-rank of a matrix $A \in \mathbb{C}^{m\times n}$ with respect to the matrix norm $\|\cdot\|$ is defined as*

$$\operatorname{rank}_\varepsilon A = \min\{\operatorname{rank} T : \|A - T\| < \varepsilon\}.$$

The problem of finding the best approximant of a prescribed rank was solved by L. Mirsky [15] (see also [3]).

Theorem 1. *Let $A \in \mathbb{C}^{m\times n}$, $m \geq n$ and $\|\cdot\|$ be a unitarily invariant matrix norm. The best approximation of at most rank k to A is*

$$\|A - A_k\| = \min\{\|A - T\| : \text{rank } T \leq k\},$$

where $A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^*$ and (σ_i, u_i, v_i), $i = 1, \ldots, k$ are the k largest singular triplets. Especially

$$\min\{\|A - T\|_F^2 : \text{rank } T \leq k\} = \sum_{i=k+1}^{n} \sigma_i^2,$$

$$\min\{\|A - T\|_2 : \text{rank } T \leq k\} = \sigma_{k+1}$$

where $\|\cdot\|_F$, $\|\cdot\|_2$ denote the Frobenius and the spectral norm respectively.

Thus the optimal approximant to a prescribed accuracy is easy to find if the singular value decomposition (SVD) is accessible. But the SVD is computationally very expensive. Furthermore every entry of the original matrix A has to be calculated first. This is also the case if we apply the less expensive partial SVD, cf. [7]. Consequently, approximations based on the SVD cannot lead to fast algorithms. Our aim, however, is to find an approximant using less computational effort and especially only few entries from the original matrix.

Let $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_n\}$ be two sets of pairwise distinct points in $\mathbb{R}^d$ and D_X, D_Y the convex hulls of X and Y respectively.
If we use a quadrature formula to approximate the integrals, the approximant's properties with respect to availability of low-rank approximants come from the kernel. Thus we concentrate on the investigation of matrices

$$A \in \mathbb{R}^{m \times n}, \quad a_{ij} = f(x_i, y_j), \quad 1 \leq i \leq m, \ 1 \leq j \leq n$$

generated by functions $f : D_X \times D_Y \to \mathbb{R}$ of the type

$$f(x, y) = \sum_{k=0}^{N_p} g_k(x)\, h_k(y) + R_p(x, y) \tag{1}$$

where $|R_p(x, y)| \leq \varepsilon_p$ and $\varepsilon_p \to 0$ for $p \to \infty$.
It is evident that these matrices can be approximated by a matrix of rank N_p to an accuracy of order ε_p. In terms of ε-ranks this reads

$$\text{rank}_{\varepsilon_p} A = O(N_p).$$

Multipole methods use these decompositions directly. For this, however, the functions g_k and h_k have to be known for each function f. The method described in this article will only use the information about the existence of these decompositions. However, we confine ourselves to matrices generated by asymptotically smooth functions, cf. [1].

Definition 2. *A function $f : D_X \times D_Y \to \mathbb{R}$ is called asymptotically smooth if there are constants c_1, $c_2 > 0$ and $g \leq 0$ so that for any multi-index $\alpha \in \mathbb{N}_0^d$*

$$|\partial_y^\alpha f(x,y)| \leq c_1\, p!\, c_2^p\, |x-y|^{g-p}, \quad p = |\alpha|.$$

This class of functions is a subset of type (1) functions if we impose the condition

$$\operatorname{diam} D_Y \leq \eta \operatorname{dist}(D_X, D_Y), \quad 0 < \eta < (c_2 d)^{-1} \tag{2}$$

since by using the Taylor expansion at the point $y_0 \in D_Y$ we have

$$f(x,y) = \sum_{l=0}^{p-1} \frac{1}{l!}\left((y-y_0)\partial_y\right)^l f(x,y_0) + \frac{1}{p!}\left((y-y_0)\partial_y\right)^p f(x,\widetilde{y})$$

for some point $\widetilde{y} \in D_Y$.
Setting

$$R_p(x,y) = \frac{1}{p!}\left((y-y_0)\partial_y\right)^p f(x,\widetilde{y})$$

we observe that for $x \in D_X$, $y \in D_Y$

$$\begin{aligned} |R_p(x,y)| &\leq \frac{1}{p!} d^p \left(\frac{|y-y_0|}{|x-\widetilde{y}|}\right)^p c_1\, p!\, c_2^p\, |x-\widetilde{y}|^g \\ &\leq c_1 \operatorname{dist}^g(D_X, D_Y)\, (c_2 d \eta)^p =: \varepsilon_p. \end{aligned}$$

With respect to y the sum $\sum_{l=0}^{p-1} \frac{1}{l!}\left((y-y_0)\partial_y\right)^l f(x,y_0)$ is in Π_{p-1}^d the space of polynomials of the degree at most $p-1$. The number of linearly independent monomials y^α, $|\alpha| = l$, is $r_l^d = \binom{l+d-1}{l}$. Thus the ε_p-rank of the matrix A is bounded by the dimension n_p of Π_{p-1}^d.

$$\operatorname{rank}_{\varepsilon_p} A \leq n_p = \sum_{l=0}^{p-1} r_l^d \leq c_d\, p^d.$$

Remark : The geometrical parameter η arising in equation (2) will be used to control the approximation error. The sets X and Y will be generated from the original set of points by recursive subdivision so that condition (2) holds. This will be explained in subsection 3.1.

Instead of approximating a function in two variables by a sum of functional skeletons using some functions g_k and h_k explicitly given for each function as multipole methods do, we suggest using the values of the function itself to generate an approximation of type (1). For convenience let

$$f(x,[y]_k) = \begin{bmatrix} f(x,y_{j_1}) \\ \vdots \\ f(x,y_{j_k}) \end{bmatrix} \in \mathbb{R}^k, \quad f([x]_k,y) = \begin{bmatrix} f(x_{i_1},y) \\ \vdots \\ f(x_{i_k},y) \end{bmatrix} \in \mathbb{R}^k.$$

Then we are looking for decompositions

$$f(x,y) = f(x,[y]_k)^T G f([x]_k,y) + R_k(x,y) \tag{3}$$

with $G \in \mathbb{R}^{k\times k}$ and some $x_{i_l} \in X$ and $y_{j_l} \in Y$, $l = 1,\dots,k$.
In terms of matrices this means that we try to find a pseudo-skeleton component (cf. [8]), i.e. a matrix CGR where $C \in \mathbb{R}^{m\times k}$ are k columns and $R \in \mathbb{R}^{k\times n}$ k rows of the matrix A. The matrix $G \in \mathbb{R}^{k\times k}$ delivers some appropriate coefficients that are calculated from the submatrix of A in the intersection of the chosen rows R and columns C.
In [8] we can find a result on the existence of these pseudo-skeleton approximants.

Theorem 2. *Assume that for some $\varepsilon > 0$ the matrix $A \in \mathbb{R}^{m\times n}$ fulfils* $\operatorname{rank}_\varepsilon A \leq r$ *with respect to the spectral norm, then we can find a pseudo-skeleton component CGR, $G \in \mathbb{R}^{r\times r}$ such that*

$$\|A - CGR\|_2 \leq \varepsilon(1 + 2\sqrt{r}(\sqrt{n} + \sqrt{m})).$$

The proof uses a condition on the singular vectors of the low-rank approximant of A which is difficult to fulfil in practice. Furthermore, the approximation rank has to be known in advance to specify the number of columns and rows needed. In multipole methods the same problem occurs, whereby it is the number of functional skeletons which has to be known in advance. One way to solve this is to determine the number of functional skeletons needed from an upper bound of the remainder, but this certainly produces a number that is too high.
In the following we will present an iterative method to generate a decomposition of type (3). From this we will develop an iterative and incomplete algorithm for the low-rank approximation of matrices generated by asymptotically smooth functions without knowing the approximative rank in advance.

