Die approbierte Originalversion dieser Dissertation ist in der Hauptbibliothek der Technischen Universität Wien aufgestellt und zugänglich.
http://www.ub.tuwien.ac.at

The approved original version of this thesis is available at the main library of the Vienna University of Technology.
http://www.ub.tuwien.ac.at/eng

TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

DISSERTATION

Tensor Grid Methods for Micromagnetic Simulations

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Doktors der technischen Wissenschaften

unter der Leitung von
Univ.Doz. Dipl.Ing. Dr.techn. Thomas Schrefl
E138 - Institut für Festkörperphysik

eingereicht an der Technischen Universität Wien
Fakultät für Physik

durch

Lukas Exl
Friedrich-Kaiser-Gasse 26, 1160 Wien
Matrikelnummer 0425472

Wien, 12. Januar 2014

Erklärung zur Verfassung der Arbeit

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Lukas Exl
Wien, Januar 2014

Abstract

This thesis is dedicated to *computational micromagnetics*, where several new numerical methods are developed. All concepts rely, at some point, on representation of the macroscopic magnetization on *tensor grids*. In this context, the data-sparse representation or approximation via *tensor formats* serves as a key motivation.

Much attention is paid to the computation of the *stray field*. A novel method determines the demagnetizing field on a tensor grid with help of data-sparse tensor format representation of magnetization components. *Kronecker product structure* of the demagnetizing field operator is shown. Also, the Hessian of the discretized total magnetic Gibbs free energy permits a Kronecker product form. This allows cheap and efficient evaluation of the energy and computation of the gradient for tensor structured input. Furthermore, the described method is even accelerated with help of *fast Fourier transform.*

A detailed overview of *micromagnetic energy minimization* is given, including a new method that is a variation of steepest descent. On this basis, energy minimization with structured tensor magnetization is considered. A *sublinearly* scaling low-rank algorithm is introduced, which relies on successive rank-k updates. The approach addresses the computation of equilibrium states and hysteresis of large ferromagnetic particles on rectangular grids.

In order to address micromagnetic simulations on *unstructured finite element meshes*, a further novel demagnetizing field method, based on *non-uniform fast Fourier transform*, is developed.

Kurzfassung

In dieser Arbeit werden einige neue numerische Methoden für *mikromagnetische Simulationen* entwickelt. Die vorgestellten Konzepte basieren alle auf der Darstellung der makroskopischen Magnetisierung auf *Tensorgittern*. Dabei dient die datenschwache Darstellung oder Approximation mittels *Tensor Formate* als Hauptmotiv.

Viel Aufmerksamkeit ist der Berechnung des *Streufeldes* gewidmet. Eine neuartige Methode bestimmt das Demagnetisierungsfeld auf Tensorgittern mit datenschwacher Tensor Format Darstellung der Magnetisierungskomponenten. *Kronecker Produkt Struktur* des Streufeldoperators und der Hessematrix der totalen magnetischen freien Gibbs Energie wird gezeigt. Dies erlaubt die billige und effiziente Auswertung der Energie und des Gradienten für tensorstrukturierten Input. Desweiteren wird die beschriebene Methode mit Hilfe von *Schneller Fourier Transformation* beschleunigt.

Ein detailierter Überblick über *mikromagnetische Energieminimierung* wird gegeben, welcher auch eine neue Variante der Methode des steilsten Abstiegs enthält. Darauf aufbauend wird Energieminimierung mit niedrig-rang Magnetisierung betrachtet. Ein sublinear skalierender Algorithmus wird vorgestellt, welcher auf Rang-k Updates beruht. Der Zugang adressiert die Berechnung von Gleichgewichtszuständen und Hysterese von großen ferromagnetischen Teilchen auf Tensorgittern.

Um mikromagnetische Simulationen auf unstrukturierten finite Elemente Gittern zu adressieren, wird ein weiterer Streufeldalgorithmus vorgestellt, welcher auf *nicht-uniformer Schneller Fourier Transformation* beruht.

