Fast Boundary Element Methods in Computational Electromagnetism

Stefan Kurz[1], Oliver Rain[1] and Sergej Rjasanow[2]

[1] Robert Bosch GmbH, Postfach 10 60 50, 70049 Stuttgart, Germany
oliver.rain@de.bosch.com

[2] Universität des Saarlandes, Fachrichtung 6.1 - Mathematik, Postfach 15 11 50, 66041 Saarbrücken, Germany
rjasanow@num.uni-sb.de

Summary. When the Boundary Element Method (BEM) is used to analyse electromagnetic problems one is able to achieve an almost linear complexity by applying matrix compression techniques. Beyond this, on symmetrical domains the computational costs can be reduced by significant factors. By using several symmetry considerations (geometry, mesh, kernel, excitation) it will be shown how the combination of the Adaptive Cross Approximation (ACA) and the symmetry exploitation allows an efficient solution of electromagnetic problems. This approach will be demonstrated on the scalar BEM formulation for electrostatics and can also be applied to the vectorial eddy current formulations. The symmetry exploting ACA algorithm not only reduces the problem size due to the symmetry but also possesses an almost linear complexity w.r.t. the number of unknowns.

1 Introduction

Electromagnetic devices can be analysed by the coupled BE-FE method, where the conducting and magnetic parts are discretised by finite elements. In contrast, the surrounding space is described with the help of the boundary element method (BEM). This discretisation scheme is well suited especially for problems including moving parts [11]. The BEM discretisation of the boundary integral operators usually leads to dense matrices without any structure. A naive strategy for the solution of the corresponding linear system would need at least $O(N^2)$ operations and memory, where N is the number of unknowns. Methods such as fast multipole [6] and panel clustering [9] provide an approximation to the matrix in almost linear complexity. These methods are based on explicitly given kernel approximations by degenerate kernels, i.e. a finite sum of separable functions, which may be seen as a blockwise low-rank approximation of the system matrix. The blockwise approximant permits a fast matrix-vector multiplication, which can be exploited in iterative solvers, and can be stored efficiently. In contrast to the methods mentioned the ACA algorithm

[2, 3] generates the low-rank approximant from the matrix itself using only few entries and without using any explicit a priori known degenerate-kernel approximation. Special emphasis is put on the handling of symmetry conditions in connection with ACA [13]. The feasibility of the proposed method is demonstrated by means of numerical examples.

2 Statement of the Problem

The electromagnetic phenomena are described by Maxwell's equations, which can be written in the form of partial differential equations as follows

$$\mathbf{curl}\mathbf{H} = \boldsymbol{\jmath} + \partial_t \mathbf{D}\,, \tag{1}$$

$$\mathbf{curl}\mathbf{E} = -\partial_t \mathbf{B}\,, \tag{2}$$

$$\operatorname{div}\mathbf{B} = 0\,, \tag{3}$$

$$\operatorname{div}\mathbf{D} = \rho\,. \tag{4}$$

The equations describe the correlation between the magnetic field $\mathbf{H}$, magnetic induction $\mathbf{B}$, electric field $\mathbf{E}$ and electric displacement $\mathbf{D}$. $\boldsymbol{\jmath}$ denotes the electric current density and ρ the electric charge density. The equations have to be supplemented by the material laws

$$\mathbf{B} = \mu\mathbf{H}\,, \tag{5}$$

$$\mathbf{D} = \varepsilon\mathbf{E}\,, \tag{6}$$

$$\boldsymbol{\jmath} = \kappa\mathbf{E} + \boldsymbol{\jmath}_S\,, \tag{7}$$

where μ is the magnetic permeability, ε the electric permittivity, κ denotes the electric conductivity and $\boldsymbol{\jmath}_S$ the impressed source current density.

2.1 Formulation of the problem

For the sake of simplicity we consider in the sequel the electrostatic case

$$\mathbf{curl}\mathbf{E} = 0\,, \tag{8}$$

$$\operatorname{div}\mathbf{D} = \rho\,, \tag{9}$$

$$\mathbf{D} = \varepsilon\mathbf{E}\,. \tag{10}$$

Based on the potential ansatz

$$\mathbf{E} = -\mathbf{grad}\varphi\,, \tag{11}$$

where φ is the electric scalar potential, we obtain the potential formulation

$$\operatorname{div}\varepsilon\,\mathbf{grad}\phi = -\rho\,, \tag{12}$$

which has to be solved in the whole $\mathbb{R}^3$. In order to apply the BE-FE discretisation scheme we perform a domain decomposition (see Fig. 1) of the computational space into the bounded domain Ω_{FEM} containing dielectric components, the unbounded domain Ω_{BEM} and the coupling boundary $\Gamma = \bar{\Omega}_{\mathrm{FEM}} \cap \bar{\Omega}_{\mathrm{BEM}}$. In this paper we put the emphasis on the BE formulation.

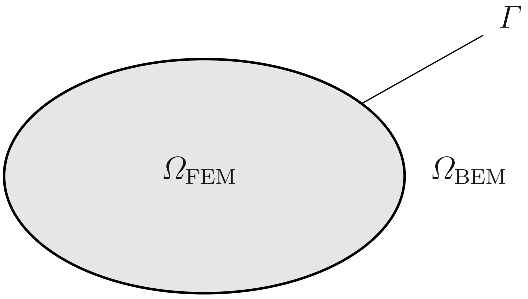

Fig. 1. Domain Decomposition.

Representation Formula

In the BE domain which usually describes the surrounding air we assume $\varepsilon \equiv \varepsilon_0$. Thus, the potential formulation (12) turns out to be the Poisson equation

$$\Delta\phi = -\frac{1}{\varepsilon_0}\,\rho\,. \tag{13}$$

Multiplying (13) by the fundamental solution of the Laplacian

$$u^*(x,y) = \frac{1}{4\pi|x-y|} \tag{14}$$

and performing integration by parts yields the representation formula for smooth boundary points $y \in \Gamma$

$$\frac{1}{2}\phi(y) = \int\limits_{\Gamma} u^*(x,y)\,\partial_n\phi(x)\,dS_x - \int\limits_{\Gamma} \partial_n u^*(x,y)\,\phi(x)\,dS_x + \varepsilon_0^{-1}\int\limits_{\Omega_{\mathrm{BEM}}} u^*(x,y)\,\rho(x)\,dx\,. \tag{15}$$

2.2 Discretisation

First, the spatial discretisation has to be introduced. Let

$$\Gamma_h = \bigcup_{j=1}^{N_{\mathrm{El}}} \Gamma_j$$

be a union of boundary elements Γ_j approximating the coupling boundary Γ and

$$\left\{\phi_j\,,\ j=1,\ldots,N_\phi\right\} \text{ and } \left\{\psi_j\,,\ j=1,\ldots,N_\psi\right\} \tag{16}$$

systems of compact-supported ansatz functions for Dirichlet and Neumann data, respectively. The scalar electric potential and its normal derivative are discretised by the corresponding scalar ansatz functions

$$\phi(x) \approx \sum_{j=1}^{N_\phi} a_j \phi_j(x)\,, \tag{17}$$

$$\partial_n \phi(x) \approx \sum_{j=1}^{N_\psi} q_j \psi_j(x) \tag{18}$$

For the Neumann-type problem the representation formula (15) will be evaluated in N_ϕ collocation points $\{y_i\,,\ i = 1,\dots,N_\phi\}$. Together with the discretisation (17-18) this yields N_ϕ discrete boundary integral equations which can be written as the matrix equation

$$(\,\frac{1}{2}\,I + H\,)\,\mathbf{a} = G\mathbf{q} + \mathbf{b} \tag{19}$$

with the matrices of the single and double layer potential

$$g_{ij} = \int\limits_{\operatorname{supp}\psi_j} u^*(x, y_i)\,\psi_j(x)\,dS_x\,, \quad i = 1,\dots,N_\phi\,,\ j = 1,\dots,N_\psi\,, \tag{20}$$

$$h_{ij} = \int\limits_{\operatorname{supp}\phi_j} \partial_n u^*(x, y_i)\,\phi_j(x)\,dS_x\,, \quad i,j = 1,\dots,N_\phi\,. \tag{21}$$

The matrices G and H are fully populated and don't possess any structure. Thus, the computational costs when setting up the matrices and the memory consumption are both of order $O(N_\phi N_\psi)$ and $O(N_\phi^2)$, respectively. Each single matrix entry is computed by the use of a combination of analytical and numerical integration.

3 Hierarchical Matrices

The formal definition and description of hierarchical matrices as well as operations involving those matrices can be found in [7, 8]. In this section we give a more intuitive introduction to this topic.