Let us first concentrate on the analytic problem of approximating a general asymptotically smooth function in two variables by a sum over functional skeletons.

2 Analytic problem

We construct the sequences $\{s_k\}$, $\{r_k\}$ by the following rule

$$r_0(x,y) = f(x,y), \quad s_0(x,y) = 0$$

and for $k = 0, 1, \ldots$

$$\begin{aligned} r_{k+1}(x,y) &= r_k(x,y) - \gamma_{k+1}\, r_k(x, y_{j_{k+1}})\, r_k(x_{i_{k+1}}, y) \\ s_{k+1}(x,y) &= s_k(x,y) + \gamma_{k+1}\, r_k(x, y_{j_{k+1}})\, r_k(x_{i_{k+1}}, y) \end{aligned}$$

where $\gamma_{k+1} = \big(r_k(x_{i_{k+1}}, y_{j_{k+1}})\big)^{-1}$ and $x_{i_{k+1}}$ and $y_{j_{k+1}}$ are chosen in every step so that $r_k(x_{i_{k+1}}, y_{j_{k+1}}) \neq 0$.

We realize that the functions r_k accumulate zeros. Thus s_k gradually interpolates f.

Lemma 1. *For $1 \leq l \leq k$ and all $x \in D_X$ we have $r_k(x, y_{j_l}) = 0$.*

Proof. The lemma holds for $l = k$ since

$$r_k(x, y_{j_k}) = r_{k-1}(x, y_{j_k}) - \gamma_k r_{k-1}(x, y_{j_k}) r_{k-1}(x_{i_k}, y_{j_k}) = 0.$$

We will prove the rest by induction from $k-1$ to k. We saw above that the lemma is true for $k = 1$. Assume it holds for $k-1$ then we have

$$r_k(x, y_{j_l}) = r_{k-1}(x, y_{j_l}) - \gamma_k r_{k-1}(x, y_{j_k}) r_{k-1}(x_{i_k}, y_{j_l}).$$

For all $x \in D_X$ and all $1 \leq l < k$ it holds that $r_{k-1}(x, y_{j_l}) = 0$ and the claim follows from this. □

The same statement holds if we interchange the roles of x and y.

We define the matrix $M_k^{(l)}(x)$ by

$$M_k^{(l)}(x) = \begin{bmatrix} f(x_{i_1}, y_{j_1}) & \ldots & f(x_{i_1}, y_{j_k}) \\ \vdots & & \vdots \\ f(x, y_{j_1}) & \ldots & f(x, y_{j_k}) \\ \vdots & & \vdots \\ f(x_{i_k}, y_{j_1}) & \ldots & f(x_{i_k}, y_{j_k}) \end{bmatrix},$$

where in the lth row we have the vector $f(x, [y]_k)$. Furthermore we set

$$M_k = M_k^{(l)}(x_{i_l}).$$

For the determinant of $M_k^{(l)}$ the following lemma can be shown.

Lemma 2. *For* $1 \leq l < k$

$$\det M_k^{(l)}(x) = r_{k-1}(x_{i_k}, y_{j_k}) \det M_{k-1}^{(l)}(x) - r_{k-1}(x, y_{j_k}) \det M_{k-1}^{(l)}(x_{i_k})$$

holds and

$$\begin{aligned} \det M_1^{(1)}(x) &= r_0(x, y_{j_1}), \\ \det M_k^{(k)}(x) &= r_{k-1}(x, y_{j_k}) \det M_{k-1}, \quad k > 1. \end{aligned}$$

Especially

$$\det M_k = r_0(x_{i_1}, y_{j_1}) \cdot \dots \cdot r_{k-1}(x_{i_k}, y_{j_k}).$$

Proof. It is easy to see that there are coefficients $\alpha_i^{(k-1)}$, $i = 1, \dots, k-1$, so that for all $x \in D_X$

$$r_{k-1}(x, y_{j_k}) = f(x, y_{j_k}) - \sum_{i=1}^{k-1} \alpha_i^{(k-1)} f(x, y_{j_i}).$$

Thus it is possible to replace each entry $f(\cdot, y_{j_k})$ in the last column of $M_k^{(l)}(x)$ by $r_{k-1}(\cdot, y_{j_k})$ and obtain $\widetilde{M}_k^{(l)}(x)$ without changing the determinant.
Since from the last lemma $r_{k-1}(x_{i_j}, y_{j_k}) = 0$, $1 \leq j \leq k-1$, only the lth and the kth entry of $\widetilde{M}_k^{(l)}(x)$ do not vanish. Using Laplace's theorem ends the proof. □

The last lemma guarantees that M_k is nonsingular and we are now able to show that the decomposition of f into s_k and r_k is of type (3).

Lemma 3. *For the generated sequences* s_k *and* r_k, $k \geq 0$

$$s_k(x, y) + r_k(x, y) = f(x, y)$$

holds, where for $k \geq 1$

$$s_k(x, y) = f(x, [y]_k)^T M_k^{-1} f([x]_k, y).$$

Proof. In the case $k = 1$ the lemma is obviously true. We proceed by induction from $k-1$ to k. From the definition of r_k and s_k we see that

$$s_k(x, y) + r_k(x, y) = s_{k-1}(x, y) + r_{k-1}(x, y)$$

which by the assumption is equal to $f(x, y)$.
It is easy to see that with s_{k-1} also s_k has the form

$$s_k(x, y) = f(x, [y]_k)^T G_k f([x]_k, y)$$

with some $G_k \in \mathbb{R}^{k \times k}$. Applying lemma 1 gives

$$f(x_{i_t}, y_{j_s}) = f(x_{i_t}, [y]_k)^T G_k f([x]_k, y_{j_s}) \iff M_k = M_k \, G_k \, M_k.$$

From this it finally follows that $G_k = M_k^{-1}$. □

In the following we will relate the remainder r_{n_p} of the approximation to the remainder of polynomial interpolation. It can already be seen from the existence result using the Taylor expansion that in multidimensional space more than one additional functional skeleton in the approximation is necessary to increase the order of approximation by one. This is why in our theoretical result, which is based on polynomial interpolation, we only expect an increase of the order of accuracy from the n_pth to the n_{p+1}th step. There may, however, be functions f for which the step size from one order of accuracy to the next is smaller.

In multidimensional space polynomial interpolation is generally not unique. The existence of interpolation polynomials is always guaranteed but the uniqueness depends on the configuration of the points. They must not lie on a hypersurface of degree p, or equivalently there is no polynomial in Π_p^d that vanishes on all of the points. However, the set of points for which Lagrange interpolation is not unique has measure zero. From the condition $r_k(x_{i_{k-1}}, y_{j_{k-1}}) \neq 0$ on the choice of the points we guarantee the uniqueness, which is important for the remainder of polynomial interpolation.

Lemma 4. *Assume that* $\det M_{n_p} \neq 0$ *then the Lagrange interpolation in the points* $y_1, \ldots, y_{n_p}$ *is unique.*

Proof. Assume we have a polynomial $P \in \Pi_{p-1}^d$ fulfilling

$$P(y_{j_l}) = 0, \quad l = 1, \ldots, n_p.$$

We have to show that P vanishes identically.
Define polynomials $p_k \in \Pi_{p-1}^d$ by

$$p_k(y_{j_l}) = f(x_{i_k}, y_{j_l}), \quad k, l = 1, \ldots, n_p.$$

Take an identically vanishing linear combination

$$\sum_{k=1}^{n_p} \alpha_k p_k(y) = 0 \quad \text{for all } y \in D_Y.$$

Especially

$$0 = \sum_{k=1}^{n_p} \alpha_k p_k(y_{j_l}) = \sum_{k=1}^{n_p} \alpha_k f(x_{i_k}, y_{j_l}) \quad \text{for } 1 \leq l \leq n_p.$$

But since M_{n_p} is nonsingular we have $\alpha_k = 0$ for $1 \leq k \leq n_p$. $\{p_k\}$ is a linearly independent system and thus a basis of Π_{p-1}^d.