Contents

1 Introduction **9**

2 Micromagnetic Background **11**
- 2.1 The Magnetic Gibbs Free Energy . . . 12
  - 2.1.1 Exchange Energy . . . 12
  - 2.1.2 Anisotropy Energy . . . 13
  - 2.1.3 Zeeman Energy . . . 14
  - 2.1.4 Stray Field and Energy . . . 14
- 2.2 Reduced Energy and Units . . . 15
- 2.3 Stray Field Computation . . . 16
  - 2.3.1 Problem Formulation . . . 17
  - 2.3.2 Overview of Existing Methods . . . 18
- 2.4 Brown's Equilibrium Equations . . . 19

3 Methods for Energy Minimization **20**
- 3.1 Problem Formulation in the Continuous Setting . . . 21
- 3.2 Problem Formulation in the Discrete Setting . . . 22
  - 3.2.1 Discretization of the Energy and Gradients . . . 24
- 3.3 Steepest Descent Method and Variations . . . 25
  - 3.3.1 A Semi-Implicit Scheme . . . 27
  - 3.3.2 Limited Memory Quasi Newton Scheme . . . 28
  - 3.3.3 Comparison . . . 29
- 3.4 Miscellaneous Approaches for Micromagnetic Energy Minimization . . . 31
  - 3.4.1 Penalty Methods . . . 32
  - 3.4.2 Newton's Method on the KKT Conditions . . . 34

4 Tensor Formats **37**
- 4.1 Motivation . . . 37
- 4.2 Canonical Tensors . . . 40

4.3 Tucker Tensors and Quasi-Best Approximation . . . 42
4.4 Relation between Tucker Tensors and Tensor Trains (TT) in 3 Dimensions . . . 46
4.5 Discrete Fourier Transform for Structured Tensors . . . 47
4.6 Approximation of Disturbed Data . . . 49

5 Stray Field Computation on Tensor Grids 51
5.1 Description of the Method . . . 51
  5.1.1 Separable Representation of Multivariate Functions . . . 52
  5.1.2 Analytical Preparations . . . 52
  5.1.3 τ-Integration . . . 53
  5.1.4 Description of the Magnetization on a Tensor Grid . . . 56
  5.1.5 Computation on a Tensor Grid . . . 57
5.2 Accuracy Tests . . . 62
5.3 Mathematical Analysis of the Method . . . 65
  5.3.1 Notation . . . 65
  5.3.2 The Collocation Scheme . . . 65
  5.3.3 Sinc Quadrature . . . 68
  5.3.4 Separable Approximation . . . 69
  5.3.5 Some Practical Issues . . . 73
5.4 FFT Acceleration . . . 74

6 Approximation of Magnetization in the Tucker format 78
6.1 Approximation of the Indicator Function of a Sphere . . . 79
6.2 Approximation for zero External Field . . . 80
6.3 Approximation during Demagnetization . . . 81
6.4 Conclusions . . . 87

7 Energy Minimization with Structured Tensors 89
7.1 Discretization of the Energy on a Tensor Grid . . . 90
7.2 Discrete Energy Minimization Problem on Tensor Spaces and Subsets . . . 92
7.3 Minimization with Truncated Iteration . . . 95
  7.3.1 The Constraint on the Magnitude as Optimization Problem . . . 95
7.4 Minimization within the Representation . . . 96
  7.4.1 Reformulation for Fixed Format . . . 96
  7.4.2 Derivatives involving Tensor Formats . . . 97
  7.4.3 Minimization with fixed CP rank . . . 100
7.5 Successive Rank-k Update . . . 102
7.6 Summary and Conclusions . . . 108

8 Non-Uniform FFT for the Finite Element Computation of the Scalar Potential 109
- 8.1 Introduction and Motivation . . . 109
- 8.2 Ansatz of García-Cervera and Roma . . . 111
- 8.3 FEM for the Dirichlet problem . . . 113
- 8.4 Single-Layer Potential . . . 114
- 8.5 Non-Uniform FFT for the Single Layer Potential . . . 117
  - 8.5.1 Computation of the Gridding Tensor . . . 119
  - 8.5.2 Window Functions . . . 122
  - 8.5.3 Kernel Approximation . . . 122
  - 8.5.4 Near field correction . . . 125
- 8.6 Numerics . . . 126
- 8.7 A Closer Look at Errors and Complexity . . . 130
- 8.8 Computing the Stray Field . . . 133
- 8.9 Summary and Discussion . . . 135