3.1 Motivation

Let $K : [0,1] \times [0,1] \to \mathbb{R}$ be a given function of two scalar variables and $A \in \mathbb{R}^{N\times M}$ a given matrix having the entries

$$a_{k\ell} = K(x_k, y_\ell)\,,\ k = 1,\dots,N\,,\ \ell = 1,\dots,M\,, \tag{22}$$

with $(x_k, y_\ell) \in [0,1] \times [0,1]$. It is obvious, that the asymptotic memory requirement for the matrix A is $\mathrm{Mem}(A) = \mathcal{O}(N\,M)$ and the asymptotic number of arithmetical operations required for the matrix-vector multiplication $\mathrm{Op}(A\,s) = \mathcal{O}(N\,M)$ if $N, M \to \infty$. This quadratic amount is too high already for moderate values of N and M. However, if we agree to store an approximation $\tilde{A}$ of the matrix A and to deal with the product $\tilde{A}\,s$ instead of the exact value $A\,s$ the situation may change. However, then it is necessary to control the error, i.e. to guarantee the inequality

$$\|A - \tilde{A}\|_F \le \varepsilon \|A\|_F\,, \tag{23}$$

where $\|A\|_F$ denotes the Frobenius norm of the matrix A

$$\|A\|_F = \left(\sum_{k,\ell} a_{k\ell}^2 \right)^{1/2} \tag{24}$$

for some prescribed accuracy ε. The best approximation of the matrix A is given by its partial singular value decomposition

$$A \approx \tilde{A} = \tilde{A}(r) = \sum_{i=1}^{r} \sigma_i\, u_i\, v_i^\top \tag{25}$$

where the rank $r = r(\varepsilon)$ is chosen corresponding to the condition

$$\|A - \tilde{A}\|_F^2 \le \sum_{i=r+1}^{\min(N,M)} \sigma_i^2 \le \varepsilon^2 \sum_{i=1}^{\min(N,M)} \sigma_i^2 = \varepsilon^2 \|A\|_F^2. \tag{26}$$

Unfortunately, the complete singular value decomposition of the matrix A requires $\mathcal{O}(N^3)$ arithmetical operations when assuming $N \sim M$, and therefore, is too expensive for practical computations. However, the singular value decomposition can be perfectly used for the illustration of the main ideas.

Example 1. Let us consider the following function on $[0,1] \times [0,1]$

$$K(x,y) = \frac{1}{\alpha + (x-y)^2}\,, \tag{27}$$

where $\alpha > 0$ is a parameter. For $\alpha \sim 1$ the function K is smooth but for small values of α the function K becomes an artificial "strong singularity" at the diagonal $\{(x,x)\}$ of the square $[0,1] \times [0,1]$.

The domain $[0,1] \times [0,1]$ is uniformly discretised using the nodes

$$(x_k, y_\ell) = \Big((k-1)h_x\,,\, (\ell-1)h_y \Big)\,,\ h_x = \frac{1}{N-1}\,,\ h_y = \frac{1}{M-1} \tag{28}$$

for $k = 1, \dots, N$ and $\ell = 1, \dots, M$. In Fig. 2 the logarithmic plot of the singular values of the matrix (22) (i.e. the quantities $\log_{10} \sigma_i\,,\ i = 1, \dots, N$)

for $N = M = 32$ (left plot) and $N = M = 1024$ (right plot) is presented for $\alpha = 1$. It is clear to see that only very few singular values are needed to represent the matrix A in its singular value decomposition (25) for moderate value of the parameter $\varepsilon = 10^{-5} - 10^{-6}$. Almost all singular values are close to the computer zero for $N = M = 1024$. Thus the behaviour of the singular values determines the quality of the low rank approximation (25).

The situation changes if the "singularity" of the function K is more serious. In Fig. 3 (left plot) the rank $r(\varepsilon)$ for $\varepsilon = 10^{-6}$ and $N = M = 256$ is shown as a function of the parameter α. The horizontal axis corresponds to the values $-\log_2(\alpha)$ while α changes from 2^0 till 2^{-8}. However, if we "separate" the variables x and y, i.e. consider only a quarter $[0, 0.5] \times [0.5, 1]$ of the square $[0, 1] \times [0, 1]$ then the situation is better. The right plot in Fig. 3 shows the same curve for separated x and y which is more or less constant now.

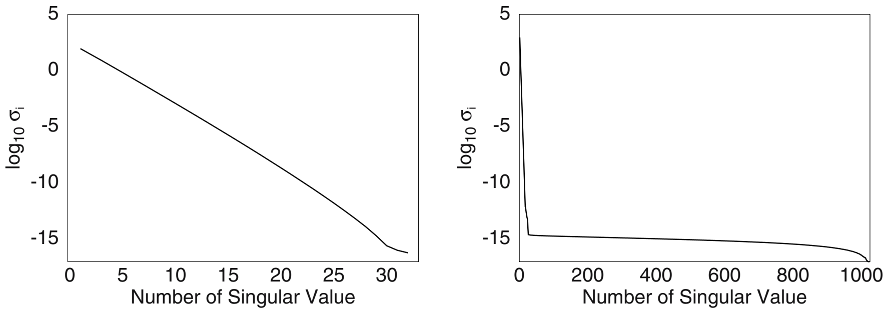

Fig. 2. Distribution of singular values for $N = 32$ (left) and $N = 1024$ (right).

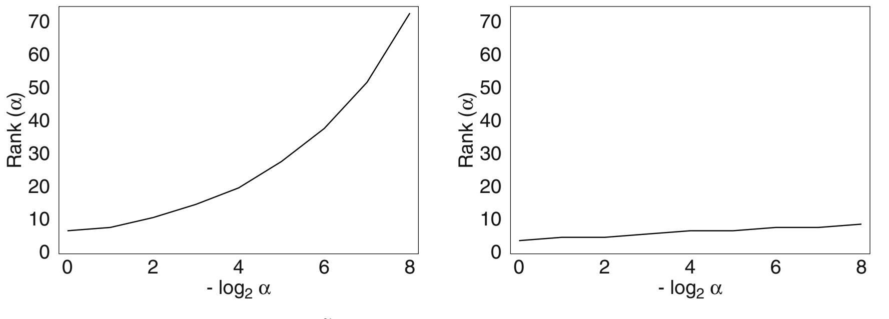

Fig. 3. Rank of the matrix $\tilde{A}$ depending on parameter α for non-separated (left) and separated (right) domains.

Now the main idea of hierarchical methods is very clear. If we decompose the whole matrix A in four blocks corresponding to the domains $[0, 0.5] \times [0, 0.5], [0, 0.5] \times [0.5, 1], [0.5, 1] \times [0, 0.5]$ and $[0.5, 1] \times [0.5, 1]$ we will be able to approximate two of these four blocks efficiently. The two remaining, main diagonal blocks have the same structure as the initial matrix but only the half of the size and their rank will be smaller. In Fig. 4, the left diagram corresponds to the whole matrix and its rank $r(\varepsilon) = 73$ is obtained for $\alpha = 2^{-9}$ and $\varepsilon = 10^{-6}$ for $N = M = 256$. The 2×2 block matrix together with ranks of the blocks is shown in the second diagram of Fig. 4. The approximation of the separated blocks is now acceptable and we continue to decompose only the blocks on the main diagonal. The results can be seen in the third and in the fourth diagram of Fig. 4. The memory requirements for these four matrices is quite different. The first matrix needs $146N$ words of memory, the second $94N$, the third $74N$ and finally we will need $72N$ words of memory for the last block matrix in Fig. 4. Thus a hierarchical decomposition in blocks and their separate approximation using a singular value decomposition leads to a drastic reduction of memory requirements even for this rather small matrix having "diagonal singularity". Note that the rank of the blocks on the main diagonal increases almost linear with the dimension: $12 - 20 - 38 - 73$ while the rank of separated blocks has at most logarithmic growth: $7 - 8 - 9$.

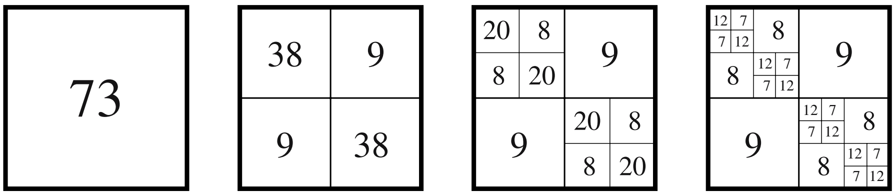

Fig. 4. Initial matrix and its hierarchical decomposition in blocks.

Thus a hierarchical approximation of large dense matrices arising from some generating function having diagonal singularity consist of three steps

- Construction of clusters for variables x and y,
- Finding of possible admissible blocks (i.e. blocks with separated x and y),
- Low rank approximation of admissible blocks.

