A fast finite-difference algorithm for topology optimization of permanent magnets

Claas Abert,1 Christian Huber,1 Florian Bruckner,1 Christoph Vogler,2 Gregor Wautischer,1 and Dieter Suess1

1Christian Doppler Laboratory of Advanced Magnetic Sensing and Materials, Faculty of Physics, University of Vienna, Austria
2Faculty of Physics, University of Vienna, Austria
(Dated: August 1, 2017)

Abstract

We present a finite-difference method for the topology optimization of permanent magnets that is based on the FFT accelerated computation of the stray-field. The presented method employs the density approach for topology optimization and uses an adjoint method for the gradient computation. Comparsion to various state-of-the-art finite-element implementations shows a superior performance and accuracy. Moreover, the presented method is very flexible and easy to implement due to various preexisting FFT stray-field implementations that can be used.

claas.abert@univie.ac.at
I. INTRODUCTION

Permanent magnets are a key technology for many industrial applications ranging from sensors [1, 2] to electric generators and motors [3]. The field generated by these magnets is usually required to have certain properties such as high values and high/low gradients in certain regions. These properties can be controlled by either designing the magnetization configuration of the magnet or its geometry. Since the production of magnets with complicated inhomogeneous magnetization configurations is rather involved, the optimization of the geometry of a homogeneously magnetized material is often the most promising approach for field optimization. The production of arbitrarily shaped magnets has become very affordable due to recent developments in the 3D-printing technology [4–6].

In numerical mathematics there are several approaches to geometry optimization. In general there are two classes of methods, namely shape optimization and topology optimization. For shape optimization the geometry is usually parametrized with a relatively low number of degrees of freedom and optimized with respect to these variables.

Topology optimization is much less restrictive. As the name suggests, the geometry may even change its topology during optimization. However, this generality usually comes at the price of a large number of degrees of freedom which leads to high computational costs. Topology optimization has a long history in the magnetic community [7]. Most of the previously presented methods employ the finite-element method for the field calculation. In this work we present a finite-difference method for shape optimization that uses an FFT accelerated convolution for the field computation. Compared to previously presented approaches the presented method is exceptionally fast and easy to implement.

II. METHODOLOGY

We employ the density approach for topology optimization [8]. In this approach, the topology is described by a scalar indicator function

\[
\rho(x) = \begin{cases} 
0 : & \text{no material} \\
1 : & \text{material} 
\end{cases} .
\] (1)
For the optimization process we also allow intermediate values $0 \leq \rho \leq 1$. With this indicator function the magnetization field $M$ can be written as

$$M(\rho) = \rho^p M_0$$

(2)

where $M_0$ is the prescribed magnetization that may be either spatially constant or varying depending on the application. Note, that we introduced the exponent $p$ as suggested in [8] in order to penalize intermediate values of $\rho$. For the optimization we consider a general objective function of the form

$$\hat{J}(\rho) = J(H(\rho), \rho)$$

(3)

that should be minimized. Here, $H = -\nabla u$ is the magnetic stray field generated by the magnetization $M(\rho)$ with $u$ being its scalar potential governed by the Poisson equation

$$F = \Delta u - \nabla \cdot M = 0$$

(4)

with open boundary conditions

$$u(x) = O(1/|x|) \quad \text{if} \quad |x| \to \infty.$$  

(5)

A minimum of the objective function $\hat{J}$ with respect to the indicator function $\rho$ requires the derivative $d\hat{J}/d\rho$ to vanish. The computation of the derivative is also desirable from a numerical point of view since it can be used for iterative minimization with gradient based methods. The derivative of $\hat{J}$ can be written as

$$\frac{d\hat{J}}{d\rho} = \frac{\partial \hat{J}}{\partial H} \frac{dH}{du} \frac{du}{d\rho} + \frac{\partial \hat{J}}{\partial \rho}.$$  

(6)

