Journal of Computational Physics 229 (2010) 2544–2549

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

# Kronecker product approximation of demagnetizing tensors for micromagnetics

A.V. Goncharov [a,*], G. Hrkac [a], J.S. Dean [a], T. Schrefl [a,b]

[a] Department of Engineering Materials, The University of Sheffield, Sheffield, UK
[b] Department of Computer Science, St. Poelten University of Applied Science, St. Poelten, Austria

ARTICLE INFO

*Article history:*
Received 18 June 2009
Received in revised form 26 November 2009
Accepted 2 December 2009
Available online 11 December 2009

*Keywords:*
Micromagnetics
Fast field
Low-rank

ABSTRACT

A comparison study of the asymptotic behavior between different compression techniques is reported. We show that by applying the Kronecker product approximation, the storage of a three-dimensional demagnetizing tensor with $N^6$ entries can be reduced to $O(N^2)$, showing a superlinear compression behavior. When magnetization and magnetostatic field vectors are stored in compressed forms, a superlinear speedup of a field evaluation is gained.


## 1. Introduction

The increasing industrial demand for large scale simulations leads to huge scale matrix equations, which require a high computational power. Various methods have been developed for data compression to treat such large matrices. Such techniques include Fast Multipole Method (FMM), *H*-matrices etc. [1,2]. Recently it has been shown that matrices, arising from the discretization of integral equations with fast decaying kernels, possess a good Kronecker product approximation [3–5]. Those matrices normally have a very small rank ($R \ll N$), which makes application of low-rank approximations feasible. More details on the Kronecker product approximation are given in Section 3. The advantage of this type of approximation compared to other techniques is its superlinear compression property. If in large scale three-dimensional simulations one space dimension is discretized by $N$ cells, then a total number of cells is $N^3$. Direct integration algorithms will scale with a total number of cells squared giving $N^6$ for the full $N^3 \times N^3$ matrix. The Kronecker approximation allows us to store only $O(N^2)$ entries, which is less than the order of the original matrix. In the following we apply this type of approximation to the point-function demagnetizing tensor discretized on a tensor product grid, with $N$ 1 nm cubic cells in each dimension. Using the demagnetizing tensor in the compressed form, the magnetostatic energy is calculated for problem sizes ranging from $10^3$ cells to $8 \times 10^6$ cells. Corresponding problem sizes range from 13 Mb to approximately 500 Tb of RAM, for double precision numbers. Results of memory consumption are then compared with other common compression techniques. Magnetostatic energy and magnetization vectors are compressed using Kronecker format and evaluation times are also plotted versus problem sizes. All computations were performed using a 2 GHz processor with 3Mb L2 cache.

* Corresponding author. Tel.: +44 (0)1142226021.
*E-mail addresses:* a.goncharov@sheffield.ac.uk (A.V. Goncharov), g.hrkac@sheffield.ac.uk (G. Hrkac), j.dean@sheffield.ac.uk (J.S. Dean), t.schrefl@sheffield.ac.uk (T. Schrefl).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.12.004

## 2. Discretization

The brief introduction of the problem being solved is given in this section. In following calculations the magnetostatic scalar potential formalism is used for the field evaluation inside the ferromagnetic body. The magnetic scalar potential at the position given by $\mathbf{r}$ and induced by the magnetization distributed over the domain $\Omega'$ is given by the following volume integral [6]:

$$\phi(\mathbf{r}) = \int_{\Omega'} \mathbf{M}(\mathbf{r}') \cdot \nabla' \left( \frac{\mathbf{1}}{|\mathbf{r} - \mathbf{r}'|} \right) d^3\mathbf{r}' \tag{1}$$

The magnetostatic field at location $\mathbf{r}$ is then given by

$$\mathbf{H}(\mathbf{r}) = -\nabla\phi = -\frac{\mathbf{1}}{\mathbf{4}\pi} \nabla \int_{\Omega'} \mathbf{M}(\mathbf{r}') \cdot \nabla' \left( \frac{\mathbf{1}}{|\mathbf{r} - \mathbf{r}'|} \right) d^3\mathbf{r}' \tag{2}$$