In the above example the clusters were simply the sets of points x_k which belong to smaller and smaller intervals. The problem is more complicated for three-dimensional irregular point sets. Also the admissible blocks in the above example are very natural. They are just blocks outside of the main diagonal. In the general case we will need some permutations of rows and columns of the matrix to construct such blocks. Finally, the singular value decomposition approximation we have used is not applicable for more realistic examples. We will need more efficient algorithms to approximate admissible blocks. The approximation of the blocks for separated variables x and y in the above example is based on the smoothness of the function K for $x \neq y$. However, if the function K is degenerated, i.e. it is a finite sum of products of functions depending only on x and y

$$K(x,y) = \sum_{i=1}^{r} p_i(x) q_i(y) \tag{29}$$

then the rank of the matrix A defined in (22) is equal to r independent of its dimension. Thus for $N, M \gg r$ the matrix A is a low rank matrix. This property is independent of the smoothness of the functions p_i, q_i in (29). The low rank representation of the matrix A is now

$$A = \sum_{i=1}^{r} u_i v_i^\top \,, \tag{30}$$

with

$$(u_i)_k = p_i(x_k)\,,\ (v_i)_\ell = q_i(y_\ell) \tag{31}$$

for $k = 1, \ldots, N$ and $\ell = 1, \ldots, M$. Note that this representation is not the singular value decomposition (25). If the function is smooth enough then we can use its Taylor series with respect to the variable x in some point x^*

$$K(x,y) = \sum_{i=1}^{r} \frac{1}{i\,!} \frac{\partial^i K(x^*,y)}{\partial\, x^i} (x - x^*)^i + R_r(x,y) \tag{32}$$

to obtain a degenerated approximation

$$A \approx \tilde{A} = \sum_{i=1}^{r} u_i v_i^\top \,, \tag{33}$$

with

$$(u_i)_k = (x_k - x^*)^i\,,\ (v_i)_\ell = \frac{1}{i\,!} \frac{\partial^i K(x^*,y_\ell)}{\partial\, x^i} \tag{34}$$

for $k = 1, \ldots, N$ and $\ell = 1, \ldots, M$. Note again that (33) is not the singular value decomposition of the matrix $\tilde{A}$. If the remainder R_r is uniformly bounded by the original function K

$$\Big| R_r(x,y) \Big| \leq \varepsilon \Big| K(x,y) \Big| \tag{35}$$

for all x and y with some $r = r(\varepsilon)$ then we can guarantee the accuracy of the low rank matrix approximation

$$\|A - \tilde{A}\| \leq \varepsilon \|A\|_F \tag{36}$$

for all dimensions N and M. The rank $r = r(\varepsilon)$ of the matrix $\tilde{A}$ is also independent of its dimension. Thus, for $N \approx M$ the matrix $\tilde{A}$ requires only $\mathrm{Mem}(\tilde{A}) = \mathcal{O}(N)$ words of computer memory. However, an efficient construction of the Taylor series for a given function in three-dimensional case is practically impossible. Thus it is rather an illustration for the fact that there exist low rank decompositions which are not based on the singular value decomposition. A further example of low rank approximation of the given function is a decomposition of the fundamental solution of the Laplace operator

$$u^*(x,y) = \frac{1}{4\,\pi}\frac{1}{|x-y|} \quad \text{for } x,y \in \mathbb{R}^3$$

in spherical harmonics which is used by multipole methods (see [6]).

3.2 Hierarchical Clustering

To find a suitable permutation, a cluster tree is constructed by recursively partitioning some weighted characteristic points

$$\Big\{(x_k, g_k)\,,\ k = 1,\ldots,N\Big\} \subset \mathbb{R}^3 \times \mathbb{R}_+ \tag{37}$$

and

$$\Big\{(y_\ell, q_\ell)\,,\ \ell = 1,\ldots,M\Big\} \subset \mathbb{R}^3 \times \mathbb{R}_+ \tag{38}$$

in order to separate the variables x and y. A large distance between two characteristic points results in a large difference of the respective equation numbers. While dealing with boundary element matrices the characteristic points can be the collocation points and the weights the areas of the supports of the trial functions. A given cluster

$$Cl = \Big\{(x_k, g_k)\,,\ k = 1,\ldots,n\Big\}$$

with $n > 1$ can be separated in two sons using the following algorithm.

Algorithm 1

```
1. Mass of the cluster
```

$$G = \sum_{k=1}^{n} g_k \in \mathbb{R}_+\,,$$

```
2. Centre of the cluster
```

$$X = \frac{1}{G}\sum_{k=1}^{n} g_k\, x_k \in \mathbb{R}^3$$

```
3. Covariance matrix of the cluster
```

$$C = \sum_{k=1}^{n} g_k \, (x_k - X) \, (x_k - X)^\top \in \mathbb{R}^{3\times 3} \, ,$$

```
4. Eigenvalues and eigenvectors
```

$$C \, v_i = \lambda_i \, v_i \, , \quad i = 1, 2, 3 \, , \quad \lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0 \, ,$$

```
5. separation
  5.1 initialisation
```

$$Cl_1 := \oslash \, , \quad Cl_2 := \oslash \, ,$$

```
  5.2 for $k = 1, \ldots, n$
        if $(x_k - X, v_1) \geq 0$ then $Cl_1 := Cl_1 \cup (x_k, g_k)$
                                   else $Cl_2 := Cl_2 \cup (x_k, g_k)$ .
```

The eigenvector v_1 of the matrix C corresponds to the largest eigenvalue of this matrix and shows in the direction of the longest extension of the cluster. The separation plane $\Big\{ x \in \mathbb{R}^3 \, : \, (x - X, v_1) = 0 \Big\}$ goes through the centre X of the cluster and is orthogonal to the eigenvector v_1. Thus, Algorithm 1 divides a given arbitrary cluster of weighted points in two more or less equal sons. In Fig. 5 the first two levels of separation of a simplified model of an exhaust manifold are shown. The separation of a given cluster in two sons defines a permutation of the points in the cluster. The points in the first son will be numbered first and then in the second son. Algorithm 1 will be applied recursively to the sons until they contain less than or equal to some prescribed (small and independent of N) number $n_{\min}$ of points. Next, cluster pairs which are geometrically well separated are identified. They will be

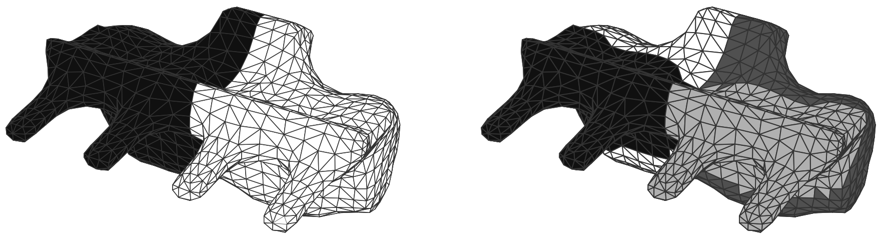
Fig. 5. Clusters of the first two levels.

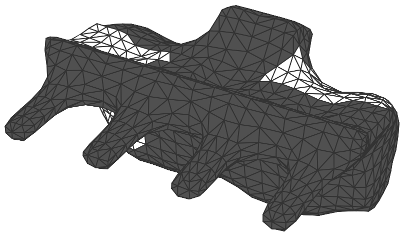

Fig. 6. An admissible cluster pair.

regarded as admissible cluster pairs, e.g. the clusters in Fig. 6. An appropriate admissibility criterion is the following simple geometrical condition. A pair of clusters (Cl_x, Cl_y) with $n_x > n_{\min}$ and $m_y > n_{\min}$ elements is admissible if

$$\min\Big(\operatorname{diam}(Cl_x), \operatorname{diam}(Cl_y)\Big) \le \eta \operatorname{dist}(Cl_x, Cl_y)\,, \tag{39}$$

where $0 < \eta < 1$ is a given parameter. Although the criterion (39) is quite simple a rather large computational effort (quadratic with respect to the number of elements in the clusters Cl_x and Cl_y) is required for calculating the exact values

$$\begin{aligned}
\operatorname{diam}(Cl_x) &= \max_{k_1,k_2} |x_{k_1} - x_{k_2}|\,,\\
\operatorname{diam}(Cl_y) &= \max_{\ell_1,\ell_2} |y_{\ell_1} - y_{\ell_2}|\,,\\
\operatorname{dist}(Cl_x, Cl_y) &= \min_{k,\ell} |x_k - y_\ell|\,.
\end{aligned}$$

In practice we use more rough and more restrictive but easily computable bounds

$$\begin{aligned}
\operatorname{diam}(Cl_x) &\le 2\,\max_{k} |X - x_k|\,,\\
\operatorname{diam}(Cl_y) &\le 2\,\max_{\ell} |Y - y_\ell|\,,\\
\operatorname{dist}(Cl_x, Cl_y) &\ge |X - Y| - \frac{1}{2}\Big(\operatorname{diam}(Cl_x) + \operatorname{diam}(Cl_y)\Big)\,,
\end{aligned}$$

where X and Y are the already computed centres (cf. Algorithm 1) of the clusters Cl_x and Cl_y, for the admissibility condition. If a cluster pair is not admissible and $n_x > n_{\min}$ and $m_y > n_{\min}$ then there exist sons of the both clusters

$$Cl_x = Cl_{x,1} \cup Cl_{x,2}\,,\ Cl_y = Cl_{y,1} \cup Cl_{y,2}\,.$$

Let us assume for simplicity that the cluster Cl_x is bigger: $\operatorname{diam}(Cl_x) \ge \operatorname{diam}(Cl_y)$. In this case we check two new pairs

$$\left(Cl_{x,1}, Cl_y\right), \ \left(Cl_{x,2}, Cl_y\right)$$

for admissibility and so on. This recursive procedure stops if $n_x \leq n_{\min}$ or $m_y \leq n_{\min}$. The corresponding block of the matrix is small and will be computed exactly. The cluster trees for the variables x and y together with the set of admissible cluster pairs as well as of small cluster pairs allow to split the matrix into a collection of blocks of various sizes. The block structure of the Galerkin matrix for the single layer potential on the surface form Figs. 5–6 is shown in Fig. 7. The colour of the blocks indicates the "quality" of the approximation. The light grey colour corresponds to well approximated blocks while dark grey and especially black colour indicates less good approximation or even exact computation. Thus the main problem remains is how to approximate the big blocks without using the singular value decomposition. The corresponding procedures will be described in the forthcoming section.