For typical choices of the objective function $\hat{J}$, the partial derivatives $\partial \hat{J}/\partial H$ and $\partial \hat{J}/\partial \rho$ can be expressed in a closed analytical form. However, the computation of $du/d\rho$ is nontrivial since the dependence of the scalar potential $u$ on the indicator function $\rho$ is given by the constraint $F$ which is a partial differential equation. Numerical computation of $du/d\rho$ by finite differences is possible but infeasible since this procedure requires the solution of $F$ for every degree of freedom of $\rho$ individually. This shortcoming can be overcome by solution of an adjoint equation [9]. Consider the derivative of the constraint $F$

$$\frac{dF}{d\rho} = \frac{\partial F}{\partial u} \frac{du}{d\rho} + \frac{\partial F}{\partial \rho} = 0.$$  

(7)
Solving for $du/d\rho$, inserting into (6), and applying the adjoint approach yields

$$\frac{d\hat{J}}{d\rho} = \lambda^* \frac{\partial F}{\partial \rho} + \frac{\partial \hat{J}}{\partial \rho} \quad (8)$$

$$\frac{\partial F^*}{\partial u} \lambda = -\frac{\partial H^*}{\partial u} \frac{\partial \hat{J}^*}{\partial \rho} \quad (9)$$

with $\lambda$ being the so-called adjoint variable. Inserting (4) results in the system

$$\frac{d\hat{J}}{d\rho} = p\rho^{p-1} M_0 \cdot \nabla \lambda + \frac{\partial \hat{J}}{\partial \rho} \quad (10)$$

$$\Delta \lambda = \nabla \cdot \left( \frac{\partial \hat{J}}{\partial H} \right). \quad (11)$$

Note that (11) has exactly the same form as the constraint (4), i.e. the right-hand side of the Poisson equation is given as the divergence of a vector entity. Moreover, the knowledge of the gradient of the adjoint variable $\nabla \lambda$ is sufficient for the computation of the derivative (10). This means that both the forward problem $F$ as well as the adjoint problem (10) and (11) can be expressed in terms of the stray-field operator

$$\hat{D} : C^0(\mathbb{R}^3, \mathbb{R}^3) \rightarrow C^0(\mathbb{R}^3, \mathbb{R}^3) \quad (12)$$

with

$$H = \hat{D}(M) \quad (13)$$

that maps the magnetization vector field $M$ onto the vector field $H$. With this definition the objective function and its derivative can be written as

$$\hat{J} = J(\hat{D}[M(\rho)], \rho) \quad (14)$$

$$\frac{d\hat{J}}{d\rho} = -p\rho^{p-1} M_0 \cdot \hat{D} \left( \frac{\partial \hat{J}}{\partial H} \right) + \frac{\partial \hat{J}}{\partial \rho} \quad (15)$$

This formulation can be readily used with arbitrary numerical methods for the stray-field computation to perform topology optimization.

### III. DISCRETIZATION

Various numerical algorithms for the discrete computation of the stray-field have been proposed, see e.g. [10]. Among the fastest and most accurate algorithms is the fast-Fourier-transform (FFT) accelerated convolution with the demagnetization tensor $\hat{N}$. The
prerequisite for this method is a regular cuboid grid that enables the formulation of the demagnetization-field problem as a discrete convolution

\[
H_i = \sum_j \tilde{N}_{i-j} m_j \tag{16}
\]

\[
\tilde{N}_{i-j} = -\frac{1}{4\pi V_{\text{cell}}} \int_{\Omega_i} \int_{\Omega_j} \nabla \nabla' \frac{1}{|x - x'|} \, dx \, dx' \tag{17}
\]

where \( V_{\text{cell}} \) is the volume of a single simulation cell and \( \Omega_i \) and \( \Omega_j \) are the simulation cells at multiindex \( i \) and \( j \) respectively. Note that, due to the regularity of the grid, the sixfold integral only depends on the difference of multiindices \( i \) and \( j \) and not on their specific values. While a naive implementation of the convolution (16) would require a computational complexity of \( O(N^2) \), the computation in Fourier space and application of the FFT reduces the complexity to \( O(N \log N) \). This procedure, including the accurate computation of the discrete demagnetization tensor (17) and the optimal implementation of the fast convolution is well documented, e.g. in micromagnetic literature [11–13]. Moreover, various open-source implementations exist that can be used to implement the presented topology-optimization strategy [13–15]. For this work, we use the CPU code of the micromagnetic simulator magnum.fd [16]. Minimization of the objective function is performed with a quasi-Newton method that is able to handle the constraint \( 0 \leq \rho \leq 1 \) of the indicator function.

