Scalable Parallel Micromagnetic Solvers for Magnetic Nanostructures

Werner Scholz Josef Fidler Thomas Schrefl Dieter Suess
Rok Dittrich Hermann Forster Vassilios Tsiantos

Vienna University of Technology, Institute of Solid State Physics, Wiedner Hauptstrasse 8-10/138, A-1040 Vienna, Austria

Abstract

A parallel finite element micromagnetics package has been implemented, that is highly scalable, easily portable and combines different solvers for the micromagnetic equations. The implementation is based on the standard Galerkin discretization on tetrahedral meshes with linear basis functions. A static energy minimization, a dynamic time integration, and the nudged elastic band method have been implemented. The details of the implementation and some aspects of the optimization are discussed and timing and speedup results are given. Nucleation and magnetization reversal processes in permalloy nanodots are investigated with this micromagnetics package.

Key words: micromagnetics, finite element method, scalable, parallel, nanostructures
PACS: 02.70.Dh, 75.75.+a

1 Introduction

In various fields of computer aided engineering like structural analysis, fluid dynamics, and electromagnetic field computation, as well as micromagnetics [1] the finite element method has been successfully applied. In particular, its flexibility in modeling arbitrary geometries has made it very popular. In the light of the importance of the microstructure of magnetic materials the finite element method has been chosen for the implementation of a micromagnetics package.

Email address: werner.scholz@tuwien.ac.at (Werner Scholz).
URL: http://magnet.atp.tuwien.ac.at (Werner Scholz).

There are several commercial and open source micromagnetics packages available [2–8], however all of them use the finite difference method.

Our finite element micromagnetics package combines several unique features:

It is

- entirely based on portable, free, open source software packages,
- highly portable to different hardware platforms, which range from simple PCs to massively parallel supercomputers,
- highly optimized and scalable,
- well integrated, combining static energy minimization, dynamic time integration, and nudged elastic band methods.

The basic micromagnetic equations and their discretization in the context of the finite element method are outlined in Sec. 2, while the appropriate solution methods are described in Sec. 3. The details of the implementation of the micromagnetics package are discussed in Sec. 4 and the optimization strategies are explained in Sec. 5. Finally, its application to the simulation of nucleation and magnetization reversal processes in permalloy nanodots is presented in Sec. 6.

2 Finite Element Micromagnetics

The finite element method requires the discretization of the spatial domain Ω with "finite elements" – a regular triangulation. For two dimensional problems triangles and rectangles, in three dimensions tetrahedral and hexahedral elements are commonly used. Also a mixture of different types of elements is possible, but after the evaluation of various other implementations and for simplicity a tetrahedral discretization has been implemented. As compared to finite difference methods [9] the finite element mesh may be entirely unstructured, which makes the modeling of complicated geometries and irregular microstructures more convenient. However, the creation of the geometrical model and its triangulation are still very demanding tasks, which require sophisticated (commercial) tools. PATRAN by MSC.SOFTWARE [10] and GID [11] (there is a free academic version available) are suitable software packages for which import filters have been implemented.

The required resolution of the finite element mesh – the maximum size of the finite elements – is determined by the smallest features which might occur in the solution of the PDE. These features have to be properly resolved and the approximation of the exact solution by the test functions has to be sufficiently accurate to give meaningful results.

In micromagnetics these limits are defined by three typical lengths:

$$l_{\mathrm{bex}} = \sqrt{A/K_1} \quad , \quad \delta_{\mathrm{BW}} = \pi l_{\mathrm{bex}} \tag{1}$$

δ_{BW} is the width of Bloch type magnetic domain walls (typically found in bulk hard magnets), A is the exchange constant and K_1 is the magnetocrystalline anisotropy constant. [12]

$$l_{\mathrm{nex}} = \sqrt{2\mu_0 A/J_{\mathrm{s}}^2} \quad , \quad \delta_{\mathrm{NW}} = \pi l_{\mathrm{nex}} \tag{2}$$

δ_{NW} is the width of Néel walls (typically found in soft magnetic materials and thin films), J_{s} is the saturation polarization.

$$l_{\mathrm{thex}} = \sqrt{\frac{A}{J_{\mathrm{s}} H_{\mathrm{th}}}} \quad , \quad H_{\mathrm{th}} = \sqrt{\frac{2\alpha k_{\mathrm{B}} T}{\Delta t \gamma J_{\mathrm{s}} l^3}} \tag{3}$$

l_{thex} is the thermal exchange length with the thermal field H_{th}, which depends on the Gilbert damping constant α, the temperature T, time step of the numerical time integration scheme Δt, the gyromagnetic ratio γ, and the spatial correlation length l (which equals the cell size) [13].

In the most general case, the maximum cell size of the finite element mesh has to be smaller than the minimum of the three lengths defined above [14].

However, if the structure of the solution is roughly known, it is possible to use an "adapted" mesh for the simulations. It has a high resolution (small elements) in areas with very small features (e.g. domain walls) and a low resolution (large elements) in other areas where the solution (magnetization) is very uniform. As a result, the number of nodes (and therefore the number of unknowns) and, thus, the time required for the simulation can be greatly reduced. For dynamic time dependent problems a rigid adapted mesh is often not suitable, because the solution changes over time and the smallest features of the solution move through the mesh. If they leave the high resolution mesh and reach areas with larger finite elements, artificial pinning on the mesh will occur and give wrong results. Thus, an adaptive mesh refinement method is required, which changes the structure of the mesh during the simulation and adapts the resolution of the mesh to the solution. It can be shown, that these methods lead to (almost) optimal complexity and give most accurate results with the smallest numerical (computational) effort [15]. Adaptive mesh refinement methods are still a very active research area, and they have been successfully applied also in numerical micromagnetics [16–20]. However, this adaptive meshing technique has not been implemented (yet) in our micromagnetics package.

2.1 Gibbs Free Energy

The total energy of a micromagnetic system is given by the Gibbs free energy E_{tot}, which depends on the magnetic polarization, the external field and some (temperature dependent) material parameters. It includes macroscopic contributions such as the Zeeman energy and the magnetostatic energy as well as microscopic contributions like the magnetocrystalline anisotropy energy and the exchange energy. The external field is independent of the magnetization distribution and the exchange and anisotropy energy are short range interactions, which depend only on the local magnetization distribution. Thus, they can be computed very efficiently. However, the magnetostatic field is a long-range interaction, that is the most expensive part in terms of memory requirements and computation time. Its calculation is usually based on a magnetic vector [21] or scalar potential (cf. Sec. 2.3.4). In addition, it is an open boundary problem, for which various methods have been developed [22,23].

The total Gibbs free energy is given by [24,25]

$$E_{\text{tot}} = \int_\Omega (w_{\text{exch}} + w_{\text{ani}} + w_{\text{ext}} + w_{\text{demag}}) \, dv =$$

$$= \int_\Omega \Bigg(A\Big((\nabla \boldsymbol{u}_x)^2 + (\nabla \boldsymbol{u}_y)^2 + (\nabla \boldsymbol{u}_z)^2 \Big) +$$

$$K_1(1 - (\boldsymbol{a} \cdot \boldsymbol{u})^2) - \boldsymbol{J} \cdot \boldsymbol{H}_{\text{ext}} - \frac{1}{2}\boldsymbol{J} \cdot \boldsymbol{H}_{\text{demag}} \Bigg) dv \quad , \tag{4}$$

where

$$\boldsymbol{J}(\boldsymbol{x}, t) = J_{\text{s}}(\boldsymbol{x}) \cdot \boldsymbol{u}(\boldsymbol{x}, t) \quad , \quad |\boldsymbol{u}| = 1 \tag{5}$$

describes the magnetic polarization as a function of space and time. A is the exchange constant, K_1 is the first magnetocrystalline anisotropy constant and $\boldsymbol{a}$ the unit vector parallel to the easy axis, $\boldsymbol{H}_{\text{ext}}$ the external field, and $\boldsymbol{H}_{\text{demag}}$ the demagnetizing field.

In thermodynamic equilibrium a micromagnetic system tries to reach a state with minimum total energy. The aim of micromagnetic theory is to find the magnetic polarization in equilibrium. Brown proposed a variational method [24], that is based on the calculation of the variational derivative of the total energy with respect to the magnetic polarization. In equilibrium (in an energy minimum) the coefficients of the linear term vanish for any variation $\delta \boldsymbol{u}$

$$\frac{\delta E_{\text{tot}}}{\delta \boldsymbol{u}} = 0 \quad . \tag{6}$$

This leads to Brown's equations

$$\boldsymbol{u} \times \left(2A\Delta\boldsymbol{u} + 2K_1\boldsymbol{a}(\boldsymbol{u} \cdot \boldsymbol{a}) + J_s\boldsymbol{H}_{\text{ext}} + J_s\boldsymbol{H}_{\text{demag}} \right) = 0 \tag{7}$$

Thus, in equilibrium the magnetic polarization $\boldsymbol{J}$ is parallel to an "effective field"

$$\boldsymbol{H}_{\text{eff}} = \frac{2A}{J_s}\Delta\boldsymbol{u} + \frac{2K_1}{J_s}\boldsymbol{a}(\boldsymbol{u} \cdot \boldsymbol{a}) + \boldsymbol{H}_{\text{ext}} + \boldsymbol{H}_{\text{demag}} \tag{8}$$

and the torque, which acts on the polarization, vanishes

$$\boldsymbol{J} \times \boldsymbol{H}_{\text{eff}} = 0 \quad . \tag{9}$$

Since any contribution parallel to the polarization $\boldsymbol{J}$ does not add to the torque, it does not make any difference if the magnetic field $\boldsymbol{H}$ or the magnetic induction $\boldsymbol{B} = \mu_0\boldsymbol{H} + \boldsymbol{J}$ is used for the effective field.