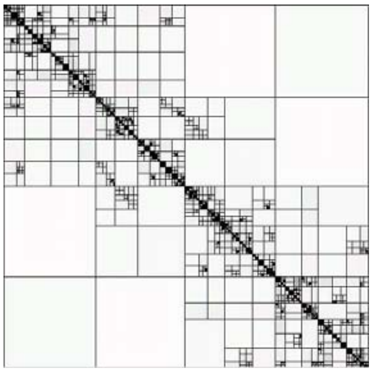

Fig. 7. Matrix decomposition.

3.3 Adaptive Cross Approximation

On the matrix level the fully pivoted ACA algorithm can be written in the following form:

Algorithm 2

```
1. Initialisation
```

$$R_0 = A\,, \ \ S_0 = 0\,.$$

```
2. For $i = 0, 1, 2, \ldots$ compute
  2.1. pivot element
```

$$(k_{i+1}, \ell_{i+1}) = \operatorname{ArgMax} |(R_i)_{k\ell}| \ ,$$

```
  2.2. normalising constant
```

$$\gamma_{i+1} = \left((R_i)_{k_{i+1}\ell_{i+1}}\right)^{-1} ,$$

```
  2.3. new vectors
```

$$u_{i+1} = \gamma_{i+1} R_i e_{\ell_{i+1}} , \;\; v_{i+1} = R_i^\top e_{k_{i+1}},$$

```
  2.4. new residuum
```

$$R_{i+1} = R_i - u_{i+1} v_{i+1}^\top ,$$

```
  2.5. new approximation
```

$$S_{i+1} = S_i + u_{i+1} v_{i+1}^\top .$$

The whole residuum matrix R_i is inspected in `Step 2.1` of Algorithm 2 for its maximal entry. Thus the appropriate stopping criterion for a given $\varepsilon > 0$ at step r is

$$\|R_r\|_F \le \varepsilon \|A\|_F .$$

Note that the crosses built from the column-row pairs with the indices k_i, ℓ_i for $i = 1, \ldots, r$ will be computed exactly, while all other elements are approximated. The number of operations required to generate the approximation $\tilde{A} = S_r$ is $\mathcal{O}(r^2\, N\, M)$. The memory requirement for Algorithm 2 is $\mathcal{O}(N\, M)$ since the whole matrix A is assumed to be given at the beginning. Thus, Algorithm 2 is much faster than a singular value decomposition but still rather expensive for large matrices. If the matrix A has not yet been generated but there is a possibility of generating its entries $a_{k\ell}$ individually then the following partially pivoted ACA algorithm can be used for the approximation.

Algorithm 3

```
1. Initialisation
```

$$S_0 = 0 , \;\; \mathcal{I} = \oslash , \;\; c = 0 \in \mathbb{R}^N ,$$

```
2. Recursion
  2.1. Choice of the next not yet generated row
```

$$k_{i+1} = \min\{k \, : \, k \notin \mathcal{I}\} , \;\; \mathcal{I} = \mathcal{I} \cup \{k_{i+1}\} ,$$

```
       or stop if all rows are generated, i.e. $\mathcal{I} = \{1, \ldots, N\}$ ,
  2.2. Generation of the row
```

$$a = A^T e_{k_{i+1}} ,$$

```
 2.3. Row of the residuum and the pivot column
```

$$R_i^\top e_{k_{i+1}} = a - \sum_{m=1}^{i} (u_m)_{k_{i+1}} v_m \, ,$$
$$\ell_{i+1} = \mathrm{ArgMax} \left|(R_i)_{k_{i+1}\ell}\right| \, ,$$

```
 2.4. Test
```

$$\texttt{if Max} \left|(R_i)_{k_{i+1}\ell}\right| = 0 \texttt{ then goto 2.1.}$$

```
 2.5. Normalising constant
```

$$\gamma_{i+1} = \left((R_i)_{k_{i+1}\ell_{i+1}}\right)^{-1} \, ,$$

```
 2.6. Generation of the column, Update of the control vector
```

$$a = A\, e_{\ell_{i+1}} \, , \;\; c = c + |a| \, ,$$

```
 2.7. Column of the residuum and the pivot row
```

$$R_i e_{\ell_{i+1}} = a - \sum_{m=1}^{i} (v_m)_{\ell_{i+1}} u_m \, ,$$
$$k_{i+2} = \mathrm{ArgMax} \left|(R_i)_{k\ell_{k+1}}\right| \, ,$$

```
 2.8. New vectors
```

$$u_{i+1} = \gamma_{i+1} R_i e_{\ell_{i+1}} \, , \;\; v_{i+1} = R_i^\top e_{k_{i+1}} ,$$

```
 2.9. New approximation
```

$$S_{i+1} = S_i + u_{i+1} v_{i+1}^\top \, .$$

```
2.10. Recursion
```

$$i := i + 1 \, , \;\; \texttt{goto 2.2}$$

Since the matrix A will not be generated completely we can use the norm of its approximant S_i to define a stopping criterion. This norm can be computed recursively as follows,

$$\|S_{i+1}\|_F^2 = \|S_i\|_F^2 + 2 \sum_{m=1}^{i} u_{i+1}^\top u_m \, v_m^\top v_{i+1} + \|u_{i+1}\|_F^2 \|v_{i+1}\|_F^2 \, . \qquad (40)$$

An appropriate stopping criterion in `Step 2.8` is then

$$\|u_r\|_F \|v_r\|_F \le \varepsilon \|S_r\|_F \, . \qquad (41)$$

However, since the whole matrix A will not be generated while using partially pivoted ACA algorithm, it is necessary to check the control vector c updated after every column generation for zero components in not yet generated rows. If there is some index $i^* \notin \mathcal{I}$ with $c_{i^*} = 0$ then the row i^* has not yet contributed to the matrix. It can happen that this row contains relevant information and, therefore, we have to set $i := i+1, k_{i+1} = i^*$ and to restart the algorithm in `Step 2.2`. With this trivial modification Alg. 3 can be used not only for dense matrices but also for reducible and even for sparse matrices.

Algorithm 3 requires only $\mathcal{O}(r^2(N+M))$ arithmetical operations and its memory requirement is $\mathcal{O}(r(N+M))$. Thus this algorithm is perfect for large matrices. Using the theory of polynomial multidimensional interpolation the following result was proven in [2].

Theorem 4. *Let the function $K(x,y)$ be asymptotically smooth with respect to y, i.e. $K(x,\cdot) \in C^\infty(\mathbb{R}^3 \backslash \{x\})$ for all $x \in \mathbb{R}^3$, satisfying*

$$|\partial_y^\alpha K(x,y)| \le c_p \, |x-y|^{g-p} \,, \; p = |\alpha| \tag{42}$$

for all multiindices $\alpha \in \mathbb{N}_0^3$ with a constant $g < 0$. Moreover, the matrix $A \in \mathbb{R}^{N \times M}$ is decomposed in blocks corresponding to the admissibility condition

$$\operatorname{diam}(Cl_y) \le \eta \operatorname{dist}(Cl_x, Cl_y) \,, \; \eta < 1 \,. \tag{43}$$

Then the matrix A with $M \sim N$ can be approximated up to an arbitrary given accuracy $\varepsilon > 0$ using a system of given points $(\tilde{x}_k, \tilde{y}_\ell)$,

$$\|A - \tilde{A}\|_F \le \varepsilon \|A\|_F, \tag{44}$$

and

$$\operatorname{Op}(\tilde{A}) = \operatorname{Op}(\tilde{A}\, s) = \operatorname{Mem}(\tilde{A}) = \mathcal{O}(N^{1+\delta}\varepsilon^{-\delta}) \quad \textit{for all}\, \delta > 0 \,. \tag{45}$$

4 Exploitation of Symmetry

The exploitation of symmetry is another possibility to reduce computational costs and has been presented in [1, 4, 5] using linear representation theory for finite groups. The aim is a decomposition of function spaces into orthogonal subspaces of symmetric functions, such that each subproblem is defined on a so called symmetry cell. The global solution can then be reconstructed from these components. In the following we will give an overview of exploiting symmetry in the BEM. The considered procedure can easily be extended to a vector case, e.g. a magnetostatic or eddy current problem as shown in the numerical results section.

4.1 Algebraic Description

A (complete) *geometrical symmetry* of the domain Ω_{BEM} is given if there exists a finite group $\mathcal{Q}$ of isometries of $\mathbb{R}^3$, such that Ω_{BEM} is invariant w.r.t. $\mathcal{Q}$. For each element of the symmetry group $\mathcal{Q}$ there is an orthogonal matrix $Q \in \mathbb{R}^3$ (i.e. $QQ^T = Q^TQ = I$) and a symmetry point $x_0 \in \mathbb{R}^3$ such that

$$x' = x_0 + Q(x - x_0) \in \Omega_{\mathrm{BEM}}\,,\ \forall\, x \in \Omega_{\mathrm{BEM}}\,. \tag{46}$$

For the sake of simplicity we assume in the sequel that $x_0 = 0$. The geometrical symmetry of the domain Ω_{BEM} implies that geometrical symmetry also holds for its boundary Γ, i.e. the symmetry mapping Q fulfills

$$x' = Qx \in \Gamma\,,\ \forall\, x \in \Gamma\,. \tag{47}$$

For smooth boundary Γ the condition (47) implies the following connection of unit normal vectors to Γ at x and $x' = Qx$

$$Qn_x = n_{Qx} = n_{x'}\,,\ \forall\, x \in \Gamma\,. \tag{48}$$

For a symmetric problem only a part, the so called *symmetry cell* needs to be discretised and considered. The symmetry cell is the smallest subdomain which generates the entire domain under the action of the symmetry group. Let Γ_h be the discretisation of the symmetry cell of the boundary Γ. An entire boundary mesh can then be obtained by $m-1$ consecutive applications of Q on Γ_h as shown in Fig. 8.