This integral can be quite difficult to solve analytically for the arbitrary shaped ferromagnetic body with a nonuniform magnetization distribution. The value of the field is found numerically using discretization. The computational domain $\Omega'$ is divided into computational cells. The integration over the domain with nonuniform magnetization is split into integrals over elements where magnetization is assumed to be constant. It is known that for a uniformly magnetized ferromagnetic body the demagnetizing field can be computed using the demagnetizing tensor [7]. In this case the new discretized equation is given by:

$$\mathbf{H}(\mathbf{r_i}) = \sum_{j=1}^{N^3} N(\mathbf{r_i})_\mathbf{j} \mathbf{M_j}, \tag{3}$$

where $N^3$ is the total number of cells and

$$N(\mathbf{r_i})_\mathbf{j} = -\frac{1}{4\pi} \int_{V_\mathbf{j}} \nabla\nabla' \left( \frac{1}{|\mathbf{r_i} - \mathbf{r}'|} \right) d^3\mathbf{r}' \tag{4}$$

is the point-function demagnetizing tensor for the **j**th cell. The field is evaluated at location $\mathbf{r_i}$, which is the centre of the **i**th cell. Integrals in (4) over cubic cells can be evaluated analytically as in [8]. The tensor has nine components and it is convenient to rewrite Eq. (3) in component form with $\mathbf{r_i}$ replaced by **i** index:

$$H^p_\mathbf{i} = \sum_{\mathbf{j}=\mathbf{1}}^{N^3} N^{pq}_\mathbf{ij} M^q_\mathbf{j} \tag{5}$$

Indices $p$ and $q$ in (5) run from 1 to 3 (for $x$, $y$ and $z$, respectively) and same index implies summation over all components of the magnetization. Eq. (5) can be written in matrix form as:

$$H^p = N^{pq} M^q, \tag{6}$$

where $N^{pq}$ is now a $N^3 \times N^3$ matrix with $N^6$ entries and $H^p$ and $M^q$ are vectors containing cartesian components of field and magnetization (both have length $N^3$).

## 3. Kronecker approximation

To show how the Kronecker approximation arises we shall start from the Eq. (6) for the magnetostatic field. Let us limit to the case where $p = q = 1$, i.e. $H^x(r) = N^{xx}(r, r')M^x(r')$. The procedure for all other cases is the same. On a cartesian grid, the indices $\mathbf{i}, \mathbf{j}$ in (3)–(6) are replaced with $(i, j, k)$ and $(i', j', k')$, respectively. Then (5) can be written in tensor form:

$$H^x_{ijk} = \sum_{i'=1}^{N} \sum_{j'=1}^{N} \sum_{k'=1}^{N} N^{xx}_{ijki'j'k'} M^x_{i'j'k'} \tag{7}$$

where $H^x_{ijk} = H^x(x_i, y_j, z_k)$ is the $x$-component of the magnetostatic field in the centre of the $(ijk)$ cell and $M^x_{i'j'k'}$ is the constant magnetization in the $(i'j'k')$ cell. The matrix $N^{xx}$ can be viewed as a tensor:

$$N^{xx}_{ijki'j'k'} = \int_{V_{i'j'k'}} N^{xx}(x_i, y_j, z_k, x', y', z')\, dx'dy'dz' \tag{8}$$

If the kernel $N^{xx}$ decays fast with the distance between the source and the field points, then the full tensor (8) has a small rank and can be approximated by a data sparse tensor [4,5]. If this is the case then the kernel allows a separable approximation with a small rank $R \ll N$:

$$N^{xx}(x, y, z, x', y', z') = \sum_{r=1}^{R} P^r(x, x') Q^r(y, y') R^r(z, z'), \tag{9}$$