Consequently we can find $\lambda_k \in \mathbb{R}$ so that $P = \sum_{k=1}^{n_p} \lambda_k p_k$. But from

$$0 = P(y_{j_l}) = \sum_{k=1}^{n_p} \lambda_k p_k(y_{j_l})$$

we obtain again $\lambda_k = 0$, $1 \le k \le n_p$. Thus P vanishes identically. □

For a fixed $x \in D_X$ denote by f_x the function $f_x(y) = f(x,y)$ and by $L_{p-1}(f_x)$ the interpolation polynomial to f_x of degree $p-1$. We are now in a position to relate the remainder term r_{n_p} of the approximation to the remainder of the Lagrange interpolation $E_p(f_x)(y) = f_x(y) - L_{p-1}(f_x)(y)$.

Lemma 5. *For the functions r_{n_p} it holds that*

$$r_{n_p}(x,y) = E_p(f_x)(y) - \sum_{l=1}^{n_p} \frac{\det M_{n_p}^{(l)}(x)}{\det M_{n_p}} E_p(f_{x_{i_l}})(y).$$

Proof. Let L_k be the kth Lagrange polynomial in Π_{p-1}^d, i.e. $L_k(y_{j_l}) = \delta_{kl}$, and

$$L(y) = \begin{bmatrix} L_1(y) \\ \vdots \\ L_{n_p}(y) \end{bmatrix}$$

the vector of the Lagrange polynomials to the points $y_1, \dots, y_{n_p}$.
Using lemma 3 gives

$$\begin{aligned} r_{n_p}(x,y) &= f(x,y) - f(x,[y]_{n_p})^T M_{n_p}^{-1} f([x]_{n_p}, y) \\ &= f(x,y) - f(x,[y]_{n_p})^T L(y) - \\ &\quad - f(x,[y]_{n_p})^T M_{n_p}^{-1} \left(f([x]_{n_p}, y) - M_{n_p} L(y) \right) \\ &= E_p(f_x)(y) - \sum_{l=1}^{n_p} \left(f(x,[y]_{n_p})^T M_{n_p}^{-1} \right)_l E_p(f_{x_{i_l}})(y). \end{aligned}$$

Since $M_{n_p}^{-T} f(x_{i_j}, [y]_{n_p}) = e_j$ the jth canonical vector, it is easy to check that

$$\frac{\det M_{n_p}^{(l)}(x)}{\det M_{n_p}} = \det M_{n_p}^{(l)}(x) M_{n_p}^{-1} = \left(f(x,[y]_{n_p})^T M_{n_p}^{-1} \right)_l.$$

□

We are able to control the approximation error by making assumptions on the choice of the points $x_{i_1}, \dots, x_{i_{n_p}}$. Assume that our choice of points

$x_{i_1}, \ldots, x_{i_{n_p}}$ leads to a submatrix M_{n_p} whose determinant cannot be increased by interchanging one row by any vector $f(x, [y]_{n_p})$, $x \in D_X$, i.e.

$$|\det M_{n_p}| \geq |\det M_{n_p}^{(l)}(x)|, \quad 1 \leq l \leq n_p,\ x \in D_X. \tag{4}$$

In interpolation theory these maximum volume matrices play an important role (cf. [4], [8]). From the previous lemma we obtain

$$|r_{n_p}(x,y)| \leq (1+n_p) \sup_{x \in D_X} |E_p(f_x)(y)|. \tag{5}$$

Instead of choosing the points $x_{i_1}, \ldots, x_{i_{n_p}}$ from the complicated condition (4) we may choose x_{i_k} in each step so that $r_{k-1}(x_{i_k}, y_{i_k})$ is the maximum element in modulus. From lemma 2 we see that this is the best possible choice with respect to maximum determinants if we keep all other previously chosen elements fixed.

Lemma 6. *Assume in each step we choose* x_{i_k} *so that*

$$|r_{k-1}(x_{i_k}, y_{j_k})| \geq |r_{k-1}(x, y_{j_k})| \quad \textit{for all } x \in D_X.$$

Then for $1 \leq l \leq k$ *it holds that*

$$\sup_{x \in D_X} \frac{|\det M_k^{(l)}(x)|}{|\det M_k|} \leq 2^{k-l}.$$

Proof. Using lemma 2 gives for $1 \leq l < k$

$$\frac{\det M_k^{(l)}(x)}{\det M_k} = \frac{\det M_{k-1}^{(l)}(x)}{\det M_{k-1}} - \frac{r_{k-1}(x, y_{j_k})}{r_{k-1}(x_{i_k}, y_{j_k})} \frac{\det M_{k-1}^{(l)}(x_{i_k})}{\det M_{k-1}}$$

and

$$\frac{\det M_k^{(k)}(x)}{\det M_k} = \frac{r_{k-1}(x, y_{j_k})}{r_{k-1}(x_{i_k}, y_{j_k})}.$$

Thus we obtain for $1 \leq l < k$

$$\sup_{x \in D_X} \frac{|\det M_k^{(l)}(x)|}{|\det M_k|} \leq 2 \sup_{x \in D_X} \frac{|\det M_{k-1}^{(l)}(x)|}{|\det M_{k-1}|}$$

from what the claim follows. □

Consequently instead of (5) we find

$$|r_{n_p}(x,y)| \leq (1+2^{n_p}) \sup_{x \in D_X} |E_p(f_x)(y)|. \tag{6}$$

The following theorem is due to Sauer and Xu. It gives an upper bound for the error of multivariate polynomial interpolation. We will use the same notations as in [18].

Theorem 3. *Assume that the Lagrange interpolation in the points $\boldsymbol{x}^0, \ldots, \boldsymbol{x}^n$ is unique. For $f \in C^{n+1}(\mathbb{R}^d)$ it holds that*

$$|E_{n+1}(f)(x)| \le \sum_{\mu \in \Lambda_n} \frac{1}{(n+1)!} |P_{\mu_n}^{[n]}(x) \pi_\mu(\boldsymbol{x}^\mu)| \, \|D_{x - x_{\mu_n}^{(n)}} D_{\boldsymbol{x}^\mu}^n f\|_\infty, \quad x \in \mathbb{R}^d$$

where it suffices to take the supremum over the convex hull of $\{\boldsymbol{x}^0, \ldots, \boldsymbol{x}^n, x\}$.

Using this expression for the error of Lagrange interpolation we are now able to state our main result. It relies on the fact that the expression $P_{\mu_n}^{[n]}(x)\pi_\mu(\boldsymbol{x}^\mu)$ from the previous theorem does not depend on f and will be formulated for the choice of the points x_{i_k} according to the maximum element strategy, which is much easier to use than the maximum volume strategy.

Theorem 4. *Let x_{i_k} be chosen so that*

$$|r_{k-1}(x_{i_k}, y_{j_k})| \ge |r_{k-1}(x, y_{j_k})| \quad \textit{for all } x \in D_X.$$

Then for an asymptotically smooth function f the sequences r_k and s_k generated at the beginning of section 2 fulfil

$$f(x,y) = s_{n_p}(x,y) + r_{n_p}(x,y),$$

where for $x \in D_X$ and $y \in D_Y$

$$s_{n_p}(x,y) = f(x, [y]_{n_p}) M_{n_p}^{-1} f([x]_{n_p}, y)$$

and

$$|r_{n_p}(x,y)| \le c_p \mathrm{dist}^g(D_X, D_Y)\, \eta^p,$$

where c_p does not depend on η but only on the points $y_1, \ldots, y_{n_p}$.