2.2 Gilbert Equation of Motion

The minimization of Eq. (4) can find an equilibrium magnetization distribution. However, the energy landscape of micromagnetic systems is usually very complicated and contains many local maxima, minima, and saddle points. Therefore, the choice of the initial magnetization distribution has a strong influence on the result. A more physical approach to the problem and a more realistic approach of the system to its equilibrium in a local minimum is provided by a dynamic description of the path through the energy landscape.

The motion of a magnetic moment in a magnetic field is mainly governed by its Larmor precession around the local magnetic field. The damping of the precession causes the relaxation to equilibrium. There are many processes that contribute to the damping in a magnetic solid like magnon-magnon and magnon-phonon interactions, interactions between localized and itinerant electrons and eddy currents, for example [26–28].

The *Gilbert equation* [29,30] describes the precession and combines all damping effects in a phenomenological damping term with a single damping constant α

$$\frac{d\boldsymbol{J}}{dt} = -\gamma\boldsymbol{J} \times \boldsymbol{H} + \frac{\alpha}{J_s}\boldsymbol{J} \times \frac{d\boldsymbol{J}}{dt} \quad , \tag{10}$$

where $\gamma = 2.210173 \times 10^5 \frac{\text{m}}{\text{As}}$ is the gyromagnetic ratio.

This formulation is equivalent to the *Landau-Lifshitz-Gilbert (LLG) equation*

$$\frac{d\boldsymbol{J}}{dt} = -\gamma'\boldsymbol{J} \times \boldsymbol{H} - \frac{\alpha\gamma'}{J_s}\boldsymbol{J} \times (\boldsymbol{J} \times \boldsymbol{H}) \quad , \tag{11}$$

with $\gamma' = \frac{\gamma}{1+\alpha^2}$.

The intrinsic timescale is determined by the Larmor frequency $\omega = \gamma\boldsymbol{H}_{\text{eff}}$, which is usually in the order of GHz. Thus, the precession time is smaller than a nanosecond, which requires time steps in the order of picoseconds or even less. This limits the maximum simulated time to about 100 ns.

2.3 Discretization

In the following sections we will discretize the contributions to the total energy with the finite element method. For static energy minimization methods as well as for the calculation of the effective field Eq. (14) we have to calculate the derivative of the total energy with respect to the local magnetic polarization $\boldsymbol{J}$. In the following sections we will also derive these gradients.

First we have to define the discrete approximation of the magnetic polarization $\boldsymbol{J}(\boldsymbol{x})$ (Eq. (5)) by

$$\boldsymbol{J}(\boldsymbol{x}) \approx J_s(\boldsymbol{x})\sum_i \boldsymbol{u}_i\eta_i \approx \sum_i J_{s,i}\boldsymbol{u}_i\eta_i = \sum_i \boldsymbol{J}_i \quad , \tag{12}$$

where η_i denotes the basis function (hat function) at node i of the finite element mesh. We are using isoparametric elements, i.e. we use the same polynomials (linear basis functions) for the approximation of the geometry and the solution. The material parameters A, K, and J_s are defined element by element and they are assumed to be constant within each element. However, the magnetic polarization, which depends on the saturation polarization J_s, is defined on the nodes. Thus, we have to introduce the node based discrete approximation $J_{s,i}$ of the saturation polarization $J_s(\boldsymbol{x})$ as

$$J_{s,i} = \frac{\int_\Omega J_s(\boldsymbol{x})\eta_i \, dv}{\int_\Omega \eta_i \, dv} \quad . \tag{13}$$

Since $\boldsymbol{J}$ is a vector with three Cartesian components, the number of unknowns to be calculated is three times the number of nodes.

6

Thus, we get for the effective field using the box scheme [31]

$$\boldsymbol{H}_{i,\text{eff}} = -\frac{1}{V_i}\left(\frac{\delta E_{\text{tot}}}{\delta \boldsymbol{J}}\right)_i \approx -\frac{1}{V_i}\frac{\partial E_{\text{tot}}}{\partial \boldsymbol{J}_i} = -\frac{1}{V_i J_{\text{s},i}}\frac{\partial E_{\text{tot}}}{\partial \boldsymbol{u}_i} \quad , \tag{14}$$

where $V_i = \int_\Omega \eta_i \, dv$ denotes the volume that is assigned to node i of the mesh.

2.3.1 Exchange Energy

The exchange energy for one Cartesian component is given by

$$E_{\text{exch}} = \int_\Omega \sum_j A(\nabla u_j \eta_j)^2 \, dv \quad . \tag{15}$$

For the gradient we obtain

$$\frac{\partial E_{\text{exch}}}{\partial u_i} = \int_\Omega \sum_j A\frac{\partial}{\partial u_i}\left(\nabla u_j \eta_j\right)^2 \, dv \tag{16}$$

$$\frac{\partial}{\partial u_i}\left(\nabla u_j \eta_j\right)^2 = 2u_j\nabla\eta_j \cdot \frac{\partial u_j\nabla\eta_j}{\partial u_i} = \tag{17}$$

$$= 2u_j\nabla\eta_j \cdot \nabla\eta_j \delta_{ij} = \tag{18}$$

$$= 2u_j\nabla\eta_j \cdot \nabla\eta_i \quad . \tag{19}$$

Finally, the gradient of the exchange energy is given by

$$\frac{\partial E_{\text{exch}}}{\partial u_i} = 2A\int_\Omega \sum_j u_j\nabla\eta_j \cdot \nabla\eta_i \, dv \quad , \tag{20}$$

which can be written as a linear system of equations with the coefficient matrix

$$G_{\text{exch},ij} = 2A\int_\Omega \nabla\eta_j \cdot \nabla\eta_i \, dv \quad . \tag{21}$$

The gradient can then be simply calculated as

$$\boldsymbol{g}_{\text{exch}} = G_{\text{exch}} \cdot \boldsymbol{u} \quad . \tag{22}$$

The expressions for the x, y, and z-component are identical and there are no mixed terms.

This exchange energy matrix is proportional to the stiffness matrix of the Laplacian operator, which is also obvious from Brown's equation Eq. (7) and the effective field Eq. (8).

2.3.2   Magnetocrystalline Anisotropy Energy

The magnetocrystalline anisotropy energy for uniaxial anisotropy is given by

$$E_{\mathrm{ani}} = \int_\Omega \sum_j K_1 (1 - (\boldsymbol{a} \cdot \boldsymbol{u}_j \eta_j)^2) \, dv \quad . \tag{23}$$

The gradient is given by

$$\frac{\partial E_{\mathrm{ani}}}{\partial u_{i,l}} = \int_\Omega \sum_j K_1 \frac{\partial}{\partial u_{i,l}} \left(1 - \left(\sum_k^{\{x,y,z\}} (a_k \cdot u_{j,k} \eta_j) \right)^2 \right) dv \tag{24}$$

$$\frac{\partial}{\partial u_{i,l}} \left(\sum_k^{\{x,y,z\}} (a_k \cdot u_{j,k} \eta_j) \right)^2 = 2 \sum_k^{\{x,y,z\}} (a_k \cdot u_{j,k} \eta_j) \cdot \sum_m^{\{x,y,z\}} (a_m \delta_{ij} \delta_{lm} \eta_j) =$$

$$= 2 \sum_k^{\{x,y,z\}} (a_k \cdot u_{j,k} \eta_j) \cdot a_l \eta_i \tag{25}$$

and we get the result

$$\frac{\partial E_{\mathrm{ani}}}{\partial u_{i,l}} = -2K_1 a_l \int_\Omega \sum_j \sum_k^{\{x,y,z\}} a_k u_{j,k} \eta_j \cdot \eta_i \, dv \quad . \tag{26}$$

This can be rewritten in matrix notation as

$$\boldsymbol{g}_{\mathrm{ani}} = G_{\mathrm{ani}} \cdot \boldsymbol{u} \tag{27}$$

with

$$G_{\mathrm{ani},i,l} = -2K_1 a_l \int_\Omega \sum_k^{\{x,y,z\}} a_k \eta_j \cdot \eta_i \, dv \quad . \tag{28}$$

2.3.3   Zeeman Energy

The Zeeman energy of a magnetic body $\boldsymbol{J}(\boldsymbol{x})$ in an external field $\boldsymbol{H}_{\mathrm{ext}}(\boldsymbol{x})$ is simply given by

$$E_{\text{ext}} = \int_\Omega w_{\text{ext}}(\boldsymbol{J}) \, dv = \int_\Omega \left(-\boldsymbol{J} \cdot \boldsymbol{H}_{\text{ext}} \right) dv = \tag{29}$$

$$= \int_\Omega \sum_j J_{\text{s},j} \sum_k^{\{x,y,z\}} u_{j,k} \eta_j H_{\text{ext},k} \, dv \quad . \tag{30}$$

Since we know the external field explicitly, we can just simply add it to the other contributions to the effective field.