By substituting (9) in (8) the low-rank tensor product approximation is obtained:

$$N^{xx}_{i,j,k,i',j',k'} = \sum_{r=1}^{R} P^r_{i,i'} Q^r_{j,j'} R^r_{k,k'}, \tag{10}$$

where $R$ is the rank of the approximation and Kronecker factors $P^r,\ Q^r,\ R^r$ are $N \times N$ matrices, computed as follows:

$$\begin{aligned} P^r_{i,i'} &= \int_{x^0_{i'}}^{x^1_{i'}} P^r(x_i, x')dx', \\ Q^r_{j,j'} &= \int_{y^0_{j'}}^{y^1_{j'}} Q^r(y_j, y')dy', \\ R^r_{k,k'} &= \int_{z^0_{k'}}^{z^1_{k'}} R^r(z_k, z')dz' \end{aligned} \tag{11}$$

where $x^0_{i'},\ x^1_{i'},\ y^0_{j'},\ y^1_{j'},\ z^0_{k'},\ z^1_{k'}$ are boundaries of the $(i'j'k')$ cell in $x,\ y$ and $z$ directions, respectively. If the analytical approximation (9) is not known, then the Kronecker factors can be computed using numerical optimization algorithms, such as an alternating least squares (ALS) [11]. More details on the Kronecker approximation and its existence for matrices arising from the discretization of integral operators with fast decaying kernels in two and higher dimensions are given in [3,10]. In the remaining part of this section a brief description of how the theory is applied to the matrix Eq. (6) is given. It will be shown how the demagnetizing matrix (4) can be written in Tensor product form using the Kronecker approximation. We will discuss how the Kronecker approximation can be computed by first computing a Tucker decomposition [12] of the demagnetizing tensor which is then modified to compute the Kronecker approximation. For this purpose we write the approximation (10) in equivalent form using tensor product notation:

$$\mathbf{N} = \sum_{r=1}^{R} \mathbf{P}_r \otimes \mathbf{Q}_r \otimes \mathbf{R}_r, \tag{12}$$

where $\mathbf{P}_r,\ \mathbf{Q}_r$ and $\mathbf{R}_r$ are $N \times N$ matrices of order $N$, $R$ is a Kronecker rank and $\otimes$ means tensor product. If Kronecker factors in (12) are viewed as $N^2$ vectors then (12) represents a canonical decomposition of a 3D $N^2 \times N^2 \times N^2$ tensor:

$$\begin{aligned} \mathbf{N} &= \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{v}_r \otimes \mathbf{w}_r \\ n_{lmn} &= \sum_{r=1}^{R} u_{lr} \cdot v_{mr} \cdot w_{nr} \end{aligned} \tag{13}$$

Now $\mathbf{u}_r,\ \mathbf{v_r}$ and $\mathbf{w_r}$ are $r$th columns of $N^2 \times R$ matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$. These columns are $N \times N$ Kronecker factors from Eq. (12) packed in a column-wise format. Eq. (13) is a sum of rank-1 tensors or triads and is called PARAFAC (PARAllel FACtors) decomposition [9]. Equivalence of Eqs. (12) and (13) follows from the properties of the Kronecker product of matrices and was shown in details for two dimensions in [3]. Extension to the three-dimensional case is straightforward. Assembly of the tensor out of the matrix can be done by using the bijection method as in [10]. It follows that for an $N^3 \times N^3$ matrix arising from a 3D tensor product grid any row $I$ and column $J$ can be treated as an $N \times N \times N$ tensor with a column-row-wise ordering. Then matrix indices $I$ and $J$ can be replaced with triplets:

$$\begin{aligned} I &= i + (j-1)N + (k-1)N^2, \\ J &= i' + (j'-1)N + (k'-1)N^2 \end{aligned} \tag{14}$$