In order to compare our implementation with respect to accuracy and performance, we implement two additional methods based on the finite-element method (FEM). Finite-element methods solve the Poisson equation (4) by means of a variational approach and work on arbitrary tetrahedral meshes. However, the treatment of the required open boundary conditions is nontrivial with FEM. For the first FEM approach we extend the mesh beyond the region of interest that is used for the topology optimization. The size of the extended mesh is chosen to be approximately 5 times larger than the original mesh in each spatial dimension and we apply zero Dirichlet boundary conditions on the outer boundary. This so-called truncation approach was already shown to provide good results for topology optimization in [17]. We solve the stray-field potential \( u \) by the weak formulation

\[
\int_{\Omega_{\text{all}}} \nabla u \cdot \nabla v \, dx = \int_{\Omega_{\text{mag}}} M \cdot \nabla v \, dx \tag{18}
\]

where \( \Omega_{\text{all}} \) is the complete meshed region, \( \Omega_{\text{mag}} \) is the magnetic region and the trial and test functions \( u \) and \( v \) are discretized with piecewise affine, globally continuous functions.
The downside of the truncation approach is the requirement of additional mesh nodes which increases both the storage requirements as well as the computational costs. The additional mesh nodes can be avoided by application of a hybrid finite-element/boundary-element method (FEM/BEM) [18]. We use the same function spaces as for the pure FEM truncation approach. For the BEM part we use different implementations, namely a collocation approach [19] and a Galerkin approach with and without matrix compression via $\mathcal{H}$-matrices [20].

For the FEM implementation we use the multipurpose library FEniCS [21], for the BEM implementation we use BEM++ [22] and for $\mathcal{H}$-matrix compression H2Lib [23]. The minimization for all methods is done with the L-BFGS-B minimizer of the SciPy library [24]. In the following we will refer to the truncation approach as FEM and to the hybrid method as FEM/BEM. The FFT accelerated method will be referred to as FD.

IV. VALIDATION AND BENCHMARKS

For validation and benchmarking purposes, we consider a simple test problem. We aim to maximize the $z$-component of the stray field in a small box above a unit cube with magnetization $M_0 = (0, 0, 1)$ that is considered for topology optimization, see Fig. 1. For the FEM method we add an external mesh which increases the number of mesh nodes from 13,072 to 25,055 and the number of cells from 60,189 to 146,522. The objective function for the problem reads

$$\hat{J} = -\frac{1}{2} \int_{\Omega} H_z^2 \, d\mathbf{x}$$

where $\Omega$ is the region where the field is maximized. The corresponding derivative reads

$$\frac{d\hat{J}}{d\rho} = p\rho^{p-1} M_0 \cdot \hat{D} (\chi_\Omega H_z)$$

6
FIG. 1. Geometry for the topology-optimization benchmark problem. The large blue cube marks the region considered for topology optimization. The z-component of the stray field is maximized in the green box. (a) Finite-difference mesh with 62,361 cells. (b) Finite-element mesh with 13,072 nodes and 60,189 cells.

FIG. 2. Optimized topology for maximum z-component of the stray-field in a small box above the optimization region that is magnetized in z-direction. The results for the presented finite-difference algorithm (a),(d) is shown along with the results for FEM (b),(e) and collocation FEM/BEM (c),(d). (a)–(c) Indicator function $\rho$ in the optimization region (red = 1, blue = 0). (d)–(f) Optimized geometry ($\rho > 0.1$).

with $\chi_{\Omega}$ being the characteristic function of the region $\Omega$. We set $p = 3$ and choose $\rho(\mathbf{x}) = 1$ as start condition for the iterative optimization.