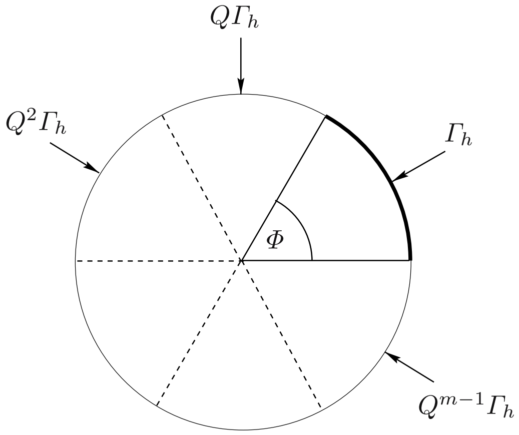

Fig. 8. Discretisation symmetry.

In the following we consider the most simple case when the system matrix A can be renumbered and partitioned into $m \times m$ blocks structure having blocks of size exactly $n = N/m$. This is the case for a piecewise constant discretisation scheme, where N is the number of unknowns, m is the size of the symmetry group and n is the number of unknowns in each symmetry cell.

All considerations can be extended to a more general discretisation scheme (17)-(18) with $N_\phi \neq N_\psi$.

The system of boundary elements, collocation points and the ansatz functions features *discretisation symmetry* if there exists a permutation σ such that the index set $\{1,\dots,N\}$ of all degrees of freedom can be written as

$$\{1,\dots,n,\sigma(1),\dots,\sigma(n),\dots,\sigma^{m-1}(1),\dots,\sigma^{m-1}(n)\} \tag{49}$$

with $\sigma^m(i) = i\,,\ \forall i = 1,\dots,n$, and, additionally, for collocation points and ansatz functions holds

$$Q(\operatorname{supp}\phi_j) = \operatorname{supp}\phi_{\sigma(j)}\ ,\ j = 1,\dots,N\,, \tag{50}$$

$$\phi_j(x) = \phi_{\sigma(j)}(Qx)\,,\ \forall x \in \operatorname{supp}\phi_j\ ,\ j = 1,\dots,N\,, \tag{51}$$

$$Qy_i = y_{\sigma(i)}\ ,\ i = 1,\dots,N\,. \tag{52}$$

The permutation σ offers the possibility for renumbering unknowns corresponding to the symmetry of the problem. For the general case $N_\phi \neq N_\psi$ two different permutations σ_ϕ and σ_ψ of the index sets $\{1,\dots,N_\phi\}$ and $\{1,\dots,N_\psi\}$, respectively, have to be introduced.

The problem features the *symmetry of the kernel* if the following condition does hold for the kernel K

$$K(Q^k x, Q^l y) = K(x, Q^{l-k} y)\,,\ \forall x,y \in \Gamma\,,\ \forall k,l \in \mathbb{Z}\,. \tag{53}$$

Especially, for $k = l$ we obtain $K(Q^k x, Q^k y) = K(x,y)\,,\ \forall k \in \mathbb{Z}$. Note that the BEM matrices in (20)-(21) are both generated by symmetrical kernels.

Lemma 1. *The symmetrical BEM discretisation* (50)–(52) *of the geometrically symmetrical problem* (48) *having kernel symmetry* (53) *leads, after numbering of unknowns corresponding to* (49)*, to the following property of the matrix entries:*

$$a_{ij} = a_{\sigma(i)\sigma(j)}\ ,\ \forall\, i,j\,. \tag{54}$$

Proof. Definition of the matrix entries leads after substitution (47) to

$$\begin{aligned} a_{ij} &= \int\limits_{\operatorname{supp}\phi_j} K(x,y_i)\,\phi_j(x)\,dS_x \\ &= \int\limits_{\operatorname{supp}\phi_j} K(Qx,Qy_i)\phi_{\sigma(j)}(Qx)\,dS_x \\ &= \int\limits_{Q(\operatorname{supp}\phi_j)} K(x',Qy_i)\phi_{\sigma(j)}(x')\,dS_{x'} \\ &= \int\limits_{\operatorname{supp}\phi_{\sigma(j)}} K(x,y_{\sigma(i)})\phi_{\sigma(j)}(x')\,dS_{x'} \ = \ a_{\sigma(i)\sigma(j)}\,, \end{aligned} \tag{55}$$

where the properties (48)–(52) have been used. □

Since $\sigma^{m-k}(\sigma^k(i)) = i$ for all i the property (54) implies

$$a_{\sigma^k(i)\,j} = a_{i\,\sigma^{m-k}(j)}\;,\;\forall\, i,j\,.$$

Thus the system of linear equations of the symmetrical BEM takes the following block-circulant form

$$\begin{pmatrix} A_1 & A_2 & \dots & A_m \\ A_m & A_1 & \dots & A_{m-1} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ A_2 & A_3 & \dots & A_1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \dots \\ \dots \\ u_m \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \dots \\ \dots \\ b_m \end{pmatrix}. \tag{56}$$

Thus only the basis matrices $A_1, A_2, \dots, A_m$ should be generated and stored. The amount of numerical work and of memory will therefore be reduced from N^2 to N^2/m. This factor can be very useful for practical computations. The numerical solution of the system of linear equations having a block-circulant matrix can also be implemented much more efficiently than a straightforward direct elimination method which would lead to $O(N^3)$ arithmetical operations [18]. The main property of the circulant matrices

$$A = \begin{pmatrix} a_1 & a_2 & a_3 & \dots & a_{m-1} & a_m \\ a_m & a_1 & a_2 & a_3 & \dots & a_{m-1} \\ a_{m-1} & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & a_2 \\ a_2 & a_3 & \dots & a_{m-1} & a_m & a_1 \end{pmatrix} \in \mathbb{C}^{m\times m}$$

is that all of them are simultaneously diagonalised by the matrix of the discrete Fourier transform F_m :

$$\begin{aligned} A &= \frac{1}{m} F_m\, \Lambda\, F_m^*\,, \\ f_{k,l} &= \omega_m^{(k-1)(l-1)} = e^{\imath \frac{2\pi}{m}(k-1)(l-1)}\,. \end{aligned} \tag{57}$$

The most simple nontrivial circulant matrix

$$J = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \\ \dots & \dots & \dots & \dots & \\ 0 & 0 & 0 & \dots & 1 \\ 1 & 0 & 0 & \dots & 0 \end{pmatrix}$$

has the following eigenvalues

$$\Lambda = \operatorname{diag}\left(\omega_m^{l-1}\,,\; l = 1,\dots,m\right)\,.$$

Using the Kronecker product $\otimes$ of matrices we rewrite the block-circulant matrix A of the system (56) in the form (cf. (57))

$$A = \sum_{k=1}^{m} J^{k-1} \otimes A_k = \frac{1}{m} \sum_{k=1}^{m} \left(F_m \, \Lambda^{k-1} \, F_m^* \right) \otimes A_k \,,$$

where the dimension of the matrices A_k is now $n = N/m$. Since

$$F_m \, F_m^* = F_m^* \, F_m = I_m$$

and using the known property of the Kronecker product

$$(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$$

we obtain

$$\begin{aligned}(F_m^* \otimes I_n) \, A \, (F_m \otimes I_n) &= \sum_{k=1}^{m} (F_m^* \otimes I_n) \, (J^{k-1} \otimes A_k) \, (F_m \otimes I_n) \\ &= \sum_{k=1}^{m} (F_m^* J^{k-1} F_m) \otimes (I_n A_k I_n) \;=\; \frac{1}{m} \sum_{k=1}^{m} \Lambda^{k-1} \otimes A_k \,.\end{aligned}$$

The system (56) can now be rewritten in the block-diagonal form

$$\begin{pmatrix} D_1 & 0 & \dots & 0 \\ 0 & D_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & D_m \end{pmatrix} \begin{pmatrix} \tilde{u}_1 \\ \tilde{u}_2 \\ \dots \\ \dots \\ \tilde{u}_m \end{pmatrix} = \begin{pmatrix} \tilde{b}_1 \\ \tilde{b}_2 \\ \dots \\ \dots \\ \tilde{b}_m \end{pmatrix} , \tag{58}$$

where

$$D_l = \frac{1}{m} \sum_{k=1}^{m} \omega_m^{(l-1)(k-1)} A_k \in \mathbb{C}^{n \times n} \tag{59}$$

and

$$\tilde{u} = (F_m^* \otimes I_n) u \,, \; \tilde{b} = (F_m^* \otimes I_n) b \,.$$

Thus the following algorithm has been derived (similar to proposed in [1])

1. *Compute all basis matrices* $A_k \,, \; k = 1, \dots, m$
2. *Compute*

$$\tilde{b} = (F_m^* \otimes I_n) b$$

   using n *Fast Fourier Transforms (FFT).*
3. *For* $l = 1, \dots, m$

3.1 Generate the matrix

$$D_l = \frac{1}{m}\sum_{k=1}^{m} \omega_m^{(l-1)(k-1)} A_k$$

3.2 Solve the system

$$D_l \tilde{u}_l = \tilde{b}_l$$

4. Compute

$$u = (F_m \otimes I_n)\tilde{u}$$

using n FFT's.