Proof. We apply the Sauer-Xu formula to $f_x(y)$. For $x \in D_X$, $y \in D_Y$ holds

$$\begin{aligned} |D_{y - y_{\mu_{p-1}}^{(p-1)}} D_{\boldsymbol{y}^\mu}^{p-1} f_x(y)| &\le \mathrm{diam}^p(D_Y) d^p c_1\, p!\, c_2^p \mathrm{dist}^{g-p}(D_X, D_Y) \\ &\le c_1 p!\, (c_2 d\eta)^p \mathrm{dist}^g(D_X, D_Y). \end{aligned}$$

Thus

$$|E_p(f_x)(y)| \le c_1 (c_2 d\eta)^p \mathrm{dist}^g(D_X, D_Y) \sum_{\mu \in \Lambda_{p-1}} |P_{\mu_{p-1}}^{[p-1]}(y) \pi_\mu(\boldsymbol{y}^\mu)|.$$

According to lemma 5 we have

$$|r_{n_p}(x,y)| \le (1 + 2^{n_p}) \sup_{x \in D_X} |E_p(f_x)(y)|,$$

which finally leads to

$$|r_{n_p}(x,y)| \leq c_p \mathrm{dist}^g(D_X, D_Y)\, \eta^p, \tag{7}$$

where we set

$$c_p = c_1(c_2 d)^p(1+2^{n_p}) \sup_{y\in D_Y} \sum_{\mu\in\Lambda_{p-1}} |P^{[p-1]}_{\mu_{p-1}}(y)\pi_\mu(\boldsymbol{y}^\mu)|. \tag{8}$$

□

3 Numerical aspects

In the previous section we showed how to approximate a general asymptotically smooth function in two variables by a sum over functional skeletons, i.e. products of functions of one variable.

The aim of this section is to develop an efficient but simple algorithm for the approximation of matrices generated by asymptotically smooth functions based on the previous interpolation arguments.

Matrices containing blocks of this type usually appear in the solution phase of integral formulations of boundary value problems. Since they are large we have to think about reducing the storage and the number of operations needed when multiplying them with a vector. Though there are cases where a block partitioning of these matrices shows some structure (cf. [16]) which can be exploited with respect to the previous issues, in general they are unstructured and we have to approximate them by easily handled matrices. In our case these are the blockwise low-rank matrices.

Let us reformulate the construction of the functions r_k and s_k in matrix form.

Algorithm 1. *Set*

$$\begin{aligned} (R_0)_{ij} &= f(x_i, y_j), \quad i=1,\ldots,m,\ j=1,\ldots,n \\ S_0 &= 0 \end{aligned}$$

and for $k = 0, 1, \ldots$

$$\begin{aligned} \gamma_{k+1} &= \left(e^T_{i_{k+1}} R_k e_{j_{k+1}}\right)^{-1} \\ R_{k+1} &= R_k - \gamma_{k+1}\, R_k e_{j_{k+1}} e^T_{i_{k+1}}\, R_k \\ S_{k+1} &= S_k + \gamma_{k+1}\, R_k e_{j_{k+1}} e^T_{i_{k+1}}\, R_k \end{aligned}$$

Since we are now looking for the approximation of f in only the points (x_i, y_j) it suffices to determine i_k from the maximum element in modulus in the column j_k, which has to be chosen so that it contains non-zero elements.

Without loss of generality we may assume for the moment that $i_l = j_l = l$, $l = 1, \ldots, k+1$, otherwise interchange the rows and columns of the original matrix R_0.
Then

$$R_{k+1} = (I - \gamma_{k+1} R_k e_{k+1} e_{k+1}^T) R_k = L_{k+1} R_k,$$

where $L_k \in \mathbb{R}^{m \times n}$ is the matrix

$$L_k = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & & 0 & & \\ & & & -\frac{e_{k+1}^T R_{k-1} e_k}{e_k^T R_{k-1} e_k} & 1 & \\ & & & \vdots & & \ddots & \\ & & & -\frac{e_m^T R_{k-1} e_k}{e_k^T R_{k-1} e_k} & & & 1 \end{bmatrix}.$$

L_k differs from a Gaussian matrix only in the position $(k,\, k)$.
In some sense the algorithm produces a column-pivoted LU decomposition for the approximation of a matrix. As we know, it is possible that the elements grow in the LU decomposition algorithm, cf. [6]. Thus the term 2^{n_p} in (6) is not a result of overestimation.
Since for $d > 1$ at least the term 2^{n_p} in (8) grows faster than η^p in (7) decreases, the only way to control the error is by using the geometrical parameter η and keeping the number of approximation steps n_p bounded.
However, we must bear in mind that any growth of elements in the LU decomposition is difficult to observe. In our case this means that the term 2^{n_p} from (8) is not visible in real calculations.

Let us reformulate the algorithm so that it becomes efficient, i.e. uses only a small part of the original matrix. For this purpose define

$$\widetilde{u}_k = R_{k-1} e_{j_k}, \quad v_k = R_{k-1}^T e_{i_k}.$$

Now the algorithm reads

Algorithm 2. *For* $k = 1, 2, \ldots$

$$\begin{aligned}
(\widetilde{u}_k)_i &= f(x_i, y_{j_k}) - \sum_{l=1}^{k-1} (v_l)_{j_k} (u_l)_i, \quad i = 1, \ldots, m \\
u_k &= (\widetilde{u}_k)_{i_k}^{-1} \widetilde{u}_k \\
(v_k)_j &= f(x_{i_k}, y_j) - \sum_{l=1}^{k-1} (u_l)_{i_k} (v_l)_j, \quad j = 1, \ldots, n.
\end{aligned}$$

For the approximant S_k it holds that

$$S_k = \sum_{l=1}^{k} u_l v_l^T.$$

Thus for the whole approximation we need only the evaluation of $f(x_i, y_{j_k})$, $i = 1, \ldots, m$, and $f(x_{i_k}, y_j)$, $j = 1, \ldots, n$. The rest is algebraic transformations, which are easy to implement, whereby it should be remembered that the entries of u_k at the positions i_l and the entries of v_k at the positions j_l, $l < k$, are zero.

To obtain S_k we need $(m+n)k$ units of storage and

$(m+n)k$	evaluations of f,
$(m+n)(k-1)k$	additions and multiplications,
mk	divisions.

Thus we need $O((m+n)k^2)$ operations to generate the approximant S_k.

Although we will use a bounded number of iteration steps it is useful to stop the algorithm if a prescribed accuracy is reached. Since we do not want to calculate the whole original matrix we cannot compute the error exactly. Thus a good approximation for the error is the only way to control the algorithm.

Since

$$R_{n_{p+1}} = R_{n_p} - \sum_{l=n_p+1}^{n_{p+1}} u_l v_l^T$$

and $\|R_{n_{p+1}}\|_F$ is of one order smaller than $\|R_{n_p}\|_F$ the value

$$\| \sum_{l=n_p+1}^{n_{p+1}} u_l v_l^T \|_F$$

may be used as a good approximation to $\|R_{n_p}\|_F$.
When computing $\|\sum_{l=n_p+1}^{n_{p+1}} u_l v_l^T\|_F$ it should be noted that

$$\|\sum_{l=n_p+1}^{n_{p+1}} u_l v_l^T\|_F^2 = \sum_{l,k=n_p+1}^{n_{p+1}} \left(\sum_{i=1}^{m}(u_l)_i(u_k)_i\right)\left(\sum_{j=1}^{n}(v_l)_j(v_k)_j\right).$$

The evaluation of the last expression can be made in $(m+n)(r_p^d)^2$ operations.