Figure 2 shows the resulting topology as computed with the presented finite-difference algorithm compared to the results computed with FEM and collocation FEM/BEM. For all approaches the same L-BFGS-B method with identical tolerances was applied. At a first glance the quality of the FD solution seems better than those of the FEM and FEM/BEM so-
FIG. 3. Cell distribution of indicator function for the optimized topology computed with various methods. The number of cells $N$ is plotted against the value of $\rho$.

solutions. The regular cuboid grid leads to a relatively smooth representation of the optimized geometry compared to the other methods. Furthermore, there seem to be less simulation cells with intermediate values for the indicator function $\rho$. This impression is confirmed by Fig. 3 that shows the cell distribution of the indicator function $\rho$ for the different methods. For this particular problem, every individual simulation cell either increases or decreases the objective function. Hence, intermediate values of $\rho$ indicate an inaccurate simulation result. A possible reason for this behaviour is the accuracy of the discrete stray-field operator $\hat{D}$. If the operator is inaccurate it might not reflect the self-adjoint properties of the original problem. Thus, the gradient computed by (11) might not accurately fit the objective function (3) which leads to bad convergence of the iterative optimization procedure.

Figure 4 shows a comparison of the convergence for the individual methods. In the beginning, all methods except the collocation FEM/BEM with $H$-matrix compression show almost the same descent velocity in the volume fraction of the optimized topology. Both FEM/BEM methods with $H$-matrix compression require significantly more function evaluations to converge than the remaining methods. Obviously, the gradient computation is very sensitive to approximations in the stray-field computation. The FEM method converges only slightly slower than FD and collocation FEM/BEM with dense BEM matrix. However, it saturates at a significantly higher volume fraction than the reference solution.

Table I shows the timings for the individual methods. All simulations were carried out as single threaded on a standard laptop computer with an Intel Core i7 2.90 GHz CPU and
FIG. 4. Convergence speed comparison of the topology optimization with various stray-field methods. The volume fraction of the optimized volume $V$ is plotted against the number of right-hand-side evaluations $i_{\text{iter}}$. The black dashed line denotes the reference solution computed with the FD method and a 4 times higher resolution in every spatial dimension.

| Method               | $N_{\text{iter}}$ | $T_{\text{stray}}$ [ms] | $T_{\text{iter}}$ [s] | $T_{\text{total}}$ [s] |
|----------------------|--------------------|--------------------------|------------------------|-------------------------|
| FD                   | 25                 | 31.0                     | 0.196                  | 4.9                     |
| FEM                  | 31                 | 57.0                     | 2.303                  | 71.4                    |
| FEM/BEM col. dense   | 19                 | 69.0                     | 0.826                  | 15.7                    |
| FEM/BEM col. $\mathcal{H}$-matrix | 97               | 49.6                     | 0.684                  | 66.4                    |
| FEM/BEM Gal. $\mathcal{H}$-matrix | 73               | 62.1                     | 0.781                  | 57.0                    |

TABLE I. Comparison of convergence and timings for various stray-field methods. $N_{\text{iter}}$ denotes the total number of L-BFGS-B iterations, $T_{\text{stray}}$ denotes the time for a single stray-field computation, $T_{\text{iter}}$ denotes the average time of a single L-BFGS-B iteration, and $T_{\text{total}}$ denotes the total time for the optimization excluding setup time.

8 GB RAM. The finite-element matrices were solved with the sparse direct solver MUMPS [25]. The setup time required for the assembly of the demagnetization tensor and the FEM matrices as well as the matrix factorization for the direct solver were excluded from the timings. By far the fastest method is the presented FD method that outperforms all other methods both in terms of stray-field computation as well as total time. The second fastest method is the collocation FEM/BEM with dense BEM matrix. However, we recall that the optimization result of this method is of bad quality as it contains a large number of
FIG. 5. Scaling of the stray-field computation time of the FD method compared to FEM/BEM with direct solver and conjugate gradient solver respectively. The computation time $t$ is plotted against the numbers of degrees of freedom $N$ (cells in the case of FD and nodes in the case of FEM/BEM).