2.3.4   Demagnetizing Field and Magnetostatic Energy

The demagnetizing field is a little more complicated to handle, because it is an "open boundary problem" with one of its boundary conditions at infinity. The magnetostatic potential can be calculated directly by integration using the Green's function, but this method is very slow and expensive in storage. Fast multipole methods have been developed to overcome these problems [32–34]. Fredkin and Koehler [35,1,36] proposed a hybrid finite element/boundary element method that requires no finite elements outside the magnetic domain Ω. A comparison of these methods can be found in [20].

The magnetic scalar potential φ has to solve

$$\Delta \varphi = \nabla \cdot \boldsymbol{J}(\boldsymbol{x})/\mu_0 \quad \text{for } \boldsymbol{x} \in \Omega \tag{31}$$
$$\Delta \varphi = 0 \quad \text{for } \boldsymbol{x} \notin \Omega \tag{32}$$

with the boundary conditions at the boundary Γ of Ω

$$\varphi_{\text{ext}} = \varphi_{\text{int}} \quad , \quad \frac{\partial \varphi_{\text{ext}}}{\partial \boldsymbol{n}} - \frac{\partial \varphi_{\text{int}}}{\partial \boldsymbol{n}} = -\boldsymbol{n} \cdot \boldsymbol{J}/\mu_0 \quad . \tag{33}$$

Then, the potential is split into two parts $\varphi = \varphi_1 + \varphi_2$. As a result, the problem can be reformulated for these potentials as $\Delta \varphi_1 = \nabla \cdot \boldsymbol{J}/\mu_0$ with the boundary condition $\frac{\partial \varphi_1}{\partial \boldsymbol{n}} = \boldsymbol{n} \cdot \boldsymbol{J}/\mu_0$ on Γ. In addition $\varphi_1(\boldsymbol{x}) = 0$ for $\boldsymbol{x} \notin \Omega$.

As a result, we find $\Delta \varphi_2 = 0$ with $\varphi_2^{\text{ext}} - \varphi_2^{\text{int}} = \varphi_1$ and $\frac{\partial \varphi_2^{\text{ext}}}{\partial \boldsymbol{n}} = \frac{\partial \varphi_2^{\text{int}}}{\partial \boldsymbol{n}}$. It is required that $\varphi_2 \to 0$ for $|\boldsymbol{x}| \to \infty$.

Potential theory tells us that

$$\varphi_2(\boldsymbol{x}) = \frac{1}{4\pi} \int_\Gamma \varphi_1(\boldsymbol{y}) \frac{\partial G(\boldsymbol{x}, \boldsymbol{y})}{\partial \boldsymbol{n}(\boldsymbol{y})} \, da \quad , \tag{34}$$

where $G(\boldsymbol{x}, \boldsymbol{y}) = 1/|\boldsymbol{x} - \boldsymbol{y}|$ is the Green's function.

φ_1 can be easily calculated using the standard finite element method.

The (numerically expensive) evaluation of Eq. (34) in all Ω can be avoided by just calculating the boundary values of φ_2 on Γ and then solving the Dirichlet problem $\Delta \varphi_2 = 0$ with the given boundary values. For $\boldsymbol{x} \to \Gamma$ Eq. (34) is given by

$$\varphi_2(\boldsymbol{x}) = \frac{1}{4\pi} \int_\Gamma \varphi_1(\boldsymbol{y}) \frac{\partial G(\boldsymbol{x}, \boldsymbol{y})}{\partial \boldsymbol{n}(\boldsymbol{y})} \, da + \left(\frac{S(\boldsymbol{x})}{4\pi} - 1 \right) \varphi_1(\boldsymbol{x}) \quad , \tag{35}$$

where $S(\boldsymbol{x})$ denotes the solid angle subtended by Γ at $\boldsymbol{x}$. $S(\boldsymbol{x})$ can be easily calculated at the surface nodes of the finite element mesh by summing up the contributions of each tetrahedron that shares a given node to the solid angle. Upon triangulation of the surface Γ of the domain Ω with triangular elements (which we naturally get from a triangulation of Ω with tetrahedral elements) and discretization of φ_1 and φ_2 we can rewrite Eq. (35) as

$$\boldsymbol{\varphi}_2 = B \boldsymbol{\varphi}_1 \tag{36}$$

with the boundary matrix B, which is a dense matrix with a size of $n_b \times n_b$ elements, where n_b is the number of nodes on the surface Γ. The discretization of the scalar double layer operator in Eq. (35) has been derived by Lindholm [37].

In order to calculate the demagnetizing field, we have to perform the following steps:

Initialization

(1) Discretize $\Delta \varphi_1 = \nabla \cdot \boldsymbol{J}/\mu_0$.
(2) Calculate the boundary matrix in Eq. (36).

Solution

(1) Solve $\Delta \varphi_1 = \nabla \cdot \boldsymbol{J}/\mu_0$ for a given magnetization distribution $\boldsymbol{J}$ using the standard FE method.
(2) Calculate φ_2 on the boundary Γ using Eq. (36) to get the values for the Dirichlet boundary conditions.
(3) Calculate φ_2 in the whole domain Ω using $\Delta \varphi_2 = 0$ with Dirichlet boundary values.
(4) Calculate $\boldsymbol{H}_{\text{demag}} = -\nabla(\varphi_1 + \varphi_2)$.

2.4   Effective Field

Finally we can collect all contributions to the effective field and calculate it by simple matrix-vector multiplications.

$$\boldsymbol{H}_{i,\text{eff}} \approx -\frac{1}{V_i J_{\text{s},i}} \frac{\partial E_{\text{tot}}}{\partial \boldsymbol{u}_i} = (\boldsymbol{H}_{\text{ext}} + \boldsymbol{H}_{\text{demag}})_i - \frac{1}{V_i J_{\text{s},i}} ((G_{\text{exch}} + G_{\text{ani}}) \cdot \boldsymbol{u})_i \quad (37)$$

Since the matrices for the exchange and anisotropy energies depend only on the (time independent) material parameters and the geometry, they need to be calculated only once at the beginning. In order to save time and memory, they can also be assembled into a single combined matrix, if the energies and fields are not required separately.

3   Solution of the Micromagnetic Equations

3.1   Energy Minimization

In Sec. 2 we have introduced the Gibbs free energy of a micromagnetic system, discretized various contributions to the total energy, and derived the matrix-vector formulation. Some complication was introduced by the demagnetizing field, but the hybrid finite-element/boundary element method provides an elegant way to solve the problem accurately with a finite element mesh that is restricted to the magnetic bodies.

This enables us to implement a simple energy minimization scheme to find the equilibrium magnetization distribution. If the magnetic polarization $\boldsymbol{J}$ is defined in Cartesian coordinates, $\boldsymbol{J}(\boldsymbol{x}) \approx \sum_i J_{\text{s},i} \boldsymbol{u}_i \eta_i$ one has to use a constrained solver that ensures, that the norm of $\boldsymbol{J}$ is preserved: $|\boldsymbol{J}| = J_{\text{s}}$. However, it has been shown [38,39] that the use of spherical coordinates has several advantages:

- The number of unknowns is reduced by 1/3, which speeds up the solver.
- The norm is automatically preserved, which allows the use of an unconstrained solver.
- Convergence problems with a Lagrange multiplier approach for the constraint $|\boldsymbol{J}| = J_{\text{s}}$ are avoided.

However, the direct calculation of the energy gradient in spherical coordinates causes various problems due to the periodicity of the polar and azimuth angles. Therefore, the magnetic polarization for the minimizer is given in spherical co-

ordinates. Then it is converted to Cartesian coordinates. The energy gradient is calculated in Cartesian coordinates, converted back to spherical coordinates and returned to the minimizer.

For the minimizer itself the limited memory variable metric (LMVM) algorithm – a quasi-Newton-method – of the TAO package [40,41] has been selected, because it requires only the function values and the gradient of the total energy. In replacement for the Hessian (which is not available due to the demagnetizing field) the second-order information is approximated by a limited history of previous points and gradients. A similar method has been used in [38] and showed better convergence rates than Newton- or Gauß-Seidel methods.

3.2   The Dynamic Equation

The Landau-Lifshitz-Gilbert equation (Eq. (11)) is a system of ordinary differential equations (ODEs) that can be written in a general form as

$$\frac{d\boldsymbol{y}}{dt} = \boldsymbol{f}(t, \boldsymbol{y}) \quad , \quad \boldsymbol{y} \in \mathbb{R}^N \tag{38}$$

with the initial condition

$$\boldsymbol{y}(t_0) = \boldsymbol{y}_0 \quad . \tag{39}$$

The PVODE package [42,43] is a general purpose solver for initial-value problems for stiff and non-stiff ODEs of the form of Eq. (38). It is based on CVODE [44,45] and uses MPI for parallelization and portability.

Two methods are available for the numerical solution of Eq. (38): The backward differentiation formula (BDF), which is recommended for stiff problems, and the Adams-Moulton formula for non-stiff problems, both of which feature a variable stepsize and variable order. Both formulas can be written as the linear multistep formula

$$\sum_{i=0}^{K_1} \alpha_{n,i} y_{n-i} + h_n \sum_{i=0}^{K_2} \beta_{n,i} \frac{dy_{n-i}}{dt} = 0 \quad , \tag{40}$$

where $h_n = t_n - t_{n-1}$ is the stepsize and q the order. The Adams-Moulton formula is obtained with $K_1 = 1$ and $K_2 = q - 1$ with $1 \leq q \leq 12$. The BDF formula is represented by Eq. (40) with $K_1 = q$ and $K_2 = 0$ with $1 \leq q \leq 5$. The numerical integration is started with $q = 1$ and then varied automatically and dynamically.