The last lemma in this section shows whether the algorithm is capable of finding the rank of a matrix. Assume that $\operatorname{rank} A = r$. Then r steps of our algorithm are sufficient to take over the whole matrix. Thus this algorithm is always finite.

Lemma 7. *Let $A \in \mathbb{C}^{m\times n}$ be of rank r. Then $S_r = A$.*

Proof. If $r = 1$ then it is obvious that $R_1 = 0$ and thus $S_1 = A$. Let us look at the general case $r \in \mathbb{N}$ and apply one step of our algorithm to A. Without loss of generality let $a_{11} \neq 0$. We realize that

$$A \begin{bmatrix} 1 & & & \\ -\frac{a_{21}}{a_{11}} & 1 & & \\ \vdots & & \ddots & \\ -\frac{a_{m1}}{a_{11}} & & & 1 \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & \cdots & a_{1n} \\ 0 & & & \\ \vdots & & \hat{A} & \\ 0 & & & \end{bmatrix}$$

Thus $\operatorname{rank} \hat{A} = \operatorname{rank} A - 1$. On the other hand we see that

$$\widetilde{A} = A - \frac{1}{a_{11}} A e_1 e_1^T A = \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \hat{A} & \\ 0 & & \end{bmatrix}$$

This means that $\operatorname{rank} \widetilde{A} = \operatorname{rank} \hat{A} = \operatorname{rank} A - 1$. □

From the previous proof we see that our algorithm successively reduces the rank.

3.1 Matrix partitioning

Our aim in this subsection is to show how the previous arguments can be applied if the domains where the points come from are not well separated, i.e. do not fulfil condition (2).
Assume we approximate the surface Γ on which the boundary data is given

by Γ_h using the set of triangles (panels) P. Our aim is to partition the matrix $A \in R^{N\times N}$, $N = \#P$ into submatrices so that for two sets of panels (so called clusters) τ_1, τ_2 corresponding to a submatrix

$$\text{diam } \tau_2 \le \eta \, \text{dist}(\tau_1, \tau_2) \quad \text{for some } 0 < \eta < 1$$

holds (cf. (2)) or one of them has just one element. In analogy to [13] we call a cluster pair admissible if it fulfils the previous geometrical condition. In this case and if τ_1, $\tau_2 \notin P$ the corresponding block will be approximated as described previously. All other blocks, i.e. blocks with τ_1 or $\tau_2 \in P$ will be computed exactly.

In [13] a set of clusters T that possesses a tree structure is used to suitably subdivide the set P with respect to a fixed point. We will use a set of cluster pairs having a tree structure for the partitioning of $P \times P$ the Cartesian product of the set of panels with itself. This set T' is constructed from the set T by applying the following recursion to (Γ_h, Γ_h).

Algorithm 3. *Take a cluster pair (τ_1, τ_2), $\tau_1, \tau_2 \in T$. If τ_1 and τ_2 both have children τ_{11}, τ_{12} and τ_{21}, τ_{22} in T respectively, then assign the pairs (τ_{11}, τ_{21}), (τ_{11}, τ_{22}), (τ_{12}, τ_{21}) and (τ_{12}, τ_{22}) as children to the cluster pair (τ_1, τ_2) and add them to T'. Now repeat the previous steps with each child.*

It is easy to see that in each level in the tree structure of T' all pairs of clusters (τ_1, τ_2) appear, based on τ_1, τ_2 from the corresponding level in the tree structure of T.

In the following we will show how many blocks are needed in order to decompose the matrix while each cluster pair corresponding to a block is admissible or one of the clusters is a panel. First we have to impose some conditions on the set of panels P. We assume that there are constants c_u and c_a so that

$$c_u \text{diam } \pi > h \quad \text{for all } \pi \in P, \tag{9}$$

where $h = \max\limits_{\pi \in P} \text{diam } \pi$ and

$$\text{area}\,(B_r(z) \cap \Gamma_h) \le c_a (2r)^{d-1} \quad \text{for all } r > 0 \text{ and } z \in \mathbb{R}^d, \tag{10}$$

where $B_r(z) = \{x \in \mathbb{R}^d, \|x - z\| < r\}$.

From this follows

$$h^{d-1} \ge c_b N^{-1} \tag{11}$$

with $c_b = \text{area}\,(\Gamma_h)/c_a$, because

$$\text{area}\,(\Gamma_h) = \sum_{\pi \in P} \text{area } \pi \le N \max_{\pi \in P} \text{area}\,(\pi) \le c_a N h^{d-1}.$$

Without loss of generality we look at $\mathbb{R}^d$ as being normed by $\|\cdot\|_\infty$.
In [13] it was proven that for all $0 < r \leq R$ and $a \in \mathbb{R}^d$ there exists a set $C \subseteq T$ of clusters with

$$\Gamma_h \cap \overline{B_R(a)} \subseteq \bigcup_{\tau \in C} \tau,$$

$$\operatorname{diam} \tau \leq r \quad \text{for all } \tau \in C \setminus P$$

and

$$\# C \leq c_P \left(\frac{R}{r}\right)^{d-1}.$$

Furthermore we realize that there are constants $\overline{c}$ and $\underline{c}$ such that for a cluster τ in T of level l it holds that

$$\operatorname{diam}^{d-1} \tau \leq \underline{c}\, 2^{-l} \text{area } \Gamma_h \quad \text{and} \quad \text{area } \tau \geq \overline{c}\, 2^{-l} \text{area } \Gamma_h. \tag{12}$$

From these properties of the set of clusters T we obtain the following analogue for the set T'.

Lemma 8. *For all $0 < r \leq R$ and $a, b \in \mathbb{R}^d$ there exists a set $C' \subseteq T'$ of cluster pairs fulfilling*

$$(\Gamma_h \times \Gamma_h) \cap (\overline{B_R(a)} \times \overline{B_R(b)}) \subseteq \bigcup_{(\tau_1, \tau_2) \in C'} \tau_1 \times \tau_2,$$

$$\max\{\operatorname{diam} \tau_1, \operatorname{diam} \tau_2\} \leq r \quad \textit{for all } (\tau_1, \tau_2) \in C',\ \tau_1,\ \tau_2 \notin P$$

and

$$\# C' \leq c'_P \left(\frac{R}{r}\right)^{2(d-1)}.$$

Proof. We already know that for all $0 < r \leq R$ and all $a, b \in \mathbb{R}^d$ there are two sets of clusters C_1, C_2 fulfilling

$$\Gamma_h \cap \overline{B_R(a)} \subseteq \bigcup_{\tau \in C_1} \tau, \quad \Gamma_h \cap \overline{B_R(b)} \subseteq \bigcup_{\tau \in C_2} \tau, \tag{13a}$$

$$\operatorname{diam} \tau \leq r \quad \text{for all } \tau \in (C_1 \cup C_2) \setminus P \tag{13b}$$

and

$$\# \widetilde{C} \leq c_P^2 \left(\frac{R}{r}\right)^{2(d-1)}, \quad \widetilde{C} = C_1 \times C_2 \tag{13c}$$

where the clusters in C_1 and C_2 are chosen to have lowest possible level.
To obtain a set of cluster pairs $C' \subseteq T'$ we have to guarantee that the levels of the clusters in each pair are the same.
Now let $(\tau_1, \tau_2) \in \widetilde{C}$ and $l_1 \le l_2$ be the levels of τ_1 and τ_2 in T respectively. If $l_1 < l_2$ then we first consider the case $\tau_1 \in P$.
Let τ_2^f be an ancestor of τ_2 of level l_1 in T. Remove from $\widetilde{C}$ all pairs (τ_1, τ), where τ is any descendant of τ_2^f and add (τ_1, τ_2^f) to it. Then (13) remains valid if we replace (13b) by

$$\max\{\operatorname{diam} \tau_1, \operatorname{diam} \tau_2\} \le r \quad \text{for all } (\tau_1, \tau_2) \in \widetilde{C},\ \tau_1,\, \tau_2 \notin P.$$