where $1 \leqslant I, J \leqslant N^3$ and $1 \leqslant i,\ j,\ k,\ i',\ j',\ k' \leqslant N$. Now indices $(ijk)$ and $(i'j'k')$ give positions of field and source cells in the cartesian grid and the field is calculated using a six-dimensional tensor as: $h^p_{ijk} = n^{pq}_{ijki'j'k'} \cdot m^q_{i'j'k'}$. This tensor can be transformed to the three-dimensional form if indices are combined as follows: $(i,i')$, $(j,j')$ and $(k,k')$. Each combination will produce one dimension of length $N^2$ of a resulting 3D tensor as follows:

$$\begin{aligned} l &= i + (i'-1)N \\ m &= j + (j'-1)N \\ n &= k + (k'-1)N \end{aligned} \tag{15}$$

If matrices $P,\ Q$ and $R$ in (12) are chosen as: $P = [(k,k')],\ Q = [(j,j')],\ R = [(i,i')]$, then (12) follows from the property of the Kronecker product. By looping through indices $(ijk)$ and $(i'j'k')$ and using relations (15) for $(lmn)$ indices the $N^2 \times N^2 \times N^2$ tensor can be assembled for the compression in form (13).

Applying canonical decomposition to the full size problem is an elaborate computational task. There are various methods described elsewhere [11], but they will all fail for large industrial problem sizes. However, the approximation (13) can be obtained in two steps with a relatively small effort. At the first step, the demagnetizing tensor is compressed into the Tucker

form using a 3D ACA (three-dimensional adaptive cross approximation) algorithm described in [12]. The algorithm requires only $O(N)$ operations and does not require storage of the original full tensor. The result is the Tucker decomposition of the original full size problem:

$$\begin{aligned}&\mathbf{N} = \mathbf{G}\times_1\mathbf{A}\times_2\mathbf{B}\times_3\mathbf{C},\\ &n_{ijk} = \sum_{l=1}^{R}\sum_{m=1}^{R}\sum_{n=1}^{R} g_{lmn}\cdot a_{il}\cdot b_{jm}\cdot c_{kn}\end{aligned} \tag{16}$$

where $\mathbf{G}$ is the $R\times R\times R$ core tensor, $A$, $B$ and $C$ are orthogonal Tucker factors, each of size $N^2\times R$ and $\times_n$ stands for an $n$-mode tensor-matrix multiplication. The second equation represents element-wise multiplication of factors with the core. Brief description of the Tucker format and its properties can be found in [13]. Even though the memory storage requirement is now reduced from $N^6$ down to $R^3+3RN^2$, which is already asymptotically superlinear, the presence of the 3D core brings large overhead for computations. Since the Tucker rank is usually small compared to $N$ the core can be effectively compressed into the canonical form. For this purpose the Levenberg–Marquardt algorithm has been used [14]. Usage of the nonlinear optimization is justified by the small $R$ of the core. The core can be expressed now as:

$$\mathbf{G} = \sum_{r=1}^{R}\mathbf{g}_r^1\otimes\mathbf{g}_r^2\otimes\mathbf{g}_r^3, \tag{17}$$

where $\mathbf{g}_r^1$, $\mathbf{g}_r^2$, $\mathbf{g}_r^3$ are $r$th columns of $R\times R$ PARAFAC factors. Substitution of (17) into (16) gives the required approximation (13), which is equivalent to (12). Storing demagnetizing tensor in the Kronecker format offers not only excellent compression quality but also a superlinear speedup in computations. To make an evaluation of the matrix-vector product in (6) computationally effective, field and magnetization vectors are also viewed as $N\times N\times N$ tensors and compressed into canonical forms using ALS algorithm [11]:

$$\begin{aligned}\mathbf{H}^p &= \sum_{r=1}^{F_1}\mathbf{h}_r^{p1}\otimes\mathbf{h}_r^{p2}\otimes\mathbf{h}_r^{p3}\\ \mathbf{M}^q &= \sum_{r=1}^{F_2}\mathbf{m}_r^{q1}\otimes\mathbf{m}_r^{q2}\otimes\mathbf{m}_r^{q3},\end{aligned} \tag{18}$$