intermediate values for $\rho$. All other methods are at least a factor of 10 slower than the presented method. The performance gain of the FD method compared to the FEM/BEM method will even be more significant for larger systems as suggested by Fig. 5. Both methods seem to scale approximately linear which can be explained by the algorithmic complexity of both methods of $O(N \log N)$. However, at large problem sizes the FD method offers a tremendous performance gain compared to FEM/BEM. For $10^6$ degrees of freedom, FD is more than a factor of 25 faster than FEM/BEM. Note, that the timings for FEM/BEM with a direct solver could only by determined for small systems, since the memory consumption of the matrix factorization exceeded the capabilities of the test machine for larger systems.

Note also that Fig. 5 shows the computation time of FEM/BEM with respect to the number of mesh nodes instead of cells, as the finite-element systems for the potential $u$ scales with the number of nodes. However, even if instead the number of cells is considered for the FEM/BEM scaling, the FD method outperforms FEM/BEM by a factor of 5, while still being more accurate.

V. OPTIMIZATION OF FIELD AND GRADIENT

In a more complex example the field and its gradient should be optimized in certain areas of a plane above the region subject to topology optimization, see Fig. 6(a). The objective
FIG. 6. Maximization of field and gradient on a plane above the magnet. (a) Top-view of the areas of field and gradient optimization. (b) Optimized geometry and \( z \)-component of the resulting stray-field (red = large values, blue = small values).

Function for this experiment reads

\[
\dot{J} = -\frac{1}{2} \int_{\Omega_{\text{field}}} H_z^2 \, dx - \frac{\beta}{2} \int_{\Omega_{\text{grad}}} \left( \frac{dH_z}{dn} \right)^2 \, dx
\]  

(21)

where \( n \) is the outward-pointing unit vector of the region of maximum field and \( \beta \) is chosen as \( 10^5 \). The region of maximum field \( \Omega_{\text{field}} \) has a size of \( 20 \times 20 \) simulation cells and the region of maximum gradient \( \Omega_{\text{grad}} \) has a width of 1 simulation cell and marks three sides of the maximum field region, see Fig. 6(a). The resulting derivative of the objective function reads

\[
\frac{d\dot{J}}{d\rho} = p \rho^{p-1} M_0 \cdot \dot{D} \left( \chi_{\Omega_{\text{field}}} H_z + \beta \chi_{\Omega_{\text{grad}}} \frac{d^2H_z}{dn^2} \right)
\]  

(22)

with \( \chi_{\Omega_{\text{field}}} \) and \( \chi_{\Omega_{\text{grad}}} \) being the characteristic functions of \( \Omega_{\text{field}} \) and \( \Omega_{\text{grad}} \) respectively. We use a finite-difference three-point stencil for the approximation of the second derivative of \( H_z \). Similar to the simple test problem in the preceding section we choose \( M_0 = (0, 0, 1) \) and \( p = 3 \) and perform the optimization with a L-BFGS-B method. The optimization result is depicted in Fig. 6(b).

VI. CONCLUSION

We present a fast and accurate finite-difference method for topology optimization of permanent magnets with respect to their stray field. The implementation of the method is simple due to the possibility to use existing highly optimized codes for the computation of
the magnetic stray field. We compare the method to various finite-element implementations and demonstrate that the presented method is significantly faster and more accurate than any finite-element implementation. For typical applications the possibly irregular mesh used by the finite-element method is considered an advantage over the regular cuboid grid that is required by the finite-difference method. However, this advantage only exists for predefined geometries where irregular meshes are able to accurately approximate complex structures with a relatively low number of nodes. For topology optimization the geometry is not known upfront and thus a regular mesh might even be favorable because of the simple geometric representation. The presented method is general and can easily be extended by additional terms to the objective function such a volume constraints or higher order derivatives.