At each time step an implicit nonlinear system of equations has to be solved. An efficient method for nonstiff problems is functional iteration, because it does not require the solution of a linear system of equations. However, for stiff problems it is better solved by Newton iteration, which does involve the solution of a linear system of equations. PVODE uses a Krylov subspace method – the iterative scaled preconditioned generalized minimal residual method (SPGMR) [46], whose performance can be considerably improved with suitable preconditioners [47]. In addition, this method leads to fewer function evaluations of the Landau-Lifshitz-Gilbert equation and allows larger time steps, which gives an excellent performance of the numerical time integration.

3.3 The Nudged Elastic Band Method

In order to find possible paths of a micromagnetic system through its energy landscape to a local minimum of the total energy, we have implemented the static energy minimization method (cf. Sec. 3.1) and the time integration of the Landau-Lifshitz-Gilbert equation (cf. Sec. 3.2). However, the investigation of thermal stability, which is an important topic especially in the area of magnetic storage devices, requires the calculation of transition rates between stable equilibrium states of the system. The transition rate between two stable equilibria is determined by the lowest energy barrier (saddle point), which separates them. Henkelman and Jónsson proposed the nudged elastic band method to calculate these minimum energy paths [48]. This method has been successfully applied to complex micromagnetic systems [49,50] and it is especially suitable for parallelization.

The path is represented by a sequence of "images" (magnetization distributions) that connects the two given stable equilibrium states $\boldsymbol{M}_i$ and $\boldsymbol{M}_f$. These equilibria may be obtained using the static energy minimization method, for example. The initial path is given by the initial magnetization distribution $\boldsymbol{M}_i$ and the final magnetization distribution $\boldsymbol{M}_f$ and a number of images $\boldsymbol{M}_k$ in between that can be obtained by simple linear interpolation. Then, an optimization algorithm moves the "elastic band of images" through the energy landscape towards the optimal path, which is defined by

$$(\nabla E(\boldsymbol{M}_k) \cdot \boldsymbol{t})\boldsymbol{t} = \nabla E(\boldsymbol{M}_k) \quad , \tag{41}$$

where $\nabla E(\boldsymbol{M}_k)$ denotes the gradient of the total energy at image $\boldsymbol{M}_k$ and $\boldsymbol{t}$ is the unit tangent vector along the path. In order to avoid kinks in the path the tangent is calculated using forward, backward, or second order central differences. Eq. (41) requires the component of the gradient parallel to the tangent to be equal to the gradient. In other words, the optimal path is characterized by fact that the gradient of the total energy is parallel to the tangent for any

13

image $\boldsymbol{M}_k$.

Starting from the initial path an iterative optimization scheme moves the images $\boldsymbol{M}_k$ in a direction $\boldsymbol{D}(\boldsymbol{M}_k)$, which is given by

$$\boldsymbol{D}(\boldsymbol{M}_k) = -\Big(\nabla E(\boldsymbol{M}_k) - (\nabla E(\boldsymbol{M}_k) \cdot \boldsymbol{t})\boldsymbol{t}\Big) \quad . \tag{42}$$

The interpretation of this expression is obvious: The images are moved along the negative gradient of the total energy perpendicular to the tangent. The negative gradient determines the direction towards lower energy while the distance between the images is preserved by taking only the component perpendicular to the tangent. Thus, an ordinary differential equation can be formulated

$$\frac{\partial \boldsymbol{M}_k}{\partial t} = \boldsymbol{D}(\boldsymbol{M}_k) \quad , \tag{43}$$

where t denotes some artificial time parameter. This ODE is integrated using PVODE (cf. Sec. 3.2).

After the discussion of the static energy minimization method and the time integration the Landau-Lifshitz-Gilbert equation, the implementation of the nudged elastic band method has become very simple. The static energy minimization provides us with the gradient of the total energy. For the time integration of Eq. (43) we can use the methods explained in Sec. 3.2 by just replacing the right hand side of Eq. (38) with Eq. (42).

The parallelization of this method can be done by distributing the images. Thus, every processors needs the full set of matrices (but only one copy independent of the number of images), which are required for the calculation of the local fields (and gradients). However, there is no need to partition the finite element mesh any more, because every processor has to do the full calculation for the images. This has the advantage, that no communication is required during the calculation of the gradient of the total energy. Only for the calculation of the tangents the magnetization of some images has to be copied to "neighboring" processors.

4 Implementation

The implementation of the micromagnetics package is based on the freely available open source toolbox "PETSc" – the Portable, Extensible Toolkit for Scientific Computation [51]. It provides the data structures and low level linear
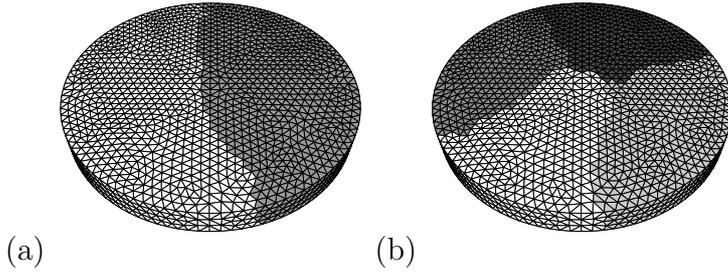
(a)   (b)

Fig. 1. Mesh partitioning of a soft magnetic nanodot for two and four processors. Different shading indicates parts that are assigned to different processors.

algebra and many utility routines for small serial programs as well as large-scale applications for massively parallel supercomputers. Most important, it has a very clear design and simple interfaces for a steep learning curve and many additional features, which speed up the development process and help identifying and avoiding the pitfalls of parallel programs.

In addition, PETSc has interfaces to several other software packages. The 'Toolkit for Advanced Optimization" (TAO) [40] is one of them, and it is used for the static energy minimization in the micromagnetics application (in fact TAO itself is built on PETSc). There are also interfaces to ParMetis for parallel graph partitioning and PVODE [42] the parallel ODE integrator. For reasons to be explained later these built-in interfaces have not been used, but still the integration into PETSc is straight forward. The Metis library is used for mesh partitioning [52], PVODE for the time integration of the LLG equation [42], zlib for compressed data output [53], and libpng for the direct output of graphics files in PNG format [54].

The micromagnetics application can be divided into four parts. First there is a short serial initialization part, which is executed on just one single processor. It reads the configuration files, various input files for the finite element mesh, the initial magnetization, material parameters, etc. If requested, the finite element mesh is refined globally before it is partitioned using the Metis library. Metis uses a very efficient serial multilevel k-way partitioning algorithm [55,56], which gives high quality partitionings.

Each processor then has (approximately) the same number of elements and nodes and a minimum of faces and nodes that are shared with other processors. Fig. 1 shows the partitioning of the finite element mesh of a flat cylinder. The model is cut parallel to the axis of the cylinder, which results in the smallest areas of the cut planes and the smallest number of faces and nodes that are shared by different partitions.

To achieve optimal performance, it is sensible to make the initial serial initialization part as short as possible. However, this may require some additional

data exchange between different processors at a later stage of the initialization phase. Even worse, it increases the amount of "bookkeeping" that is necessary to keep track of the distributed data. This trade off has shown up in the mesh partitioning phase, which can be done in parallel using ParMetis via the built-in interface of PETSc. Yet, it proved to be easier to handle, if it is done during the serial initialization phase on just a single processor. Since the whole initialization process itself usually takes only a fraction of the time of the rest of the simulation, it is not as important anyway.

According to the partitioning scheme suggested by Metis all data are then distributed to their destination processors. Many parameters like the initial time, number of nodes, elements, and faces, material parameters and vertex coordinates are broadcast to all processors. Other data arrays like the vertex to element assignment, element properties, and other element data are delivered just to those processors that "own" the element. However, these data structures make only a minor part of the total required storage space.

All these steps are done only by the first processor, but the serial initialization part is very short and not computation intensive, which is the reason for the excellent scaling of the program initialization. This will be shown in Sec. 5.3.

Then the computation intensive (second) part starts: First, the finite element mesh is analyzed thoroughly. The volumes of all elements and vertices (using the box scheme [31]), element quality factors, and the corresponding extrema and averages are calculated. Next, the stiffness matrix of the Poisson problem and the matrices for the contributions to the local field (Eqs. (21), (28)) are calculated on an element by element basis. However, the results are directly inserted in the global matrices, where the contributions from different elements (even from different processors) are added up by PETSc automatically. This behavior makes the handling of the nodes of the finite element mesh very convenient, because we do not have to manage any duplicate nodes ("ghost points").

The sparsity pattern of the stiffness matrix of the nanodot model on a single processor is shown on the left in Fig. 2. The band structure is achieved by a suitable numbering of the nodes, which in this case has been optimized by the mesh generator Patran using the Gibbs-Poole-Stockmeyer algorithm [57].

For the data output along a sampling line through the model and for the graphics output the interpolation matrices are calculated (cf. Sec. 5.2.2). Then, the requested solver (PVODE for integration of the Landau-Lifshitz-Gilbert equation, TAO for energy minimization, or the solver for the nudged elastic band method) is initialized and all additionally required data structures are created.