with vectors $\mathbf{h}_r^{pi}$ and $\mathbf{m}_r^{qi}$ being columns of the $i$th $N\times R$ PARAFAC matrix and $p$ and $q$ stand for $x$, $y$, $z$ components. The rank $F_2$ has to be equal to the product $RF_1$, where $R$ is the Kronecker rank of the demagnetizing tensor. Details of the matrix-vector evaluation in Kronecker formats are given in [10]. After the field has been computed the magnetostatic energy is evaluated in $O(N)$ operations using the inner product of field and magnetization tensors in PARAFAC format:

$$w_{mag} = \sum_{p}^{3}\sum_{r1}^{F_1}\sum_{r2}^{F_2}\langle\mathbf{h}_{r1}^{p1},\mathbf{m}_{r2}^{p1}\rangle\cdot\langle\mathbf{h}_{r1}^{p2},\mathbf{m}_{r2}^{p2}\rangle\cdot\langle\mathbf{h}_{r1}^{p3},\mathbf{m}_{r2}^{p3}\rangle \tag{19}$$

## 4. Results

The compression technique described in the previous section is now applied for a calculation of the magnetostatic energy of the ferromagnetic cube for two cases. In the first case the energy is calculated for the uniform magnetization along the positive $Z$ direction. This case was chosen deliberately because the energy for a uniformly magnetized cube is known from the theory and computed result can be checked with this value. The magnetostatic field for such a case is simply given by $H^z=-N^{zz}M^z$, where $M^z=M_sm_z$. For the sake of simplicity, magnetization of saturation is set to 1, then the field $H^z$ is equal to $-m_z/3$ and the energy density: $\mu_0 w_{mag}=m_z^2/6$. For the second case the magnetization in the cube is allowed to curl in $XY$ plane, so all three components of the magnetization were used. Due to the symmetry, only three components of the demagnetizing tensor from Eq. (6) were used for the field calculation. For the first case with uniform magnetization the compression (12) is applied to the $N^{zz}$ discretized tensor. Field and magnetization vectors $m_z$ and $H^z$ are also compressed into the Kronecker format as in (18). The technique is applied to a number of discretization cases involving up to 8 million computational cells. Corresponding number of cells in one dimension $N$ ranged from 10 to 200. Results of memory savings gained by using $N^{zz}$ in compressed form are shown in Fig. 1. The graph shows amount of RAM (log scale) required to store $N^{zz}$ in megabytes versus $N$ (also log scale). Five curves are shown in Fig. 1. One is serving as an example of memory consumption for a full problem size if no compression is applied to $N^{zz}$. There are also two curves representing asymptotical behavior of the memory requirement for some advanced techniques used today: $H$-matrix with $O(N^3\log N^3)$ and FMM with $O(N^3)$ memory cells, respectively. Asymptotical curves mean that there are no prefactors included, only the dependence on $N$ is plotted. The third asymptotic is a superlinear Kronecker curve with $O(N^2)$ behavior (shown with triangles pointing up). This curve shows a vast improvement over the linear ($O(N^3)$) one. The last curve presented in Fig. 1 is real compression data with a prefactor of $3R$. Rank $R$, which is vital for the quality of the compression is ranging from 6 to 17. These values of rank guaranteed that the error in magnetostatic energy was below $1\times10^{-2}$. It can be seen that the experimental curve follows a superlinear trend and even outperforms the linear asymptotic for $N\geqslant 40$. For the largest problem size tested with 8 million cells the