The straightforward implementation of this algorithm leads to $O(mn^2)$ operations and memory units in `Step 1.`, $O(nm\log(m))$ operations in `Step 2.`, $O(mn^2)$ operations and memory units in `Step 3.1`, $O(mn^3)$ operations for solving all systems in `Step 3.2` and finally $O(nm\log(m))$ operations in the last `Step 4.`. Thus `Step 3.2` is the most expensive and defines the final amount of numerical work for the whole algorithm $O(mn^3) = O(N^3/m^2)$. This amount remains of the same capital order of $O(N^3)$, but it is reduced by a remarkable factor m^2.

4.2 Symmetry of Excitation

As described in Section 2.1 in the BE domain the equation

$$\Delta\phi = -\frac{1}{\varepsilon_0}\rho$$

is to be solved, where ϕ is the electric scalar potential and ρ is the electric charge density. Discretisation by nodal ansatz functions and point collocation leads in case of symmetry to the equation system of the form (56). Electromagnetic devices often possess the *symmetry of excitation*, which means for the electrostatic case, that the symmetry mappings Q fulfill

$$\rho(Q^k x) = \alpha_{k+1}\rho(x)\,,\ \forall x \in \Gamma\,,\ k = 0,\dots,m-1\,, \tag{60}$$

for some $\alpha_k \in \mathbb{R}$. In case of an excitation symmetry we don't perform the Fourier transform as described in the previous section but simplify the equation system (56) in a different way. As a consequence from (60) we obtain a linear dependency of the components of the r.h.s in (56)

$$b_k \ = \ \alpha_k b_1\,,\ k = 1,\dots,m\,. \tag{61}$$

Additionally, we require the following condition to be fulfilled

$$\frac{\alpha_1}{\alpha_2} = \frac{\alpha_2}{\alpha_3} = \ldots = \frac{\alpha_{m-1}}{\alpha_m} = \frac{\alpha_m}{\alpha_1}\,. \tag{62}$$

Thus the equation system can be reduced to one subsystem

$$(\alpha_1 A_1 + \alpha_2 A_2 + \cdots + \alpha_m A_m)\, u_1 \;=\; b_1 \tag{63}$$

of dimension N/m, where N is the total number of unknowns. The remaining solution components can be computed by

$$u_k \;=\; \alpha_k u_1\,,\;\; k = 1,\ldots,m\,.$$

Thus the exploitation of the excitation symmetry leads to reduction of computational costs from N^2 to N^2/m^2.

5 Numerical Experiments

5.1 Asymptotic Behaviour

We start the numerical studies considering the most simple smooth surface $\Gamma = \partial\Omega$ for $\Omega \subset \mathbb{R}^3$, namely the surface of the unit sphere,

$$\Gamma = \Big\{ x \in \mathbb{R}^3 \,:\, |x| = 1 \Big\}\,. \tag{64}$$

As an appropriate discretisation of Γ we consider the icosahedron that is uniformly triangulated before being projected onto the circumscribed unit sphere. On this way we obtain a sequence $\{\Gamma_N\}$ of almost uniform meshes on the unit sphere which are shown in Fig. 9 for different numbers of boundary elements N. This sequence allows to study the convergence of boundary element methods for different examples. In Fig. 10 the clusters of the levels 1 and 2 obtained

Fig. 9. Discretisation of the unit sphere with $N = 320$ and $N = 1280$.

Fig. 10. Clusters of the level 1 and 2 for $N = 1280$.

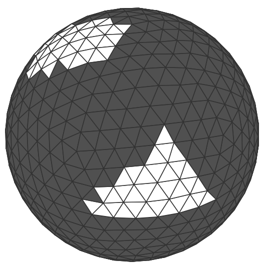

Fig. 11. An admissible cluster pair for $N = 1280$.

with Alg. 1 for $N = 1280$ are presented. In Fig. 11 a typical admissible cluster pair is shown. We solve the interior Dirichlet boundary value problems for the Laplace equation using a Galerkin boundary element method. The piecewise linear basis functions will be used for approximation of the Dirichlet datum and piecewise constant basis functions for approximation of the Neumann datum. We will use the L_2 projection for the approximation of the given part of the Cauchy data. The boundary element matrices G and H are generated in approximative form using the partially pivoted ACA algorithm with a variable relative accuracy ε_1 depending on the expected discretisation error. The resulting systems of linear equations are solved using some variants of the Conjugate Gradient Method (CGM) with or without preconditioning up to a relative accuracy $\varepsilon_2 = 10^{-8}$. The analytical solution is a harmonic function

$$\phi(x) = (1 + x_1)\exp(2\pi\, x_2)\cos(2\pi\, x_3)\,. \tag{65}$$

The results of the computations are shown in Tables 1 and 2. The number of boundary elements is listed in the first column of these tables. The second column contains the number of nodes while in the third column of Table 1 the prescribed accuracy for the ACA algorithm for approximation of both matrices $H \in \mathbb{R}^{N\times M}$ and $G \in \mathbb{R}^{N\times N}$ is given. The fourth column of this table shows the memory requirements in MByte for the approximate double

Table 1. ACA approximation of the matrices H and G, Dirichlet problem.

N	M	ε_1	MByte(H)	%	MByte(G)	%
80	42	$1.0 \cdot 10^{-2}$	0.03	97.8	0.02	48.7
320	162	$1.0 \cdot 10^{-3}$	0.26	65.6	0.21	27.2
1280	642	$1.0 \cdot 10^{-4}$	2.45	39.1	1.94	15.5
5120	2562	$1.0 \cdot 10^{-5}$	20.05	20.0	15.72	7.9
20480	10242	$1.0 \cdot 10^{-6}$	149.19	9.3	115.83	3.6
81920	40962	$1.0 \cdot 10^{-7}$	1085.0	4.2	837.50	1.6

layer potential matrix H. The quality of this approximation in percentage of the original matrix is listed in the next column. The corresponding values for the single layer potential matrix G can be seen in the columns six and seven. The partitioning of the matrix for $N = 5120$ as well as the quality of the approximation of single blocks is shown in Fig. 12. The left diagram in Fig. 12 shows the symmetric single layer potential matrix G while the rectangular double layer potential matrix H is depicted in the right diagram. The legend indicates the percentage of memory needed for the ACA approximation of the blocks compared to the full memory. Further numerical results are shown in Table 2. The third column shows the number of Conjugate Gradient iterations needed to reach the prescribed accuracy ε_2. The relative L_2-error for the Neumann datum ψ

$$Error_1 = \frac{\|\psi - \tilde{\psi}\|_{L_2(\Gamma)}}{\|\psi\|_{L_2(\Gamma)}}, \tag{66}$$

where $\tilde{\psi}$ denotes the numerical solution, is given in the fourth column. The next column represents the rate of convergence for the Neumann datum, i.e.

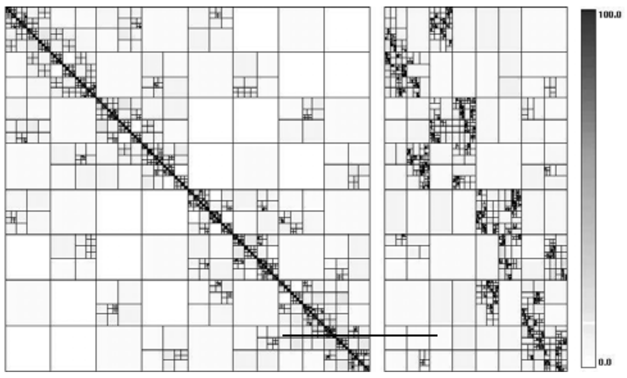

Fig. 12. Partitioning of the BEM matrices for $N = 5120$ and $M = 2562$.

Table 2. Accuracy of the Galerkin method, Dirichlet problem.

N	M	$Iter$	$Error_1$	CF_1	$Error_2$	CF_2
80	42	22	$9.34 \cdot 10^{-1}$	–	$7.29 \cdot 10^{-0}$	–
320	162	32	$5.06 \cdot 10^{-1}$	1.85	$3.29 \cdot 10^{-1}$	22.16
1280	642	45	$2.23 \cdot 10^{-1}$	2.27	$3.53 \cdot 10^{-2}$	9.32
5120	2562	56	$1.04 \cdot 10^{-1}$	2.14	$3.54 \cdot 10^{-3}$	9.97
20480	10242	72	$5.11 \cdot 10^{-2}$	2.03	$4.11 \cdot 10^{-4}$	8.61
81920	40962	94	$2.53 \cdot 10^{-2}$	2.02	$4.30 \cdot 10^{-5}$	9.56

the quotient between the errors in two consecutive lines of column four. Finally, the last two columns show the absolute error in a prescribed inner point $x^* \in \Omega$,

$$Error_2 = |\phi(x^*) - \tilde{\phi}(x^*)| \,, \; x^* = (0.250685, 0.417808, 0.584932)^\top \tag{67}$$

for the value $\tilde{\phi}(x^*)$ obtained using an approximate representation formula. Table 2 obviously shows a linear convergence $\mathcal{O}(N^{-1/2}) = \mathcal{O}(h)$ of the Galerkin boundary element method for the Neumann datum in the L_2 norm. It should be noted that this theoretically guaranteed convergence order can already be observed when approximating the matrices H and G with much less accuracy as it was used to obtain the results in Table 1. However, this high accuracy is necessary in order to be able to observe the third order (or even better) pointwise convergence rate within the domain Ω presented in the last two columns of Table 2. Especially for $N = 81920$ a very high accuracy of $\varepsilon_1 = 1.0 \cdot 10^{-7}$ of the ACA approximation is necessary.