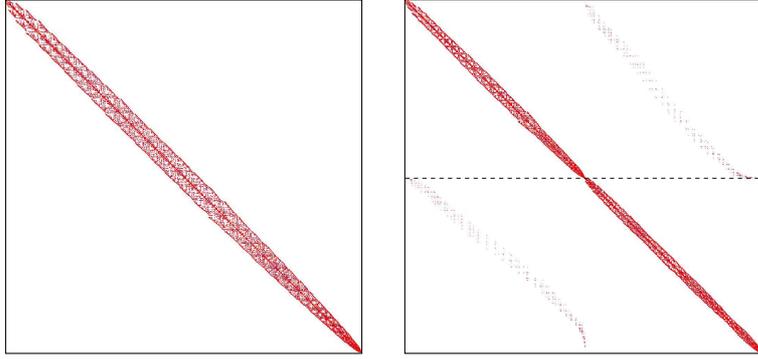The initialization phase is finally completed with the calculation of the local

Fig. 2. Sparsity pattern of the stiffness matrix of the nanodot model for a single processor (left) and distributed over two processors (right). The top half of the matrix is stored on the first processor and the bottom half on the second processor (indicated by the dashed line).

fields and energies of the initial magnetization distribution, first entries in the log files, and the removal of data structures that are not needed any more.

In the third part the solution of the micromagnetic problem is calculated with the requested solver. All solvers require the calculation of the local effective field Eq. (37) with simple matrix-vector multiplications. Since all matrices (except for the boundary matrix Eq. (36)) are sparse matrices, these matrix-vector multiplications with distributed data are very efficient.

For the integration of the LLG equation or the nudged elastic band method PVODE is called to make one time step. The energy minimization with TAO is performed until equilibrium has been reached. Then all required observables are calculated and written to a log file and additional data of the sampling line and graphics output are stored. Finally, several stopping criteria are checked and the simulation is continued with the next iterative step or it is aborted.

When the simulation has been completed, the fourth part of the program is executed. The final results are logged, the solvers are destroyed, all data structures freed and the program exits.

5   Optimization

During the implementation of the micromagnetics application, special attention has been paid to the optimization with regard to speed and storage requirements. The former is strongly determined by good scaling properties of the program on multiple processors. The latter is not an issue for the memory (owing to the distributed data structures), but it can become a problem concerning the size of the data files that are generated and stored on disk during a simulation.

5.1 Profiling

The timing of a serial (single processor, single threaded) program is usually measured as the "real CPU time" (the time which is spent executing commands of the program) by the kernel of the operating system. Thus, idle times during which the program is "sleeping" and waiting for other processes or i/o operations to complete are not taken into account.

On the contrary, the timing of a parallel program has to be done in terms of the elapsed time between its invocation and termination ("wall clock time"). Idle times of one of the different instances of the program indicate a load balancing problem, which may lead to a less than optimal scaling. As a result, careful profiling requires, that all instances have an equal environment on their processor (i.e. no other processes producing a high load should be running) and all timings have to made in terms of "elapsed wall clock time".

5.2 Data I/O

Reading files from and writing them to disk is a quite slow process, but usually the total time spent with i/o operations is negligibly small as compared to the total execution time of the program. More important is the amount of data that have to be stored. Due to the distributed data structures, the size of the finite element models can be increased (almost linearly) with the number of processors, if each of them has the same amount of private memory available. However, the generation of the FE models and the storage of the results to disk can become a problem.

5.2.1 Mesh Refinement

The requirements for a suitable finite element mesh have already been discussed in Sec. 2. In order to reduce the size of the initial finite element mesh, global or partial (adaptive) mesh refinement can be done at run time. Yet, in parallel programs adaptive refinement methods give rise to another difficulty and that is "load balancing" [58]. In order to get the maximum speed up, each processor working for a parallel program should be busy all the time, because delays and idle times reduce the overall performance. Therefore, the problem is initially split up into parts, which will give each processor the same amount of work. To be more accurate, in a heterogeneous environment where machines and processors with different speeds and capabilities are used, the numerical problem has to be distributed in such a way, that all processors finish their tasks at the same time. Then they do not have to wait for each other, when they have to synchronize their results. Adaptive mesh refinement
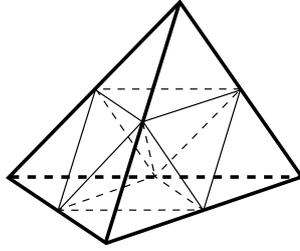
Fig. 3. Global refinement of a "parent" tetrahedral finite element (thick lines) into eight "children" [62].

methods insert and remove finite elements (and therefore nodes) and modify the number of unknowns at run time. Thus, the computational effort will increase for processors working on a partition of the finite element mesh, where elements have been inserted, and decrease for those, where elements have been removed. Of course, this disturbs the initial load balancing and requires a new partitioning of the finite element mesh, redistribution of the current data to the processors and reinitialization before the calculation can be resumed. This scheme has been implemented in the finite element package "UG" [59,60], whose data model is based on "Dynamic Distributed Data" [61], a software tool for distributed memory parallelization.

Nevertheless, global mesh refinement has been implemented, because it allows much smaller input data files and adds the flexibility of increasing the mesh resolution without remeshing the model with a mesh generator. The global refinement scheme is very simple, because it just splits up every finite element into eight "children" (cf. Fig. 3). If this scheme is applied to the whole finite element mesh, its structure remains consistent. The quality of the mesh is preserved, because the four children in the corners of the parent element are congruent to the parent and the other four from the octahedron in the center have equal volumes, too. Even though they are not congruent with the parent, it has been shown, that at most three congruence classes occur [62]. Thus, the stability of the generated triangulations is preserved. With every refinement step the number of finite elements increases by a factor of eight and the number of nodes approximately, too.

As a result, the input data files and especially the file containing the mesh geometry are much smaller. Even more important, the mesh generator does not have to generate a very fine and therefore large mesh, even though it has to be fine enough to resolve the details of the geometrical model and should give finite elements of good quality. Moreover, convergence of the results can be "easily" checked with a refined finite element mesh.
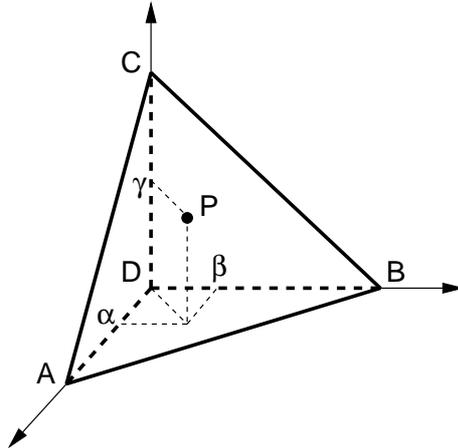
Fig. 4. Barycentric coordinates in a regular tetrahedron.

5.2.2 Data Output

For the direct output of graphics images, the PNG file format has been chosen, because it is widely supported (also by all newer WWW clients), generates very small, highly compressed files, and the reference library libpng [54] is freely available.

First, the magnetization, which is defined on the nodes of the unstructured finite element mesh, has to be interpolated on a regular mesh in the desired slice plane. The slice plane is defined in the configuration file in the form $ax + by + cz = d$ with the normal vector $\boldsymbol{n} = (a|b|c)$. Then all elements that are cut by the slice plane have to be found. In order to simplify this task the whole model (i.e. the Cartesian coordinates of the vertices) is translated and rotated in such a way, that the slice plane coincides with the x-y-plane. Then all elements that are cut by the x-y-plane – this is easily determined using the z-coordinates of the vertices – are tagged, and the maximum dimensions (the "bounding rectangle") of the finite element model in the slice plane are determined. For a given image size we can now calculate the Cartesian coordinates $\boldsymbol{P}$ of each pixel P, which then has to be assigned to one of the tagged elements, within which the pixel is located. This "point location problem", which is a standard problem in geographic information systems and computer-aided design and engineering [63], is the most time consuming part. However, the number of tagged elements (which are cut by the slice plane) is usually far smaller than the total number of elements, and so we can use a brute force yet clever approach that takes advantage of some peculiarities of our specific problem.

In principle, for each pixel we have to search all tagged elements and determine if the pixel is inside or outside. But we can speed up the search by skipping those elements, for which the distance of the pixel from the first node is larger than the maximum edge length of any tetrahedron.

20

If a suitable element is found we calculate the barycentric coordinates $\boldsymbol{B}_P = (\alpha, \beta, \gamma, \delta)$ of the pixel. The barycentric coordinates are defined as the relative position of P inside the tetrahedron $ABCD$. In a "regular" tetrahedron (which has three edges coinciding with the coordinate axes) these correspond to the relative coordinates as indicated in Fig. 4.

$$\boldsymbol{P} = \alpha \boldsymbol{A} + \beta \boldsymbol{B} + \gamma \boldsymbol{C} + \delta \boldsymbol{D} \quad , \quad \delta = 1 - \alpha - \beta - \gamma \tag{44}$$

If we solve for (α, β, γ) we get a simple linear system of equations

$$(\alpha|\beta|\gamma)^T = (\boldsymbol{A} - \boldsymbol{D}|\boldsymbol{B} - \boldsymbol{D}|\boldsymbol{C} - \boldsymbol{D})^{-1} \cdot (\boldsymbol{P} - \boldsymbol{D}) \tag{45}$$

The point P is inside the tetrahedron if and only if $\alpha, \beta, \gamma, \delta \geq 0$, which is equivalent to $\alpha, \beta, \gamma \geq 0$ and $\alpha + \beta + \gamma \leq 1$.