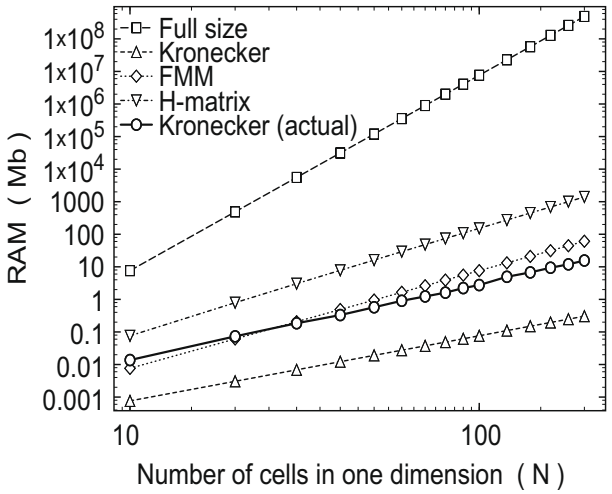


**Fig. 1.** Comparison of asymptotical compression behavior for different compression techniques versus grid size in one dimension. The line with square symbols is the full problem size, with triangles down is $N^3 \log N^3$ ($H$-matrix), with diamonds is $N^3$ (linear, FMM) and with triangles up is $N^2$ (superlinear, Kronecker). In addition there is also an experimental compression curve (circles) which contains a prefactor of $3R$ which follows a superlinear trend. Nevertheless for $N \geqslant 40$ it outperforms the linear asymptotic.

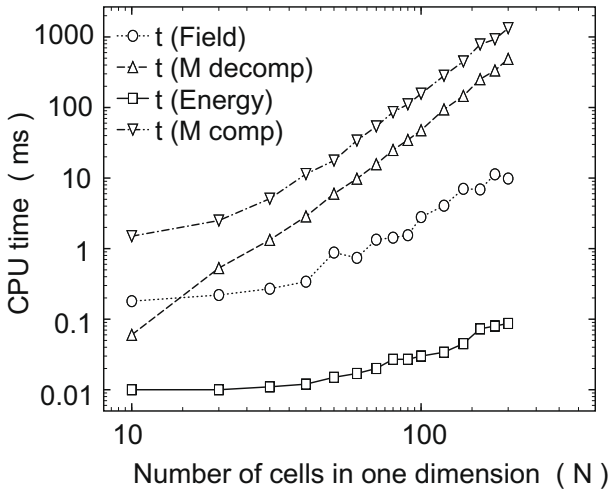


**Fig. 2.** CPU times measured in milliseconds. Times are plotted as follows: the top curve with triangles down corresponds to compression of the $m_z$, the curve with triangles up shows decompression into the vector form of $m_z$ and $H^z$, the curve with circles shows time for matrix-vector evaluation in Kronecker format and the curve with squares is for energy calculation. All are versus $N$ – number of cells in one dimension.

amount of RAM required to store $N^{zz}$ is only 15 Mb. To test a speedup provided by the compression, the CPU time required for calculation of the magnetostatic field was measured during computations. Results of these findings are presented in Fig. 2. The curve shows a drastic decrease in evaluation time of the matrix-vector product. For the problem with 8 million cells the time spent is only 10 ms. Magnetostatic energy calculations using formula (19) requires less than 0.1 ms for this problem size. However, if the dynamic micromagnetic simulations are considered, then there are two linear complexity problems which have to be considered. At each time step, compression of the magnetization into the PARAFAC form (18) is done before energy calculation. After the effective field has been computed the field and magnetization tensors must be converted back into the vector form for the time integration. This can be done in $O(N^3 + N^2)$ operations, which is a linear complexity for the original $N^3 \times N^3$ matrix. The energy for the second case, where all three components of the magnetization are used, is calculated for the problem with $N^3 = 8 \times 10^6$ cells. Due to the cubic symmetry only 3 components of the demagnetizing tensor: $N^{xx}$, $N^{xy}$ and $N^{xz}$ need to be compressed. The compression rank for the tensors equal to $R = 18$ and the error in the magnetostatic energy is $3 \times 10^{-3}$. The CPU time needed for the matrix-vector evaluation, energy computation and decompression are 111 ms, 4.5 ms, and 7.8 s, respectively.