ACKNOWLEDGEMENTS

The financial support by the Austrian Federal Ministry of Science, Research and Economy and the National Foundation for Research, Technology and Development as well as the Austrian Science Fund (FWF) under grant F4112 SFB ViCoM, and the Vienna Science and Technology Fund (WWTF) under grant MA14-44, is gratefully acknowledged.

[1] C. Treutler, Sensors and Actuators A: Physical 91, 2 (2001), third European Conference on Magnetic Sensors & Actuators.
[2] J. Coey, Journal of Magnetism and Magnetic Materials 248, 441 (2002).
[3] O. Gutfeldisch, M. A. Willard, E. Brück, C. H. Chen, S. Sankar, and J. P. Liu, Advanced materials 23, 821 (2011).
[4] C. Huber, C. Abert, F. Bruckner, M. Groenefeld, O. Muthsam, S. Schuschnigg, K. Sirak, R. Thanhofer, I. Teliban, C. Vogler, et al., Applied Physics Letters 109, 162401 (2016).
[5] B. G. Compton, J. W. Kemp, T. V. Novikov, R. C. Pack, C. I. Nlebedim, C. E. Duty, O. Rios, and M. P. Paranthaman, Materials and Manufacturing Processes 1, 1 (2016).
[6] J. Jaćimović, F. Binda, L. G. Herrmann, F. Greuter, J. Genta, M. Calvo, T. Tomše, and R. A. Simon, Advanced Engineering Materials (2017).
[7] F. Campelo, J. Ramrez, and H. Igarashi, (2010).
[8] M. P. Bendsoe and O. Sigmund, *Topology Optimization* (Springer, 2004).

[9] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich, *Optimization with PDE constraints*, Vol. 23 (Springer Science & Business Media, 2008).

[10] C. Abert, L. Exl, G. Selke, A. Drews, and T. Schrefl, Journal of Magnetism and Magnetic Materials **326**, 176 (2013).

[11] S. W. Yuan and H. N. Bertram, IEEE transactions on magnetics **28**, 2031 (1992).

[12] N. Hayashi, K. Saito, and Y. Nakatani, Japanese journal of applied physics **35**, 6065 (1996).

[13] M. J. Donahue, IEEE Transactions on Magnetics **45**, 3923 (2009).

[14] A. Vansteenkiste, J. Leliaert, M. Dvornik, M. Helsen, F. Garcia-Sanchez, and B. Van Waeyenberge, Aip Advances **4**, 107133 (2014).

[15] C. Abert, F. Bruckner, C. Vogler, R. Windl, R. Thanhoffer, and D. Suess, Journal of Magnetism and Magnetic Materials **387**, 13 (2015).

[16] “magnum.fd,” [http://micromagnetics.org/magnum.fd/](http://micromagnetics.org/magnum.fd/).

[17] C. Huber, C. Abert, F. Bruckner, M. Groenefeld, I. Teliban, C. Vogler, and D. Suess, arXiv preprint [arXiv:1702.01968](http://arxiv.org/abs/1702.01968) (2017).

[18] D. Fredkin and T. Koehler, IEEE Transactions on Magnetics **26**, 415 (1990).

[19] D. Lindholm, IEEE transactions on magnetics **20**, 2025 (1984).

[20] W. Hackbusch, *Hierarchical matrices: algorithms and analysis*, Vol. 49 (Springer, 2015).

[21] M. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. E. Rognes, and G. N. Wells, Archive of Numerical Software **3**, 9 (2015).

[22] W. Śmigaj, T. Betcke, S. Arridge, J. Phillips, and M. Schweiger, ACM Transactions on Mathematical Software (TOMS) **41**, 6 (2015).

[23] “H2lib,” [http://www.h2lib.org](http://www.h2lib.org).

[24] E. Jones, T. Oliphant, P. Peterson, *et al.*, “SciPy: Open source scientific tools for Python,” (2001–), [Online; accessed [today]].

[25] P. R. Amestoy, I. S. Duff, J.-Y. LExcellent, and J. Koster, in *International Workshop on Applied Parallel Computing* (Springer, 2000) pp. 121–130.