The barycentric coordinates $(\alpha, \beta, \gamma, \delta)$ give the weight, by which the data values at the corresponding vertices of the tetrahedron contribute to the linear interpolation at P. Thus, for some data x_i defined on the vertices of the tetrahedron, we find the linear interpolation at P with $x_P = \alpha x_A + \beta x_B + \gamma x_C + \delta x_D$.

As a result we can assemble an interpolation matrix that calculates the magnetization for all pixels of the image. Let $\boldsymbol{M}_{\mathrm{FE}}$ denote the PETSc vector of one Cartesian component of the magnetization of all nodes of the finite element mesh and $\boldsymbol{M}_{\mathrm{ip}}$ the PETSc vector of interpolated values, then we have $\boldsymbol{M}_{\mathrm{ip}} = A_{\mathrm{ip}} \boldsymbol{M}_{\mathrm{FE}}$ with the $(m \times n)$ interpolation matrix ($m = a \times b$ rows for images with $a \times b$ pixels and n columns for a finite element mesh with n nodes). The matrix elements $A_{ij,\mathrm{ip}}$ are given by the barycentric coordinate of pixel P_i corresponding to node with the global index j of the finite element, in which the pixel is located. This matrix is very sparse as it has only four entries per row (the four barycentric coordinates of a given pixel).

Since the interpolation matrix depends only on the triangulation, we can calculate it during the initialization phase of the program. Whenever a snapshot of one magnetization component should be stored to disk, a simple matrix-vector multiplication gives the interpolated values at the pixel positions. Then the data are color coded. Finally, the PNG library [54] is called to convert these data into a PNG graphics file.

In addition to snapshots of the magnetization it is often desirable to measure some data along a probing line through the model. This feature has been implemented based on the algorithm described above. For the probing line we translate and rotate the model to make the probing line coincide with the x-axis. Then we tag all elements that are cut by the x-axis and determine the barycentric coordinates of every point on the sampling line. We finally end up

with another interpolation matrix that calculates the data values along the probing line. These can then be stored in a data file on disk.

Finally, the complete set of data, which includes the magnetization, the magnetic scalar potential, and the local fields are stored in files in UCD format [64]. This file format can be read by many advanced visualization packages like AVS [65] and MicroAVS [66]. However, the (human-readable) ASCII UCD format generates huge files, because the whole finite element mesh is stored in each file. Thus, two measures have been taken to tackle this problem.

First, the finite element mesh is stored in a single separate file in UCD format during the initialization phase. The data files that are generated during the simulation are then stored without the mesh definition, which saves about 50% of disk space. Secondly, the data files are compressed using the zlib-library [53] in gzip-compatible format, which shrinks the data to about 30% of their initial size. Thus, we end up with data files, which require only 15% of the uncompressed storage space. To restore the data in proper UCD format, a simple shell script uncompresses the data and merges them with the mesh data. Alternatively, there is also a binary UCD format, which gives smaller files than the standard ASCII format, but it is not as widely supported and may lead to compatiblity problems, when the files are transferred to machines with different hardware architecture and operating system.

5.3 Performance

The performance (the speedup in particular) of the micromagnetics application has been measured on a Compaq SC45 cluster consisting of 11 nodes AlphaServer ES45 with 4 Alpha processors (EV68 @ 1 GHz, 8 MB Cache/CPU) and 16 GB of shared memory each. The nodes are interconnected with a Quadrics switch, which provides a maximum MPI bandwidth of 600 MB/s. Since this machine has been shared with several other users, up to 24 processors have been available for speedup measurements.

The speedup has been measured as

$$S_P = \frac{t_1}{t_P} \quad ,$$

where t_1 is the execution time of the program for a given problem on a single processor and t_P is the execution time for the same problem on P processors [67].

The energy minimization method, which uses the LMVM method of the TAO package (cf. Sec. 3.1), has been applied to calculate the nucleation field of FePt nanoparticles. The timing results are summarized in Fig. 5. On 8 and
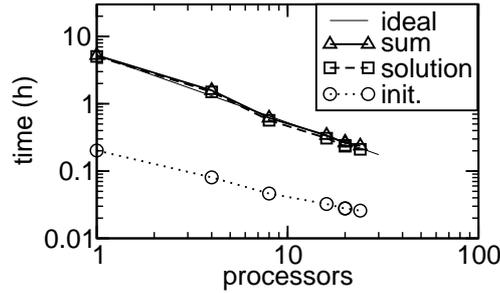
22

Fig. 5. Scaling behavior of the parallel energy minimization algorithm (TAO). The thin solid line shows the ideal (linear) scaling. "init." (open circles) indicates the time required for the serial and parallel initialization. "solution" (open squares) indicates the time required for parallel solution of the problem. "sum" indicates the total simulation time.
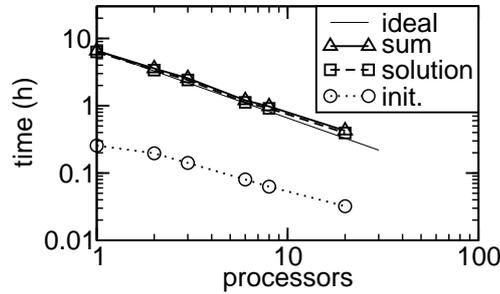

Fig. 6. Scaling behavior of the parallel time integration (PVODE) on an AlphaServer.

16 processors we find a "superlinear" behavior of the solution part of the application. This is a well known phenomenon in parallel computing and can be attributed to caching effects. As the same total amount of data is distributed over more processors, the relative amount decreases and may reach a size, where it fits into the fast cache memory of modern computer architectures. As a result, the data need not be fetched from the main memory (which is a lot slower than the cache memory) and the calculations are completed a lot faster. However, as ever more processors are used, communication requires more and more time, which eventually leads to a saturation of the speedup factor.

The parallel time integration using PVODE is not as efficiently parallelized as the TAO package, which is shown in Fig. 6.

For comparison, Fig. 7 shows the speedup obtained on a Beowulf type cluster of 900 MHz AMD PCs running Linux [68] (for a different micromagnetic problem). These machines are linked with a standard switched 100 MBit Ethernet network.
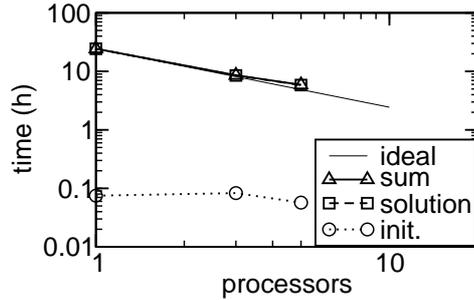
Fig. 7. Scaling behavior of the parallel time integration (PVODE) on a Beowulf type AMD cluster.

6 Permalloy Nanodots

We have studied dynamic precession modes of magnetic vortices in permalloy nanodots, which have also been investigated experimentally by several groups [69–71].

The hysteresis loops of magnetic nanodots have been measured by vibrating sample magnetometer [72] and magneto-optical methods [73,74]. Single domain and vortex states have been successfully identified. Furthermore, these magnetic vortex states are an interesting object for high frequency magnetization dynamics [75] experiments, which are important for high-density magnetic recording media, where high-frequency field pulses of the magnetic write head store the information by reversing the magnetization.

Fig. 8 shows the hysteresis curve for a circular nanomagnet with in-plane external field. We have assumed $J_\mathrm{s} = 1$ T for the saturation polarization, $A = 13 \times 10^{-12}$ J/m for the exchange constant, and zero magnetocrystalline anisotropy. For high in-plane external fields, the magnetization is almost uniform. As the field decreases (solid line in Fig. 8) the magnetization distribution turns into an "onion state", which is caused by the magnetostatic stray field. At the nucleation field (about 23 kA/m in our example) the symmetry of the magnetization distribution breaks and a "C" state develops, which leads to the nucleation of a vortex on the circumference. The vortex immediately moves towards its equilibrium position (close to the center of the nanodot). This dynamic process (vortex precession) is discussed below. As a result we find a sudden drop in the average magnetization. When the external field is reduced to zero the vortex moves into the center of the nanodot (cf. Fig. 9(a)). If the external field is increased in the opposite direction, the vortex is forced out of the center of the dot (cf. Fig. 9(a)). For about −80 kA/m the vortex is pushed out of the nanodot (annihilation) and we find the second jump in the hysteresis curve to (almost) saturation. A detailed study of the static properties of magnetic vortices will be reported in [76].
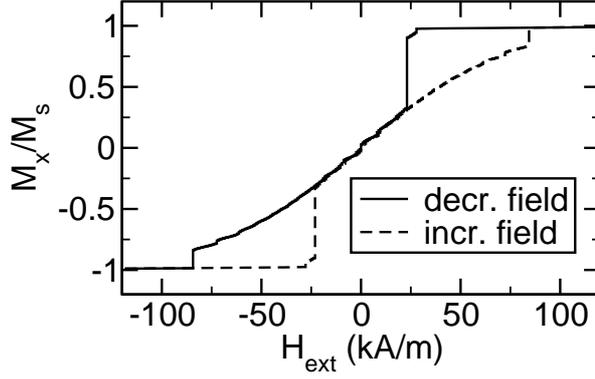
24

Fig. 8. Hysteresis curve of a nanodot with a radius of 200 nm and a thickness of 40 nm.
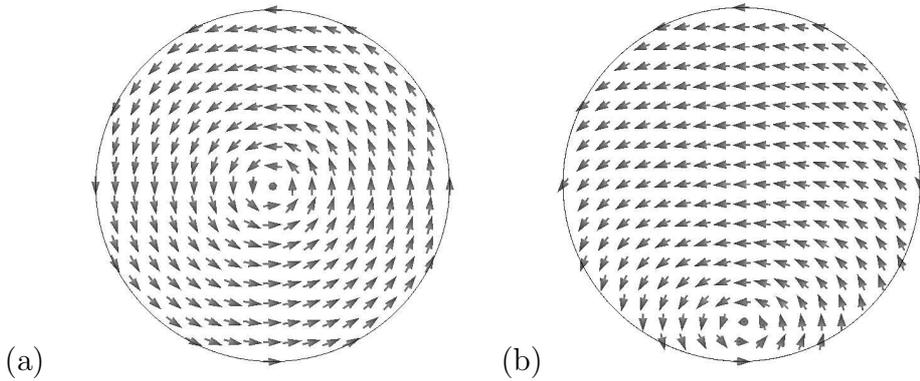

(a)   (b)

Fig. 9. Magnetization distribution of the vortex state in a permalloy nanodot in zero field and just before annihilation.