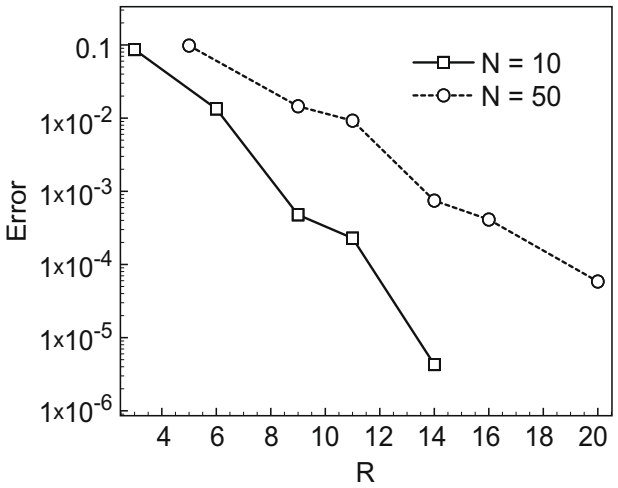


**Fig. 3.** Absolute error in the magnetostatic energy versus the rank of the approximation for two cubes with 1000 and 125,000 cells, respectively. The error is plotted using a logarithmic scale.

For the practical use of demagnetizing tensors in compressed form it is important to know how the error of the approximation changes with the rank. The error analysis was carried out using two cube models with 1000 and 125,000 computational cells. The magnetization was along the $z$ direction and the magnetostatic field was calculated using compressed demagnetizing tensors with different ranks. For each rank, the approximation of energy was computed, and compared with the exact value of $m_z^2/6$. The absolute error in energy versus the rank of the approximation is plotted in Fig. 3. It shows that approximations of the energy obtained in this work show an exponential convergence, which is in agreement with the theory [4].

## 5. Conclusion

The magnetostatic energy for the ferromagnetic cube is calculated using the demagnetizing tensor and the magnetization in Kronecker formats. Results of memory savings and CPU time follow theoretically predicted superlinear behavior. This method outperforms those with the linear scaling for matrix-vector evaluation and energy calculation. However, for dynamic micromagnetic simulations, decompression of the field and magnetization into the vector form is required and can be done in $O(N^3 + N^2)$ operations.

## References

[1] L. Greengard, V. Rokhlin, Acta Numerica 6 (1997) 229–269.
[2] W. Hackbush et al, Mathematica Bohemica 2 (2002) 229–241.
[3] E. Tyrtyshnikov, Linear Algebra and its Applications 379 (2004) 423–437.
[4] W. Hackbush, B.N. Khoromskij, Computing 76 (2006) 177–202.
[5] W. Hackbush, B.N. Khoromskij, Computing 76 (2006) 203–225.
[6] J.D. Jackson, Classical Electrodynamics, third ed., John Wiley & Sons, Inc., 1999.
[7] R. Moskowitz, E. Della Torre, IEEE Transactions on Magnetics 4 (1966) 739–744.
[8] H. Fukushima, Y. Nakatani, N. Huyashi, IEEE Transactions on Magnetics 34 (1) (1998) 193–198.
[9] R.A. Harshman, UCLA Working Papers in Phonetics 16 (1970) 1–84.
[10] W. Hackbush, B.N. Khoromskij, E.E. Tyrtyshnikov, Journal of Numerical Mathematics 13 (2) (2005) 119–156.
[11] G. Tomasi, R. Bro, Chemometrics and Intelligent Laboratory Systems 75 (2005) 163–180.
[12] I. Oseledets, D. Savostianov, E. Tyrtyshnikov, SIAM Journal on Matrix Analysis and Applications 30 (3) (2008) 939.
[13] B.N. Khoromskij, V. Khoromskaia, CEJM 5 (3) (2007) 523–550.
[14] P. Paatero, Chemometrics and Intelligent Laboratory Systems 38 (1997) 223–242.