9 Conclusions 139

List of Algorithms 141

List of Figures 142

List of Tables 145

Bibliography 149

Acknowledgments 159

Selected Publications 161

Conferences, Workshops, Talks and Posters 162

Curriculum Vitae 164

Chapter 1

Introduction

Many applications which we use in our everyday life rely on magnetism, for instance *electric motors and generators*, *hard-disk drives*, or *wind turbines*. The last one is a typical application of *permanent magnets*, where the most common consist of rare-earth elements, like *samarium-cobalt* or *neodymium-iron-boron* magnets. Energy application and the quest for rare-earth free or rare-earth reduced permanent magnets [1, 2] renewed the interest in micromagnetics of permanent magnets [3]. The continuum theory of micromagnetism intends to bridge the gap between the phenomenological *Maxwell theory* (macroscopic dimensions) of electromagnetism and *quantum theory* (atomistic level). It yields reliable models on the micron scale. Due to the rise of computational power in the last decades, the field of computational micromagnetics evolved as a powerful tool for solving the non-linear micromagnetic equations [4]. Moreover, micromagnetics gives good approximations on length scales where magnetic ab-initio calculations would simply be too fundamental and expensive. Important quantities of permanent magnets, like the coercive field and remanence, can be determined by calculating hysteresis. Hysteresis in non-linear ferromagnetic materials results from the path formed by subsequent local minima. For instance, Kinderlehrer and Ma [5] computed hysteresis loops in ferromagnets from the continuation of solutions for decreasing and increasing applied fields. Most state-of-the-art micromagnetic solvers implement the Landau-Lifshitz Gilbert (LLG) equation of motion. However, the accessible time scale of LLG simulations is the range of nanoseconds. This time scale is not always relevant, e.g. the measurement time for hysteresis loops of permanent magnets is in the range of seconds. Thus, *energy minimization* can be applied for such applications where the extra information due to time evolution is not important.

In this work the focus is on the development and analysis of methods to address the non-linear and non-convex micromagnetic constrained energy minimization problem on so-called tensor grids.

In the last few years much research was done on problems and applications where tensors appear as data or solutions. Some ideas were developed to calculate so-called low-rank solutions

[6]. In this thesis some of the existing ideas and also novel approaches are applied to computational micromagnetics.

The thesis starts with background information on the theory of micromagnetics in Ch. 2. The *micromagnetic energy minimization problem* is addressed in Ch. 3, where it is formulated as constrained optimization problem in the continuous and the discrete setting. Several algorithms which address this problem are analyzed. A new variant of the steepest descent method, a *semi-implicit scheme*, is also introduced. Moreover, *penalty approaches* from non-linear programming are applied to micromagnetics, as well as, *Newton's method to the Karush-Kuhn-Tucker (NKKT) conditions*. In order to minimize the energy on large tensor grids, the data-sparse tensor formats are introduced. A detailed description of tensor formats and approximation of tensors is given in Ch. 4. Incidentally, a FFT-based method to apply filtering of disturbed multi-way data is found.

A tensor grid method for computing the stray field for tensor structured input is described and mathematically analyzed in Ch. 5. Kronecker product structure of the demagnetizing operator is proven, which later gives rise to a similar structure for the Hessian of a second order discretization of the total magnetic energy. This structure allows the efficient evaluation for tensor structured input. Later, the tensor grid stray field method is even accelerated by means of FFT, cf. Sec. 5.4.

Chapter 6 is dedicated to approximation of magnetization configurations by the so-called Tucker format with no applied field and during demagnetization. The Tucker format allows adaptive rank determination by an algorithm based on singular value decomposition, cf. Ch. 4. The micromagnetic energy minimization problem subject to low-rank tensors is investigated and analyzed in Ch. 7. An algorithm is introduced, which is based on low-rank updates and minimization within the canonical tensor representation. In principal, this method allows applying large grids due to the sublinear scaling in the volume size. This is useful for large ferromagnetic particles which demand a high resolution due to constraints related to the exchange length or domain wall width.

The final section is dedicated to a novel finite element/boundary element (FEM/BEM) algorithm, which benefits from *non-uniform FFT*, which is especially adapted to the case of boundary integrals. The method scales quasi optimal in the number of volume and surface elements.

