A hybrid finite element method/boundary element method (FEM/BEM) is a standard approach for calculating the magnetostatic potential within Micromagnetics [1]. This involves dealing with a dense \( N \times N \)-matrix \( B_{ij} \), \( N \) being the number of mesh surface nodes. In order to apply the method to ferromagnetic structures with a large surface one needs to apply matrix compression techniques on \( B_{ij} \). An efficient approach is to approximate \( B_{ij} \) by hierarchical matrices (or \( \mathcal{H} \)-matrices). We have used HLib [2], a library containing implementations of the hierarchical matrix methodology, together with the micromagnetic finite element solver Nmag in order to optimize the hybrid FEM/BEM. In this article we present a study of the efficiency of algorithms implemented in HLib concerning the storage requirements and the matrix assembly time in micromagnetic simulations.

\[ \phi_2(\vec{R}) = \frac{1}{4\pi} \int_{\partial R} \psi_j(\vec{r}) \cdot \vec{n}(\vec{r}) \left( \frac{d^2 \vec{r} \cdot \phi_1(\vec{r})}{|\vec{R} - \vec{r}|^3} \right) d\vec{r}. \]  

Third, the \( \phi_2(\vec{R}) \) on \( \partial R \) serve as Dirichlet boundary conditions for solving the Laplace equation for \( \phi_2 \) inside the magnetic region \( \mathcal{R}_m \).

**H-matrices** The \( \mathcal{H} \)-matrix approach splits a matrix into a hierarchical tree of submatrices. The matrix itself is stored in the leaves of the tree. There are two kinds of leaves: Admissible leaves can be approximated by low rank, while inadmissible leaves need to be stored in full. Data compression is achieved through the admissible leaves. For dense matrices stemming from the discretization of a boundary integral a situation as in equation (1) is assumed, \( i.e. \) the kernel of the integral has a singularity where \( \vec{r} = \vec{R} \), but is smooth when \( |\vec{r} - \vec{R}| \) is large. An admissibility criterion, which as in [7] is used with \( \eta = 2.0 \) for all simulations, determines which submatrices correspond to a smooth kernel and therefore can be represented by a low rank matrix. The parameter \( n_{\min} \) specifies a minimal number of rows or columns that a leaf can have, and thus prevents that leaves become arbitrarily small. We found that \( n_{\min} = 30 \) is a reasonable choice.

There are several algorithms to create the low-rank blocks for admissible leaves. The algorithms discussed here can be organized into three classes:

ACA [6] and ACA+ [10] belong to the first class. They start with the already discretized matrix \( B_{ij} \) as defined in equation (2), and build the low-rank approximation...
from just a few entries of the original matrix block. The accuracy of the approximation is determined by a heuristic parameter $\epsilon_{ACA}$. A second parameter $k_{\text{max}}$ defines a maximal rank for the admissible leaves. It has been stated [7, 10] that ACA becomes unreliable for the double layer potential (as in equation (1)). ACA+ has been proposed as a working alternative.

The second class starts from the integral (in our case equation (1)) and expands the kernel with respect to $\vec{r}$ or $\vec{R}$. From that expansion one can assemble the low-rank approximation for the corresponding matrix block. Interpolation [9] uses Lagrange polynomials for this expansion, whose order $p$ determine the accuracy [9]. The algorithm shows good convergence but is inefficient concerning the matrix assembly time [7].

The hybrid cross approximation (HCA) algorithm [7] comprises elements of both, interpolation and ACA. There are two different types of this algorithm. HCA I is closer to interpolation, while HCA II is more similar to ACA. The accuracy of these algorithms can be tuned with two parameters $p$ and $\epsilon_{ACA}$. While being more accurate, HCA II has shown a comparable efficiency to ACA+ [7].

For all these algorithms one can use adaptive recompression [10] to improve storage requirements. This is done by optimizing the rank within the admissible leaves and coarsening the hierarchical structure itself. The accuracy of the recompressed hierarchical matrix can be adjusted with a parameter $\epsilon$.

There are two possible ways of combining HLib with existing code: The first is to use the heuristic ACA(+) to build the hierarchical matrix from few matrix entries, which are computed by the existing code.

Alternatively, one can write extensions to HLib, which compute the matrix entries by numerical integration (for ACA(+)) or by approximating the kernel of the integral and numerical integration (interpolation, HCA I+II). We only had to modify the HLib implementation for the Galerkin discretization of the double layer potential such that it computes the corresponding collocation matrix (1). To perform the numerical integration one needs to specify a quadrature order $q$. Our choice is $q = 3$.