Now let $\tau_1 \notin P$. Divide τ_1 according to T into clusters of level at most l_2. If this division cannot be continued for a τ then $\tau \in P$ and we go back to the previous case.
Since $\overline{c}\, 2^{-l_1}$area $\Gamma_h \le$ area $\tau_1 \le c_a$diam$^{d-1}\, \tau_1 \le c_a r^{d-1}$ we obtain

$$l_1 \ge \log_2 \left(\overline{c} \frac{\text{area } \Gamma_h}{c_a} r^{-(d-1)} \right).$$

And because the level of τ_2 is minimal, i.e. for the father τ_2^f of τ_2 in T it holds that diam $\tau_2^f > r$, we have $r^{d-1} <$ diam$^{d-1}\, \tau_2^f \le \underline{c}\, 2^{-(l_2-1)}$area Γ_h.
Thus

$$l_2 \le \log_2 \left(2\underline{c}\, \text{area } \Gamma_h r^{-(d-1)} \right)$$

and hence

$$l_2 - l_1 \le \log_2 \left(2 c_a \underline{c} / \overline{c} \right).$$

We see that τ_1 is subdivided into at most $2 c_a \overline{c} / \underline{c}$ clusters.
Generate the set C' by assembling these new cluster pairs. This guarantees that $C' \subseteq T'$ and (13c) changes to

$$\# C' \le c'_P \left(\frac{R}{r} \right)^{2(d-1)},$$

where $c'_P = 2 c_a \overline{c} / \underline{c}\, c_P^2$. □

It is obvious that the covering of $\Gamma_h \times \Gamma_h$ with the smallest number of cluster pairs can be computed by the following algorithm.

Algorithm 4. *Set $D = \emptyset$ and call Divide$((\Gamma_h, \Gamma_h), D)$ where Divide is the following recursive procedure.*

```
procedure Divide($((\tau_1, \tau_2), D)$)
  if $(\tau_1, \tau_2)$ is admissible or $\tau_1 \in P$ or $\tau_2 \in P$ then $D := D \cup \{(\tau_1, \tau_2)\}$
  else apply the procedure to each of the children of $(\tau_1, \tau_2)$ in $T'$.
```

Let $\mathcal{N}(a)$ denote the number of cluster pairs needed to produce a covering of

$$\bigcup\{\pi \in P : \pi \cap B_1 \neq \emptyset\} \times \bigcup\{\pi \in P : \pi \cap B_2 \neq \emptyset\},$$

where B_1, B_2 are any cubes of side length a, while this covering should only contain admissible cluster pairs or pairs with one of the two clusters being a panel. Furthermore, let $\mathcal{N}_{st}(a)$ be the amount of storage when blockwise approximating the part of the original matrix that corresponds to the two cubes B_1 and B_2. Blocks that correspond to cluster pairs with one of the two clusters being a panel will be stored without approximation. On all other blocks, i.e. admissible blocks that correspond to clusters $\tau_1, \tau_2 \notin P$, we perform a rank-k approximation.
If we enclose Γ_h in a cube of side length a_0 then $\mathcal{N}(a_0)$ and $\mathcal{N}_{st}(a_0)$ will be the values for the whole matrix.
Let $\beta > 0$ be arbitrarily small. We will show that

$$\mathcal{N}(a_0) = O(N^{1+\beta}\eta^{-2(d-1)})$$

and

$$\mathcal{N}_{st}(a_0) = O(kN^{1+\beta}\eta^{-2(d-1)}).$$

Recently Hackbusch and Khoromskij [11], [12] have proven that the storage requirement for such matrices is $O(N \log N)$. However, they did not investigate the dependency on the parameter η which is essential for our algorithm since the approximation error will be controlled by this parameter.
We first investigate the case when B_1 and B_2 are two neighbouring or identical cubes.

Lemma 9. *Assume that B_1 and B_2 are two neighbouring or identical cubes of side length a. Let $q \in \mathbb{N}$, then for the numbers $\mathcal{N}(a)$ and $\mathcal{N}_{st}(a)$ it holds that*

$$\mathcal{N}(a) \le c_1 \left(\frac{q}{\eta}\right)^{2(d-1)} + c_2 q^{d-1} \mathcal{N}(a/q)$$

and

$$\mathcal{N}_{st}(a) \le c_3 k \left(\frac{q}{\eta}\right)^{2(d-1)} \left(1 + \left(\frac{a}{qh}\right)^{d-1}\right) + c_2 q^{d-1} \mathcal{N}_{st}(a/q).$$

Proof. We subdivide each of the cubes B_1 and B_2 into q^d subcubes of side length $R = \frac{a}{q}$. There are at most $c_0 q^{d-1}$ subcubes that contain parts of the boundary Γ_h. From these at most $(c_0 q^{d-1})^2$ pairs take out a pair of non-adjacent subcubes $B_R(z_1)$, $B_R(z_2)$. According to the previous property of the set T' for R and $r = \frac{1}{4}\eta R$ we can find a set of cluster pairs $C' \subseteq T'$ so that

$$(\Gamma_h \times \Gamma_h) \cap (\overline{B_R(z_1)} \times \overline{B_R(z_2)}) \subseteq \bigcup_{(\tau_1,\tau_2)\in C'} \tau_1 \times \tau_2,$$

$$\max\{\text{diam } \tau_1, \text{diam } \tau_2\} \le r \quad \text{for all } (\tau_1,\tau_2) \in C',\ \tau_1,\ \tau_2 \notin P,$$

$$\#C' \le c'_P \left(\frac{R}{r}\right)^{2(d-1)} \le c'_P 4^{2(d-1)}\eta^{-2(d-1)} =: \widetilde{c}_P \eta^{-2(d-1)}.$$

If $\tau_1, \tau_2 \notin P$

$$\text{dist}(\tau_1,\tau_2) \ge R - 2r = R\left(1 - \frac{\eta}{2}\right) \ge \frac{1}{2}\frac{a}{q}, \quad \text{since } \eta < 1$$

and diam $\tau_2 \le \frac{\eta}{4}\frac{a}{q} < \eta\,\text{dist}(\tau_1,\tau_2)$. Thus this cluster pair is admissible.
Since among the at most $(c_0 q^{d-1})^2$ pairs of subcubes B with $B \cap \Gamma_h \ne \emptyset$ there are at most $3^d c_0 q^{d-1}$ pairs of adjacent subcubes we see that

$$\mathcal{N}(a) \le (c_0 q^{d-1})^2\, \widetilde{c}_P \eta^{-2(d-1)} + 3^d c_0 q^{d-1} \mathcal{N}(a/q).$$

In a cluster τ of diameter at most r not more than $c_a \underline{c}/\overline{c}\,(c_u r/h)^{d-1}$ panels fit, because according to (9) and (12)

$$c_a r^{d-1} \ge c_a \text{diam}^{d-1}\,\tau \ge \text{area } \tau \ge \#\tau \min_{\pi\in\tau} \text{area } \pi \ge \#\tau \overline{c}/\underline{c}\,(h/c_u)^{d-1}.$$

If one of the clusters τ_1, τ_2 is a panel, let this be τ_1, the corresponding block is stored without approximation. Since τ_1 and τ_2 are from the same level l in T

$$\text{diam}^{d-1}\,\tau_2 \le \underline{c}\,2^{-l}\text{area } \Gamma_h \le \underline{c}/\overline{c}\,\text{area } \tau_1 \le \underline{c}/\overline{c}\,c_a h^{d-1}.$$