This characteristic behavior has also been found experimentally using Hall-micromagnetometry by Hengstmann et al. [77], who measured the stray field of individual permalloy disks using a sub-μm Hall magnetometer. The hysteresis loops of arrays of Supermalloy nanomagnets have been measured by Cowburn et al. [73] using the Kerr effect. Their characteristic loop shape has then been used to identify the single-domain in-plane and the vortex phase.

The dynamic behavior of the magnetic vortex has been studied by applying an instantaneous external field pulse of 80 Oe (8 mT, 6.4 kA/m) in the in-plane direction. The time evolution of $\langle M_x \rangle$ (the average of M_x over the whole nanodot) for a dot with an aspect ratio of $L/R = 10$ nm/100 nm $= 0.1$ is given in Fig. 10(a). Then the damped oscillation, which is caused by the spiral motion of the vortex core towards its equilibrium position, is observed. The corresponding Fourier spectrum is given in Fig. 10(b) and shows a sharp peak at a frequency of 0.7 GHz.

Fig. 11 shows the results of the translation mode eigenfrequencies of various nanodots with a radius $R = 100$ nm and a thickness between 10 nm and 40 nm. The results are in good agreement with the results of a finite difference
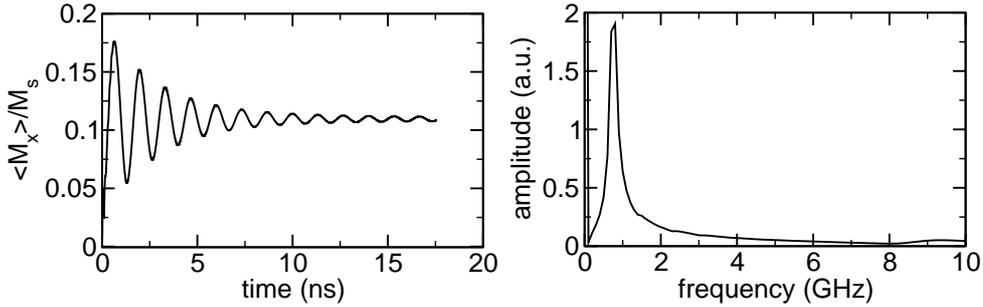
25

Fig. 10. Simulation results for vortex precession in a nanodot with $L/R = 20$ nm$/100$ nm $= 0.2$ under applied in-plane field $\mu_0 H_x = 80$ Oe. Left: Oscillation of $\langle M_x \rangle$ as a function of time. Right: Fourier spectrum
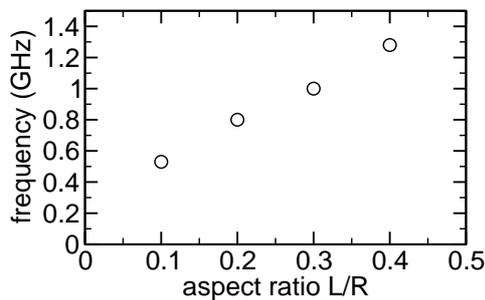

Fig. 11. Translation mode eigenfrequencies versus aspect ratio L/R for nanodots with $R = 100$ nm.

model and the analytical "two-vortices" model presented in Ref. [78]. Direct experimental observation of this mode in an isolated vortex using time-resolved Kerr microscopy has recently been reported by Park et al. [79].

7 Conclusions

A scalable parallel finite element micromagnetics package has been developed. It is entirely based on free open source software packages that have been selected for performance, scalability, portability, and ease of use. The combination of static energy minimization, time integration, and the nudged elastic band method makes it a very efficient and versatile tool. Measurements on a parallel computer and a cluster of PCs showed very efficient parallelization and speedup. The program has been used for the investigation of magnetic vortex structures in permalloy nanodots.

References

[1] T. R. Koehler, D. R. Fredkin, Finite element methods for micromagnetics, IEEE Trans. Magn. 28 (2) (1992) 1239–1244.

[2] M. R. Scheinfein, LLG Micromagnetics Simulator home page, URL: `http://llgmicro.home.mindspring.com/` (2003).

[3] K. Ramstöck, MagFEM3D home page, URL: `http://www.ramstock.de/` (2002).

[4] Euxine Technologies, Magsimus home page, URL: `http://www.euxine.com/` (2003).

[5] D. V. Berkov, N. L. Gorn, MicroMagus home page, URL: `http://www.micromagus.de/` (2002).

[6] J. Oti, PC Micromagnetic Simulator (SimulMag) home page, URL: `http://math.nist.gov/oommf/contrib/simulmag/` (1999).

[7] B. Yang, General Dynamic Micromagnetics (GDM2) home page, URL: `http://physics.ucsd.edu/~drf/pub/` (2003).

[8] M. Donahue, D. Porter, Object Oriented MicroMagnetic Framework (OOMMF) home page, URL: `http://math.nist.gov/oommf/` (2003).

[9] J. G. Zhu, Micromagnetic modelling: theory and application in magnetic thin films, MRS Bulletin 20 (1995) 49–54.

[10] M. Corporation, MSC.Patran home page, URL: `http://www.mscsoftware.com/products/` (2002).

[11] R. Ribó, M. A. de Riera Pasenau, E. Escolano, GiD home page, international Center For Numerical Methods in Engineering (CIMNE), URL: `http://gid.cimne.upc.es` (2002).

[12] A. Hubert, R. Schäfer, Magnetic Domains, Springer, Berlin, 1998.

[13] V. D. Tsiantos, D. Suess, W. Scholz, T. Schrefl, J. Fidler, The effect of the cell size in langevin micromagnetic simulations, J. Magn. Magn. Mater. 242-245 (2002) 999–1001.

[14] W. Rave, K. Ramstöck, A. Hubert, Corners and nucleation in micromagnetics, J. Magn. Magn. Mater. 183 (1998) 329–333.

[15] C. Carstensen, F. Funken, Coupling of nonconforming finite elements and boundary elements II: A posteriori estimates and adaptive mesh-refinement, Computing 62 (3) (1999) 243–259.

[16] K. M. Tako, T. Schrefl, M. A. Wongsam, R. W. Chantrell, Finite element micromagnetic simulations with adaptive mesh refinement, J. Appl. Phys. 81 (8) (1997) 4082–4084.

[17] R. Hertel, H. Kronmüller, Adaptive finite element mesh refinement techniques in 3D micromagnetic modeling, IEEE Trans. Magn. 34 (6) (1998) 3922–3930.

[18] W. Scholz, T. Schrefl, J. Fidler, Mesh refinement in FE-micromagnetics for multi-domain $Nd_2Fe_{14}B$ particles, J. Magn. Magn. Mater. 196-197 (1999) 933–934.

[19] H. Forster, T. Schrefl, D. Suess, W. Scholz, V. Tsiantos, R. Dittrich, J. Fidler, Domain wall motion in nano-wires using moving grids, J. Appl. Phys. 91 (2002) 6914–6919.

[20] H. Forster, Reversal modes in mesoscopic structures, Dissertation, TU Wien (2003).

[21] A. Aharoni, Magnetostatic energy calculation, IEEE Trans. Magn. 27 (4) (1991) 3539–3547.

[22] Q. Chen, A. Konrad, A review of finite element boundary techniqes for static and quasi-static electromagnetic field problems, IEEE Trans. Magn. 33 (1996) 663–676.

[23] B. Streibl, Dynamische mikromagnetische Simulation von μMAG Standardproblemen, Diplomarbeit, TU Wien (1998).

[24] W. F. Brown, Micromagnetics, Interscience, New York, 1963.

[25] A. Aharoni, Introduction to the Theory of Ferromagnetism, Monographs on Physics, Oxford University Press, 1996.

[26] L. Berger, Emission of spin waves by a magnetic multilayer traversed by a current, Phys. Rev. B 54 (13) (1996) 9353–9358.

[27] S. Ingvarsson, L. Ritchie, X. Y. Liu, G. Xiao, J. C. Slonczewski, P. L. Trouilloud, R. H. Koch, Role of electron scattering in the magnetization relaxation of thin $Ni_{81}Fe_{19}$ films, Phys. Rev. B 66 (2002) 214416.