**Numerical Results** In order to investigate the efficiency of the different algorithms we study a finite thin film system of square shape (see Fig. 1). The thickness of the films is $t = 5$ nm and their edge lengths $L$ vary in 10 nm steps between 10 and 260nm, while the coarseness of the mesh is kept constant. All thin films have a constant magnetization $M$ perpendicular to the square surface. The corresponding mesh contains a large number of surface nodes relative to the total number of nodes. Furthermore the magnetometric demagnetizing factor, a quantity which is proportional to the spatially averaged magnetostatic self-energy, can be computed analytically [11], serving as an useful estimate for the comparison of errors introduced by the application of $H$-matrices with errors due to the Hybrid FEM/BEM. However, due to its definition it is unsuitable for detecting local deviations in the demagnetization field. We address the local error further below. One should note that the performance of the algorithms (i.e. efficiency and accuracy) depends on the geometry of the system and the distribution of the magnetisation.

Figure 1 shows how the demagnetization factor obtained from simulations using the full boundary element matrix deviates from the analytical result: The deviation decreases with an increasing number of surface nodes and should be roughly of the order of 0.1 % for realistic node numbers above 20000.

All simulation results on $H$-matrices have been obtained with $p = 5$ for interpolation, $\epsilon_{ACA} = 10^{-4}$ and $k_{\text{max}} = 100$ for ACA and ACA+, $\epsilon_{ACA} = 10^{-7}$ and $p = 6$ for HCA I, $\epsilon = 10^{-7}$ and $p = 4$ for HCA II, and $\epsilon = 10^{-3}$ when recompression has been applied.

Figure 2 shows the deviation in the demagnetization factor between $H$-matrix approximations created by different algorithms and the full BEM solution. All errors are more than one order of magnitude lower than the er-
matrix approximations the additional memory costs due the storage of the hierarchical tree are included.

FIG. 3: Comparison between the storage requirements for \( B_{ij} \) for different algorithms and the full BEM case. For the \( \mathcal{H} \)-matrix approximations the additional memory costs due to the storage of the hierarchical tree are included.

FIG. 4: Comparison between the setup times for \( B_{ij} \) for different algorithms and the full BEM case.

error introduced through the hybrid FEM/BEM (see Fig. 1) and stable with an increasing number of surface nodes. A comparison of the curves for ACA with and without recompression yields that the use of recompression does not affect the error. Generally the deviations observed for ACA and ACA+ are higher than for the other algorithms. They could not be reduced by further decreasing \( \epsilon_{ACA} \). This seems to be related to general problems with the adaptive cross approximation [7, 8]. Looking at the local demagnetization field it turns out that (independently of the film size) the relative error as compared to the full BEM solution does not exceed 0.5% for ACA and ACA+, and 0.03% for interpolation and HCA I+II, i.e. it is of the same order of magnitude as the difference between the analytical and the BEM solution (see for example figure 1). Figure 3 compares the storage requirements for the different algorithms. ACA without recompression exhibits the poorest compression rates. Using recompression ACA and ACA+ show slightly better compression rates than interpolation and HCA I+II. According to figure 4 the matrix assembly time is increased significantly by using recompression for ACA. However, the speed of the recompression is dependent on the algorithm used in the first place [10], and corresponding tests on the ACA+ showed that for this algorithm recompression is faster. A general comparison of the matrix assembly times between the different algorithms (Fig. 4) shows that, as expected, interpolation is relatively slow. The best results are again exhibited by ACA+. HCA II, however, shows a similar efficiency.

Summary We have compared the efficiency of several algorithms implemented within the library HLib. The storage requirements can be reduced significantly by using recompression routines, while the error made by the approximation remains virtually the same. However, dependent on the algorithm, recompression can significantly increase the matrix assembly time. While interpolation, HCA I and HCA II are more accurate than ACA and ACA+, this advantage may not be significant for micromagnetic simulations as the deviation of ACA and ACA+ is at least an order of magnitude smaller than the numerical FEM/BEM inaccuracy. The superior efficiency of ACA+ concerning memory consumption and time needed for the matrix assembly suggest it as the algorithm of choice in the field of micromagnetics. The speed of the algorithms may even increase by not computing the matrix numerically but using an analytical expression [12]. The HLib-library can only be used sequentially. While most of the CPU time goes into the solving of Poisson equations and the (implicit) integration of the equations of motion, it would be desirable to use a parallel version of the HLib for the highest performance.

We thank Steffen Börm and Lars Grasedyck for their support on HLib related issues. This work has been funded by the Engineering and Physical Sciences Research Council (EPSRC) in the United Kingdom (EP/E040063/1, EP/E039944/1).