5.2 Examples with Symmetries

For numerical tests we consider TEAM workshop problem 10 [14] (TEAM= Testing Electromagnetic Analysis Methods). An exciting coil is set between two steel channels, and a steel plate is inserted between the channels. The geometry is symmetrical with respect to all three coordinate planes. In order to examine the behaviour of the ACA algorithm and the full BEM method when exploiting symmetries, we consider along with the full model three further meshes exploiting one, two and all three symmetries respectively (Fig. 13). Additionally, for each mesh of this mesh sequence we gradually perform two refinements to show the linear behaviour of the ACA algorithm with respect to the problem size. Thus we obtain three mesh sequences with altogether 12 meshes. Hexahedral second order FEM elements (20 nodes) are used in connection with rectangular second order BEM elements (8 nodes) for both Dirichlet and Neumann data.

In the case when there are some fixed collocation points (e.g. points on a symmetry face in case of a mirror symmetry), the size of each subblock in (56) is close to, but not exactly equal to, $n = N/m$ and the matrix blocks

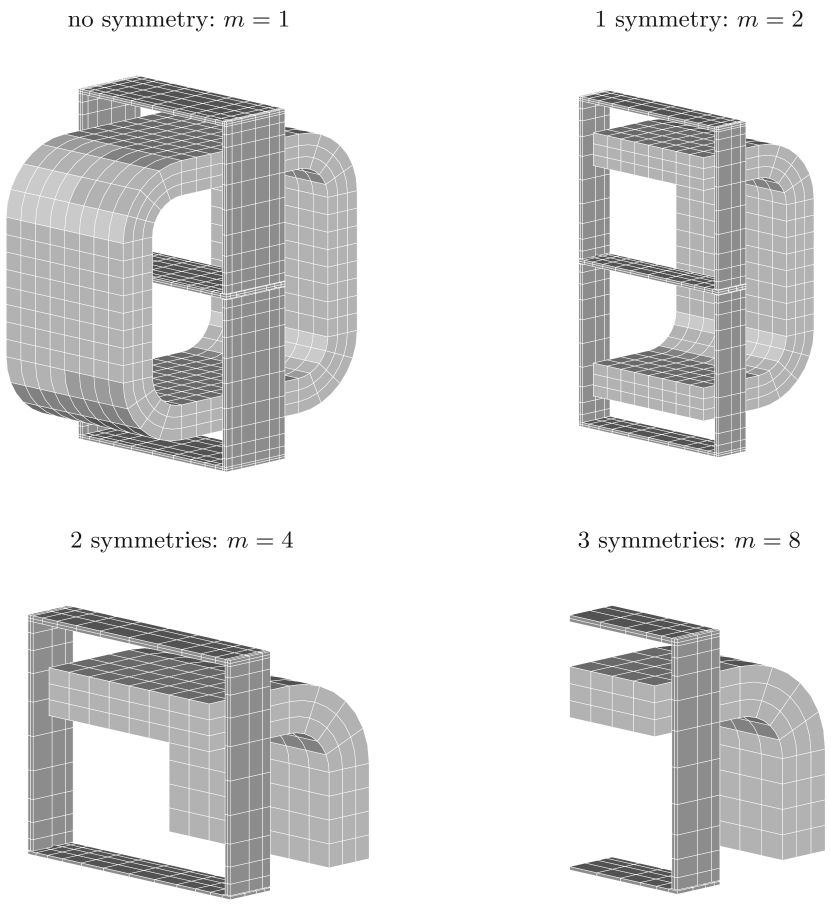

Fig. 13. TEAM problem 10. An exciting coil is set between two steel channels, and a steel plate is inserted between the channels. This geometry is symmetrical with respect to all coordinate planes.

of the single layer potential become singular. However, the global system has a unique solution [1]. There are several methods to handle the subsystems via regularisation or via projections proposed in [1]. Since in our solver no inversion of approximated matrices takes place, also singular matrices can be handled and the unique global solution can still be reconstructed without further difficulties.

TEAM problem 10 is treated as a magnetostatic problem (for details see [11, 16]). For the numerical solution the potential approach is used, so that in the BE domain the equation

$$\boldsymbol{\Delta}\mathbf{A} = -\mu_0 \boldsymbol{\jmath}_S$$

has to be solved, where $\mathbf{A}$ is the Coulomb gauged magnetic vector potential and $\boldsymbol{\jmath}_S$ is the impressed source current density. This equation decouples into three scalar Laplace equations, so that a componentwise discretisation with nodal elements leads to the equation system (56) for each Cartesian component of the vector potential. Additionally, the problem features the excitation symmetry described in Section 4.2, i.e. each Cartesian component of the excitation given by the impressed source current density $\boldsymbol{\jmath}_S$ satisfies the symmetry conditions (60)-(62). Thus, depending on whether the coefficient sets $\{\alpha_k\}$ are different for some Cartesian components, we obtain up to three different system matrices for reduced systems of equations (63). Let us denote them by D_x , D_y and D_z .

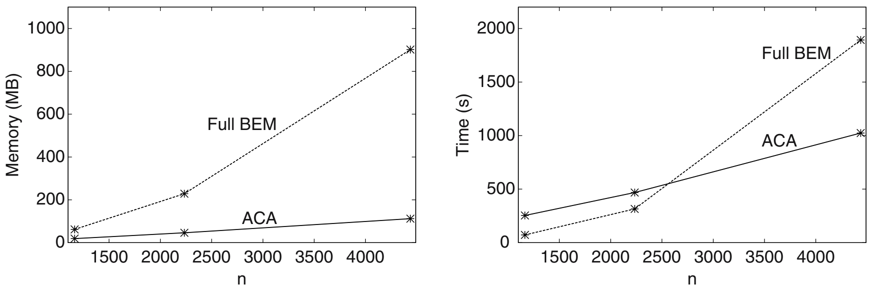

Fig. 14. Memory requirements (left) and CPU times (right) versus problem size for fixed $m = 4$ and variable mesh.

In all computations we set the ACA accuracy $\varepsilon = 10^{-4}$. The problem is solved using both the ACA algorithm and the full BEM method. Figure 14 shows for both algorithms the memory requirement of BEM matrices as well as the CPU time needed for the solution. All values refer to a 450-MHz Sun Ultra workstation. We compared the average magnetic induction in the centre of the inner steel plate ($\bar{B}_z = 1.663\ T$) with measurements ($\bar{B}_z = 1.654\ T$) [16] and found good agreement. The difference of the computed flux densities with and without ACA is neglectable ($\Delta\bar{B}_z \approx 3 \cdot 10^{-4}\ T$).

One can observe for any kind of symmetry that the increasing problem size due to the mesh refinements results in a linear behaviour of the memory consumption and the CPU time for the ACA algorithm. Fig. 14 shows the comparison between the ACA and the full BEM for one kind of symmetry. Although the ACA algorithm is slower for coarse meshes, its linear complexity makes it superior for large n.

Now we examine the effect of the symmetry exploitation. It is clear that the profit using full BEM should be of order $O(n^2)$ whereas the memory requirement and CPU time reduction using ACA is expected to be linear.

Fig. 15 shows the behaviour of the memory usage and CPU time for the medium mesh sequence.

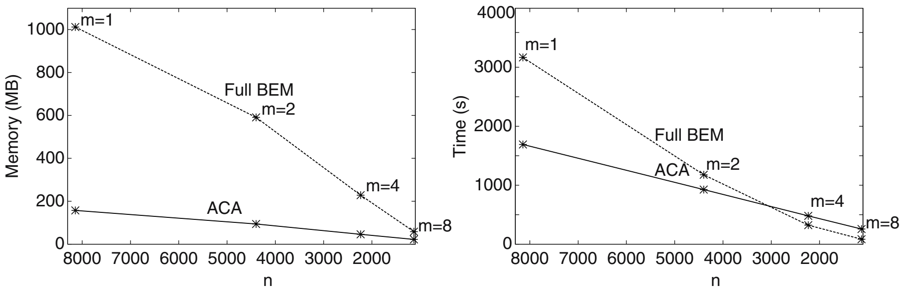

Fig. 15. Memory requirements (left) and CPU times (right) with respect to the symmetry for variable m and fixed mesh (medium discretisation).

As mentioned above in the case of ACA the individual approximation and storage of all m basis matrices will be performed. The relative size of the basis matrices resulting from the single layer potential is shown in Table 3. The assembly of the system matrices in (63) by means of linear combination is carried out in the matrix-vector multiplication.

Table 3. Relative size of BEM matrices coming from the single layer potential for the medium mesh sequence. The percentage gives the relative size after compression obtained by the ACA algorithm for each individual submatrix compared to a fully populated block. Submatrices which involve transformed nodes show a very good compression.