Thus in this case we can find a constant $C \ge 1$ which serves as an upper bound for the memory usage for this block.
If none of the clusters is a panel then the storage requirement is less than $2kc_a\underline{c}/\overline{c}\,(c_u r/h)^{d-1}$, because it takes $k(n+m)$ units of memory to store a rank-k matrix in $\mathbb{R}^{m\times n}$. Thus we need at most

$$\max\left(C,\, 2kc_a\underline{c}/\overline{c}\left(c_u\frac{a\eta}{4qh}\right)^{d-1}\right) \le C + 2kc_a\underline{c}/\overline{c}\left(c_u\frac{a}{qh}\right)^{d-1} =: M$$

units of memory to store one block. Consequently, for $\mathcal{N}_{st}(a)$ follows

$$\mathcal{N}_{st}(a) \le (c_0 q^{d-1})^2\, \widetilde{c}_P \eta^{-2(d-1)} M + 3^d c_0 q^{d-1} \mathcal{N}_{st}(a/q).$$

□

Using the previous case we are able to find an estimate for $\mathcal{N}(a_0)$ and $\mathcal{N}_{st}(a_0)$.

Lemma 10. *Let $\beta > 0$. For the numbers $\mathcal{N}(a_0)$ and $\mathcal{N}_{st}(a_0)$ it holds that*

$$\mathcal{N}(a_0) \leq c' N^{1+\beta} \eta^{-2(d-1)}$$

and

$$\mathcal{N}_{st}(a_0) \leq c'' k N^{1+\beta} \eta^{-2(d-1)},$$

where c' and c'' depend on β but not on η or N.

Proof. From the previous lemma it can be seen that ($\widetilde{q} = q^{d-1}$)

$$\mathcal{N}_{st}(a_0) \leq c_3 k \eta^{-2(d-1)} \widetilde{q}^2 \sum_{l=0}^{L-1} (c_2 \widetilde{q})^l \left(1 + \widetilde{q}^{-l-1} \left(\frac{a_0}{h}\right)^{d-1}\right) + (c_2 \widetilde{q})^L \mathcal{N}_{st}(a_0/q^L),$$

with L the smallest integer so that $L \geq \log_q(a_0/h)$, where $q \in \mathbb{N}$ is chosen so that $\widetilde{q}^\beta \geq c_2$. This guarantees that $\widetilde{q}^{-l} (a_0/h)^{d-1} \geq 1$ for $l \leq L-1$ and leads to

$$\mathcal{N}_{st}(a_0) \leq 2 c_3 k \eta^{-2(d-1)} \widetilde{q}^2 \left(\frac{a_0}{h}\right)^{d-1} \sum_{l=0}^{L-1} c_2^l + (c_2 \widetilde{q})^L \mathcal{N}_{st}(a_0/q^L).$$

We may assume that $c_2 \geq 2$. Since $\widetilde{q}^L \geq (a_0/h)^{d-1}$ and with (11)

$$\widetilde{q}^{L-1} \leq a_0^{d-1}/h^{d-1} \leq a_0^{d-1} N/c_b$$

we obtain

$$\begin{aligned}\mathcal{N}_{st}(a_0) &\leq \left(4 c_3 k \eta^{-2(d-1)} \left(\frac{a_0}{h}\right)^{d-1} \widetilde{q}^3 \widetilde{q}^{-L} + c_2 \widetilde{q} \mathcal{N}_{st}(a_0/q^L)\right) (c_2 \widetilde{q})^{L-1} \\ &\leq (4 c_3 k \eta^{-2(d-1)} \widetilde{q}^3 + c_2 \widetilde{q} \mathcal{N}_{st}(a_0/q^L))(a_0^{d-1} N/c_b)^{1+\beta}.\end{aligned}$$

A cube B of side length h can be covered by a set $\tau = \{\pi \in P : B \cap \pi \neq \emptyset\}$. Then diam $\tau \leq 3h$ and from the proof of the last lemma we see that τ contains at most $c_a \underline{c}/\overline{c}\,(3c_u)^{d-1}$ panels and hence $\mathcal{N}_{st}(a_0/q^L) \leq c$, where $c = (c_a \underline{c}/\overline{c}\,(3c_u)^{d-1})^2$. The proof for the second bound is similar. □

3.2 Computational complexity

In the following we will show how many operations and units of memory are necessary to generate an approximant of the discrete operator to a prescribed accuracy ε and how many operations are needed to perform one matrix-vector multiplication using this approximant.

The complexity of the whole algorithm is mainly determined by the number of cluster pairs $\mathcal{N}(a_0)$. Let us denote the set of blocks that correspond to an admissible cluster pair (τ_1, τ_2), $\tau_1, \tau_2 \notin P$, by S_a.
To approximate a block $M \in S_a$ of size $m \times n$ with a corresponding cluster pair (τ_1, τ_2) we need n_p skeletons to obtain an accuracy (cf. theorem 4)

$$\|M - \widetilde{M}\|_F \leq \sqrt{nm}\, c_p \operatorname{dist}^g(\tau_1, \tau_2)\, \eta^p.$$

The constant c_p only depends on the points $y_1, \ldots, y_{n_p}$ which may change with N so that c_p is unbounded even for a bounded p. This situation occurs, for example, if the points are located in the intersections of two families of parallel lines and the angle between them vanishes with growing N, cf. [5]. However, these cases are pathological and we assume a sequence of boundary meshes that guarantees that c_p is bounded for bounded p.
Since for an admissible cluster pair (τ_1, τ_2)

$$\operatorname{diam} \tau_2 \leq \eta \operatorname{dist}(\tau_1, \tau_2)$$

and according to (9) $\operatorname{diam} \pi \geq h/c_u$ holds for any $\pi \in P$, we see that

$$\operatorname{dist}(\tau_1, \tau_2) \geq \eta^{-1} h/c_u > h/c_u$$

and since we assumed $g \leq 0$ then

$$\operatorname{dist}^g(\tau_1, \tau_2) \leq (h/c_u)^g \leq c_u^{-g} (c_b N^{-1})^{\frac{g}{d-1}}.$$

While the blocks in S_a are approximated, all other blocks are computed exactly. Thus for the approximation error it holds that

$$\begin{aligned} \|A - \widetilde{A}\|_F^2 &= \sum_{M \in S_a} \|M - \widetilde{M}\|_F^2 \\ &\leq \sum_{M \in S_a} nm\, c_p^2 \operatorname{dist}^{2g}(\tau_1, \tau_2)\, \eta^{2p} = O((N^{1-\frac{g}{d-1}} \eta^p)^2). \end{aligned}$$

Let $\alpha > 0$ and $0 < \beta < \alpha$, β from lemma 10, and set

$$\eta = \left(\varepsilon N^{\frac{g}{d-1} - 1}\right)^{1/p}$$

where p is the smallest integer so that

$$p \geq 2 \frac{d - 1 - g}{\alpha - \beta}.$$

For reasonable ε and N we may assume that $0 < \eta < 1$.
With this choice of η for the approximation error it holds that

$$\|A - \widetilde{A}\|_F = O(\varepsilon).$$

Since

$$\mathcal{N}(a_0) = O(N^{1+\alpha}\varepsilon^{-\alpha})$$

the number of steps needed to obtain the set of cluster pairs is bounded by

$$\frac{1}{3}(4\mathcal{N}(a_0) - 1) = O(N^{1+\alpha}\varepsilon^{-\alpha}).$$

For the amount of storage needed

$$\mathcal{N}_{st}(a_0) = O(N^{1+\alpha}\varepsilon^{-\alpha})$$

holds. Thus the number of operations needed for one matrix-vector multiplication and the number of operations for the generation of the approximant do not exceed $O(N^{1+\alpha}\varepsilon^{-\alpha})$, because on each block these numbers differ from the amount of storage only by a constant factor.
Thus the overall complexity of this method is $O(N^{1+\alpha}\varepsilon^{-\alpha})$, $\alpha > 0$ arbitrarily small.