[28] K. Capelle, B. L. Gyorffy, Exploring dynamical magnetism with time-dependent density-functional theory: From spin fluctuations to gilbert damping, Europhys. Lett. 61 (3) (2003) 354–360.

[29] T. Gilbert, Armour Research Foundation, Report No. 11 (1955).

[30] T. L. Gilbert, A Lagrangian formulation of gyromagnetic equation of the magnetization field, Phys. Rev. 100 (1955) 1243.

[31] C. W. Gardiner, Handbook of stochastic methods: for physics, chemistry and the natural sciences, 2nd Edition, Springer, 1996.

[32] J. L. Blue, M. R. Scheinfein, Using multipoles decreases computation time for magnetostatic self-energy, IEEE Trans. Magn. 27 (6) (1991) 4778–4780.

[33] S. W. Yuan, N. Bertram, Fast adaptive algorithms for micromagnetics, IEEE Trans. Magn. 28 (5) (1992) 2031–2036.

[34] C. Seberino, H. N. Bertram, Concise, efficient three-dimensional fast multipole method for micromagnetics, IEEE Trans. Magn. 37 (3) (2001) 1078–1086.

[35] D. R. Fredkin, T. R. Koehler, Hybrid method for computing demagnetizing fields, IEEE Trans. Magn. 26 (2) (1990) 415–417.

[36] T. R. Koehler, Hybrid FEM-BEM method for fast micromagnetic calculations, Physica B: Condensed Matter 233 (4) (1997) 302–307.

[37] D. A. Lindholm, Three-dimensional magnetostatic fields from point-matched integral equations with linearly varying scalar sources, IEEE Trans. Magn. MAG-20 (5) (1984) 2025–2032.

[38] D. Süss, Mikromagnetische Simulation der Keimbildung in SE Magneten, Diplomarbeit, TU Wien (1999).

[39] D. Süss, T. Schrefl, J. Fidler, Micromagnetic simulation of high energy density permanent magnets, IEEE Trans. Magn. 36 (5) (2000) 3282–3284.

[40] S. Benson, L. C. McInnes, J. J. Moré, J. Sarich, TAO home page, URL: http://www-unix.mcs.anl.gov/tao/ (2001).

[41] S. Benson, J. J. Moré, A limited memory variable metric method for bound constrained minimization, Tech. Rep. ANL-95/11 - Revision 2.1.3, Argonne National Laboratory (2002).

[42] G. D. Byrne, A. C. Hindmarsh, PVODE home page, URL: http://www.llnl.gov/CASC/PVODE/ (2002).

[43] G. D. Byrne, A. C. Hindmarsh, PVODE, an ODE solver for parallel computers, Int. J. High Perf. Comput. Appl. 13 (4) (1999) 354–365.

[44] S. D. Cohen, A. C. Hindmarsh, CVODE user guide, lawrence Livermore National Laboratory report UCRL-MA-118618 (1994).

[45] S. D. Cohen, A. C. Hindmarsh, CVODE, a Stiff/Nonstiff ODE Solver in C, Computers in Physics 10 (2) (1996) 138–143.

[46] P. N. Brown, A. C. Hindmarsh, Reduced storage matrix methods in stiff ODE systems, J. Appl. Math. and Comp. 31 (1989) 40–91.

[47] D. Suess, V. Tsiantos, T. Schrefl, J. Fidler, W. Scholz, H. Forster, R. Dittrich, J. J. Miles, Time resolved micromagnetics using a preconditioned finite element method, J. Magn. Magn. Mater. 248 (2002) 298–311.

[48] G. Henkelman, H. Jónsson, Improved tangent estimate in the nudged elastic band method for finding minimum energy paths and saddle points, J. Chem. Phys. 113 (2000) 9978–9985.

[49] R. Dittrich, T. Schrefl, D. Suess, W. Scholz, H. Forster, J. Fidler, A path method for finding energy barriers and minimum energy paths in complex micromagnetic systems, J. Magn. Magn. Mater. 250 (2002) 12–19.

[50] R. Dittrich, Dissertation, TU Wien (2003).

[51] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. Knepley, L. C. McInnes, B. F. Smith, H. Zhang, PETSc home page, URL: `http://www.mcs.anl.gov/petsc/` (2001).

[52] G. Karypis, METIS home page, URL: `http://www-users.cs.umn.edu/~karypis/metis/` (2002).

[53] J. Gailly, M. Adler, zlib home page, URL: `http://www.gzip.org/zlib/` (2002).

[54] G. Randers-Pehrson, libpng home page, URL: `http://www.libpng.org/pub/png/libpng.html` (2002).

[55] G. Karypis, V. Kumar, Multilevel k-way partitioning scheme for irregular graphs, Journal of Parallel and Distributed Computing 48 (1) (1998) 96–129.

[56] Multilevel algorithms for multi-constraint graph partitioning, IEEE Computer Society Press, 1998.

[57] J. G. Lewis, Implementation of the gibbs-poole-stockmeyer and gibbs-king algorithms, ACM Transactions on Mathematical Software 8 (2) (1982) 180–189.

[58] P. Bastian, Load balancing for adaptive multigrid methods, SIAM Journal on Scientific Computing 19 (4) (1998) 1303–1321.

[59] P. Bastian, K. Birken, K. Johannsen, S. Lang, N. Neuss, H. Rentz-Reichert, C. Wieners, UG home page, URL: `http://cox.iwr.uni-heidelberg.de/~ug/` (2003).

[60] P. Bastian, K. Birken, K. Johannsen, S. Lang, N. Neuss, H. Rentz-Reichert, C. Wieners, UG - A flexible software toolbox for solving partial differential equations, Computing and Visualization in Science 1 (1997) 27–40.

[61] K. Birken, DDD home page, URL: `http://cox.iwr.uni-heidelberg.de/~ddd/` (1997).

[62] J. Bey, Tetrahedral grid refinement, Computing 55 (1995) 355–378.

[63] E. P. Mücke, I. A. Saias, B. Zhu, Fast randomized point location without preprocessing in two- and three-dimensional delaunay triangulations, Computational Geometry 12 (1999) 63–83.

[64] Advanced Visual Systems, UCD file format, URL: `http://help.avs.com/Express/doc/help/reference/dvmac/UCD_Form.htm` (2002).

[65] Advanced Visual Systems, AVS home page, URL: `http://www.avs.com` (2002).

[66] Kubota Graphics Technologies Inc., MicroAVS home page, URL: `http://www.kgt.co.jp/index_e.html` (2002).

[67] G. Haase, Parallelisierung numerischer Algorithmen für partielle Differentialgleichungen, Teubner, Stuttgart, Leipzig, 1999.

[68] W. Scholz, T. Schrefl, J. Fidler, Aufbau eines geschützten Subnetzes im TUNET, ZIDline, Informationen des zentralen Informatikdienstes der TU Wien (5) (2001) 29–37.

[69] T. Shinjo, T. Okuno, R. Hassdorf, K. Shigeto, T. Ono, Magnetic vortex core observation in circular dots of permalloy, Science 289 (2000) 930–932.

[70] A. Wachowiak, J. Wiebe, M. Bode, O. Pietzsch, M. Morgenstern, R. Wiesendanger, Direct observation of internal spin structure of magnetic vortex cores, Science 298 (2002) 577–580.

[71] M. Schneider, H. Hoffmann, J. Zweck, Magnetization reversal of thin submicron elliptical permalloy elements, J. Magn. Magn. Mater. 257 (2003) 1–10.

[72] C. A. Ross, M. Hwang, M. Shima, J. Y. Cheng, M. Farhoud, T. A. Savas, H. I. Smith, W. Schwarzacher, F. M. Ross, M. Redjdal, F. B. Humphrey, Micromagnetic behavior of electrodeposited cylinder arrays, Phys. Rev. B 65 (2002) 144417.

[73] R. P. Cowburn, D. K. Koltsov, A. O. Adeyeye, M. E. Welland, D. M. Tricker, Single-domain circular nanomagnets, Phys. Rev. Lett. 83 (5) (1999) 1042–1045.

[74] G. Gubbiotti, G. Carlotti, F. Nizzoli, R. Zivieri, T. Okuna, T. Shinjo, Magnetic properties of submicron circular permalloy dots, IEEE Trans. Magn. 38 (5) (2002) 2532–2534.

[75] V. Novosad, M. Grimsditch, K. Y. Guslienko, P. Vavassori, Y. Otani, S. D. Bader, Spin excitations of magnetic vortices in ferromagnetic nanodots, Phys. Rev. B 66 (2002) 052407.

[76] W. Scholz, K. Y. Guslienko, V. Novosad, D. Suess, T. Schrefl, R. W. Chantrell, J. Fidler, Transition from single-domain to vortex state in soft magnetic cylindrical nanodots, J. Magn. Magn. Mater. In press.

[77] T. M. Hengstmann, D. Grundler, C. Heyn, D. Heitmann, Stray-field investigation in permalloy nanodisks, J. Appl. Phys. 90 (12) (2001) 6542–6544.

[78] K. Y. Guslienko, B. A. Ivanov, V. Novosad, Y. Otani, H. Shima, K. Fukamichi, Eigenfrequencies of vortex state excitations in magnetic submicron-size disks, J. Appl. Phys. 91 (10) (2002) 8037–8039.

[79] J. P. Park, P. Eames, D. M. Engebretson, J. Berezovsky, P. A. Crowell, Imaging of spin dynamics in closure domain and vortex structures, Phys. Rev. B 67 (2003) 020403R.