Block matrix	$m = 1$ $n = 8142$	$m = 2$ $n = 4399$	$m = 4$ $n = 2234$	$m = 8$ $n = 1131$
A_1	12.5%	15.4%	20.5%	32.8 %
A_2	-	10.1%	12.9%	18.4 %
A_3	-	-	8.7%	12.3 %
A_4	-	-	6.3%	8.2 %
A_5	-	-	-	6.1 %
A_6	-	-	-	5.1 %
A_7	-	-	-	5.1 %
A_8	-	-	-	3.1 %
Total memory	63.2 MB	37.6 MB	18.4 MB	8.9 MB

The full BEM method performs the assembly of system matrices D_x, D_y, D_z during the matrix computation. The number of different matrices depends on the kind of geometrical and excitational symmetry. For the TEAM

10 example it holds that $D_x = D_y = D_z$ in the case without symmetry, $D_x \neq D_y = D_z$ in the case of one symmetry, and three different matrices arise in case of two or three symmetries. For this reason the curve corresponding to the total memory requirements of the full BEM method in Fig. 15 does not actually decrease like $O(n^2)$ but the memory requirements for each single matrix do.

The numerical example considered here exhibits the property of excitation symmetry. Note that in the general case of non-symmetric excitation the memory requirements would decrease linearly w.r.t. the size m of the symmetry group, as can be seen from the equation (58), and therefore like $O(n)$.

5.3 Industrial Application

In this Section a claw-pole alternator, nowadays a mass-produced article used for the generation of electrical power in vehicles, is considered as an example. The complex magnetic flux guidance requires a three-dimensional modelling of this electrical machine. For an alternator with $p = 6$ pole pairs Fig. 16, left shows a 60°-sector of the solid rotor that coincides with one pole pair. For the same sector Fig. 16, right depicts the supplementary stator part with the inlying stator coils. The entire geometry of the alternator and the magnetic fields are obtained by consecutive rotation of the discretised part by an angle $\Phi = 2\pi/p$ around the machine axis. This periodic symmetry concerning the transformation (47) is obvious.

It is well known that modelling of one pole-pitch ($\Phi = \pi/p$), that is a 30°-sector in our case, is sufficient for the computation of the magnetic field

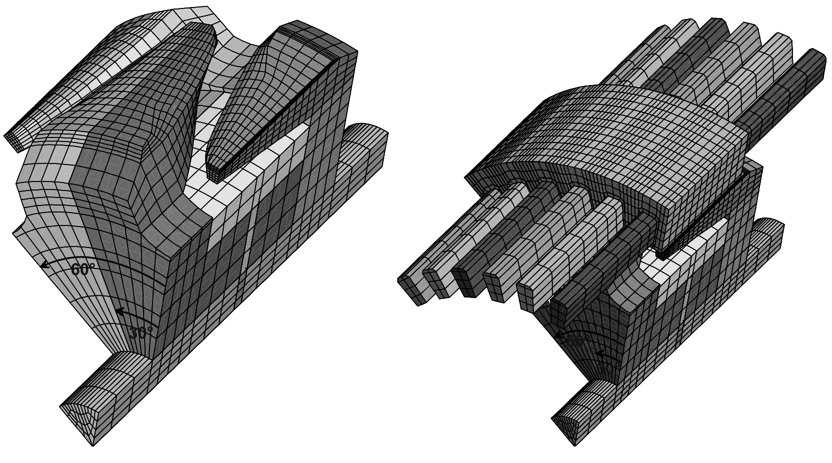

Fig. 16. Discretised rotor part (left) and stator with inlying coils (right) of a claw-pole alternator.

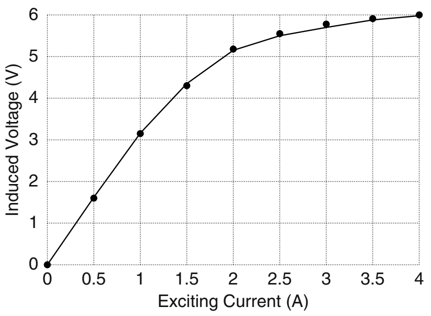

Fig. 17. No-load characteristics at 1000 1/min. The agreement of computed values (solid line) and measurements (dots) is very good.

[10]. For the solution of the eddy current problem hexahedral second order nodal FEM elements have been used, coupled to rectangular second order BEM elements. Fig. 17 shows the induced voltage in the stator coils versus the exciting current at a rotational speed of 1000 rounds per minute.

Table 4. Relative size of matrix blocks of the single layer potential. Submatrices that describe remote interactions show excellent compression.

Block matrix	$m = 3$ (**120**°) $n = 27751$	$m = 6$ (**60**°) $n = 13999$	$m = 12$ (**30**°) $n = 7123$
A_1	13.8%	16.6%	18.0%
A_2	2.8%	5.8%	8.9%
A_3	2.9%	1.8%	4.3%
A_4	-	1.1%	2.7%
A_5	-	1.9%	1.7%
A_6	-	5.9%	1.3%
A_7	-	-	1.2%
A_8	-	-	1.3%
A_9	-	-	1.8%
A_{10}	-	-	2.7%
A_{11}	-	-	4.3%
A_{12}	-	-	8.9%
Total memory	1145.7 MB	494.8 MB	221.1 MB

In order to examine the effect of symmetry exploitation a sequence of three meshes with different symmetry angles has been analysed. The number of boundary nodes n approximately bisects from a 120°- to a 60°- and to a 30°-mesh, respectively, while m reduplicates. For an implementation according to Section 4, Table 4 shows that it is reasonable to approximate and store the submatrices A_k individually. Although interactions between more sectors have to be represented while increasing symmetry exploitation emerging farpoint-interaction gives rise to good approximation of the respective submatrices leading to high-grade compression rates. This is especially important when solving time-dependent problems with motion.

6 Conclusions

The memory consumption of the standard BEM turns out to be the limiting factor in many practical applications. The above results show that the ACA technique is a feasible means to overcome these limitations. ACA can be applied to several BEM formulations [12, 15, 17] discretised by nodal or edge elements where matrices are generated by asymptotically smooth kernels. The combination of the ACA algorithm and the exploitation of symmetry yields an asymptotically optimal and practically feasible procedure for efficient solution of electromagnetic problems.

References

1. E. L. Allgower, K. Georg, R. Miranda, J. Tausch: Numerical exploitation of equivariance. Z. Angew. Math. Mech. 78 (1998) 795–806.
2. M. Bebendorf: Approximation of boundary element matrices. Numer. Math. 86 (2000) 565–589.
3. M. Bebendorf, S. Rjasanow: Adaptive Low-Rank Approximation of Collocation Matrices. Computing 70 (2003) 1–24.
4. M. Bonnet. Exploiting partial or complete geometrical symmetry in 3D symmetric Galerkin indirect BEM formulations. Int. J. Num. Meth. Engrg. 57 (2003) 1053–1083.
5. A. Bossavit: Symmetry, groups and boundary value problems: a progressive introduction to noncommutative harmonic analysis of partial differential equations in domains with geometrical symmetry. Comp. Meth. in Appl. Mech. Engrg. 56 (1986) 167–215.
6. H. Cheng, L. Greengard, V. Rokhlin: A fast adaptive multipole algorithm in three dimensions. J. Comput. Phys. 155 (1999) 468–498.
7. W. Hackbusch: A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I. Computing 62 (1999) 89–108.
8. W. Hackbusch, B. N. Khoromskij: A sparse $\mathcal{H}$–matrix arithmetic. Part II. Application to multi–dimensional problems. Computing (2000) 64 (2000) 21–47.
9. W. Hackbusch, Z. P. Nowak: On the fast matrix multiplication in the boundary element method by panel clustering. Numer. Math. 54 (1989) 463–491.
10. S. Küppers, G. Henneberger, I. Ramesohl: The influence of the number of poles on the output performance of a claw-pole alternator. Proc. of the ICEM, pp. 268–272, 1996.
11. S. Kurz, J. Fetzer, G. Lehner, W. M. Rucker: Numerical analysis of 3D eddy current problems with moving bodies using BEM-FEM coupling. Surv. Math. Industry 9 (1999) 131–150.

12. S. Kurz, O. Rain, V. Rischmüller, S. Rjasanow: Discretization of boundary integral equations by differential forms on dual grids. IEEE Trans. Magnetics 40 (2004) 826–829.
13. S. Kurz, O. Rain, S. Rjasanow: Application of the adaptive cross approximation technique for the coupled BE-FE-solution of symmetric electromagnetic problems. Comp. Mech. 32 (2003) 423–429.
14. T. Nakata, N. Takahashi, K. Fujiwara: Summary of results for benchmark problem 10 (steel plates around a coil). COMPEL 11 (1992) 335–344.
15. J. Ostrowski, Z. Andjelić, M. Bebendorf, B. Crânganu-Creţu, J. Smajić: Fast BEM-solution of Laplace problems with $\mathcal{H}$-matrices and ACA. Proceedings of the IEEE, 2005.
16. K. Preis, I. Bardi, O. Biro, C. Magele, W. Renhart, K. R. Richter, G. Vrisk: Numerical analysis of 3D magnetostatic fields. IEEE Trans. Magnetics 27 (1991) 3798–3803.
17. O. Rain: Kantenelementbasierte BEM mit DeRham-Kollokation für Elektromagnetismus. PhD thesis, Universität des Saarlandes, 2004.
18. S. Rjasanow: Effective algorithms with block circulant matrices. Linear Alg. Appl. 202 (1994) 55–69.