4 Numerical experiments

We will test our algorithm on the following surface

$$\Gamma = \left\{ x = \begin{pmatrix} R(z)\cos(2\pi t) \\ R(z)\sin(2\pi t)\,(2 - \frac{3}{2}\sin(2\pi t)) \\ z \end{pmatrix},\ 0 \le z \le 1,\ 0 \le t < 1 \right\}$$

where

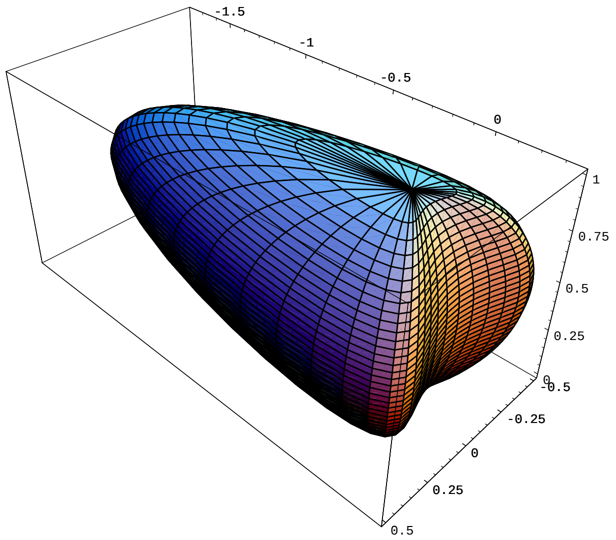
$$R(z) = \sqrt{z(1-z)}$$

and on the boundary integral equation which appears for the inner Dirichlet problem

$$\mathcal{A}u = f$$

where $\mathcal{A}$ is the single layer operator

$$\mathcal{A}u(x) = \frac{1}{4\pi}\int_\Gamma \frac{u(y)}{|x-y|}\,ds_y.$$

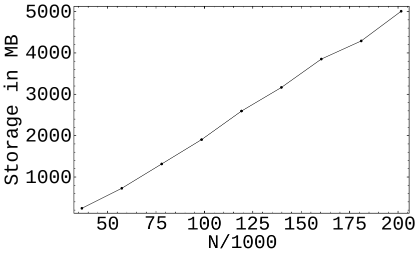


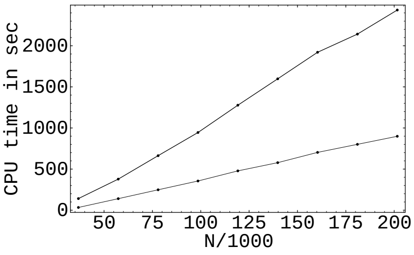
In the following table we show for different problem sizes N and $\varepsilon = 10^{-4}$ the memory and time consumption of the algorithm proposed in this paper. The integral equation is discretized by the collocation method.

Since we did not employ a preconditioner the number of iteration steps in BiCGStab increases with N, so that we merely need to compare the time we needed per iteration. All calculations were performed on an SGI Indigo^2-10k. The memory usage of our algorithm and of the standard solution strategy is shown in the second and third column. The time we needed to generate the approximant, the time for the solution of the linear system and the time per iteration step of BiCGStab is placed in columns five, six and seven.

	Storage (MB)			CPU time (sec)		
N	used	standard	%	approx.	solution	per step
16128	244	1985	12.3	141	428	33
39600	728	11964	6.1	378	1824	140
65024	1316	32258	4.1	663	3713	248
89400	1905	60976	3.1	945	5319	355
114920	2595	100758	2.6	1277	8128	478
136160	3166	141446	2.3	1598	10395	578
159200	3852	193364	2.0	1921	13355	703
175560	4290	235148	1.8	2142	14415	801
201600	5008	310078	1.6	2435	17089	899

The time per multiplication step and for the generation of the approximant grows almost linearly. The same behaviour is observable with respect to storage. The amount of storage and the CPU time for the approximation and the CPU time per step of BiCGStab are depicted in the following figures.





Acknowledgement

The author wishes to thank S. Rjasanow and E. Tyrtyshnikov for useful remarks and the German Ministry for Education and Research for financial support.

References

[1] A. Brandt. Multilevel computations of integral transforms and particle interactions with oscillatory kernels. *Comput. Phys. Comm.*, 65:24–38, 1991.

[2] C. de Boor. On the Sauer-Xu formula for the error in multivariate polynomial interpolation. *Math. Comp.*, 65(215):1231–1234, 1996.

[3] G. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrica*, 1 : 211–218, 1936.

[4] D. Gaier. *Vorlesungen über Approximation im Complexen.* Birkhäuser Verlag, 1980.

[5] M. Gasca and T. Sauer. On bivariate Hermite interpolation with minimal degree polynomials. *manuscript*, 1997.

[6] G. H. Golub and C. F. Van Loan. *Matrix computations.* Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[7] S. A. Goreinov, E. E. Tyrtyshnikov, and A. Y. Yeremin. Matrix-free iterative solution strategies for large dense linear systems. *Numer. Linear Algebra Appl.*, 4(4):273–294, 1997.

[8] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. A theory of pseudoskeleton approximations. *Linear Algebra Appl.*, 261:1–21, 1997.

[9] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comput. Phys.*, 73(2):325–348, 1987.

[10] L. Greengard and V. Rokhlin. The rapid evaluation of potential fields in three dimensions. In *Vortex methods (Los Angeles, CA, 1987)*, pages 121–141. Springer, Berlin, 1988.

[11] W. Hackbusch. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. I. Introduction to $\mathcal{H}$-matrices. *Computing*, 62(2):89–108, 1999.

[12] W. Hackbusch and B. N. Khoromskij. A Sparse $\mathcal{H}$-Matrix Arithmetic. Part II: Application to Multi-Dimensional Problems. Technical report, Max-Planck-Institut, Leipzig, 1999.

[13] W. Hackbusch and Z. P. Nowak. On the fast matrix multiplication in the boundary element method by panel clustering. *Numer. Math.*, 54(4):463–491, 1989.

[14] F. T. Korsmeyer, J. Phillips, K. Nabors, and J. White. Some empirical results on using multipole-accelerated iterative methods to solve 3-D potential integral equations. In *Parallel numerical algorithms (Hampton, VA, 1994)*, pages 267–288. Kluwer Acad. Publ., Dordrecht, 1997.

[15] L. Mirsky. Symmetric gauge functions and unitarily invariant norms. *Quart. J. Math. Oxford Ser. (2)*, 11:50–59, 1960.

[16] S. Rjasanow. Effective algorithms with circulant-block matrices. *Linear Algebra Appl.*, 202:55–69, 1994.

[17] V. Rokhlin. Rapid solution of integral equations of classical potential theory. *J. Comput. Phys.*, 60(2):187–207, 1985.

[18] T. Sauer and Y. Xu. On multivariate Lagrange interpolation. *Math. Comp.*, 64(211):1147–1170, 1995.

[19] E. Tyrtyshnikov. Mosaic-skeleton approximations. *Calcolo*, 33(1-2):47–57 (1998), 1996. Toeplitz matrices: structures, algorithms and applications (Cortona, 1996).

[20] E. Tyrtyshnikov. Mosaic ranks and skeletons. In *Numerical analysis and its applications (Rousse, 1996)*, pages 505–516. Springer, Berlin, 1997.