Chapter 2

Micromagnetic Background

The beginning of micromagnetism was a paper by *Landau* and *Lifshitz* in 1935 on the structure of the wall between two antiparallel domains and the works of *W.F. Brown, Jr.* in 1940-41, who also gave the name *micromagnetism* or *micromagnetics* [7]. This theory intended to bridge the gap between the phenomenological *Maxwell theory* (macroscopic dimensions) of electromagnetism and *quantum theory* (atomistic level) [4]. However, the term is somewhat misleading, because the microscopic details of the atomic structure are ignored, and the material is considered from a macroscopic point of view by taking it to be continuous [8]. The classical approach which leads to micromagnetics is to replace the magnetic moments by a classical vector field as well as the quantum-mechanical exchange interaction (by *Heisenberg, Dirac* 1928) [8]. There were several papers by *Stoner-Wohlfarth, Néel, Aharoni, Strikman, Treves and Brown* in the 1950s, who established micromagnetics as an efficient theory to describe magnetization processes and characteristic properties of the hysteresis loop, see [4] and references therein. The recent developments of numerical micromagnetics allows the solution to the nonlinear micromagnetic equations also for non-classical problems, see [9] and references therein.

In micromagnetics a state is fully described if for given temperature, applied field and elastic stresses the *(spontaneous) polarization* $\boldsymbol{J}(\boldsymbol{x}) = \mu_0 \boldsymbol{M}(\boldsymbol{x})$ (*(spontaneous) magnetization* $\boldsymbol{M}(\boldsymbol{x})$) is known, where μ_0 is the *vacuum permeability*. The magnetization can be seen as the (average) magnetic moment density [10] or magnetic moment per unit volume [11]. Its magnitude is fixed, i.e. $\|\boldsymbol{M}(\boldsymbol{x})\| =: M_s$ such that the state is described by the direction cosines $\boldsymbol{m}(\boldsymbol{x})$, i.e. $\boldsymbol{M}(\boldsymbol{x}) = M_s\, \boldsymbol{m}(\boldsymbol{x})$, where $\|\boldsymbol{m}(\boldsymbol{x})\| = 1$. See Sec. 2.2 concerning units.

Magnetization dynamics is not treated in the following work, but static energy minimization. However, for the sake of completeness it is shortly mentioned here that the equation of motion, which is also widely used in micromagnetics, is the *Landau-Lifshitz-Gilbert (LLG) equation*, i.e.

$$\partial_t \boldsymbol{m} = -\gamma' \boldsymbol{m} \times \boldsymbol{H}_{eff} - \alpha' \boldsymbol{m} \times (\boldsymbol{m} \times \boldsymbol{H}_{eff}), \tag{2.1}$$

where $\boldsymbol{H}_{eff}$ is the so-called effective field (cf. Sec. 2.4), γ' and α' can be expressed in terms of the electron gyromagnetic ratio and the Gilbert damping parameter [4]. The first term on the right hand side of Eqn. (2.1) is a torque (or precessional) term, whereas the second term causes damping towards the effective field. In Ch. 3.3 the connection between energy minimization with steepest descent directions and LLG without precessional term is discussed.
In the following the components of the *magnetic Gibbs free energy* are described.

2.1 The Magnetic Gibbs Free Energy

The *total magnetic Gibbs free energy* is the sum of exchange, anisotropy, stray field and external energy. Magnetostrictive energy is not considered here, see e.g. [4]. Equilibrium configurations are local minima of this total energy. The exchange and anisotropy term in the energy have a quantum mechanical origin and hence, their continuum formulation in micromagnetics is phenomenological. The magnetostatic terms (stray field and Zeeman energy) have classical descriptions and are derived from electromagnetism.

2.1.1 Exchange Energy

When the quantum mechanical spin operators are approximated by classical vectors $\boldsymbol{S}_i = S\,\boldsymbol{s}_i$ with $\|\boldsymbol{s}_i\| = 1$ the *'classical' exchange energy* turns out to be

$$\mathcal{E}_{ex} = -\sum_{\substack{i,j\\ i\neq j}} J_{ij}\, S^2\, \boldsymbol{s}_i^T \boldsymbol{s}_j = -2JS^2 \sum_i \sum_{j\in\text{next-neighbor}(i)} \cos\sphericalangle(\boldsymbol{s}_i, \boldsymbol{s}_j), \tag{2.2}$$

where J_{ij} is the *exchange integral*, which is assumed to be nonzero only for neighbors (and therefore denoted as J) [8]. This expression is now taken for the continuous magnetization $\boldsymbol{m}$ and further approximation is done, i.e.

$$\cos\sphericalangle(\boldsymbol{m}(\boldsymbol{x}_i), \boldsymbol{m}(\boldsymbol{x}_j)) = 1 - \tfrac{1}{2}\|\boldsymbol{m}(\boldsymbol{x}_i) - \boldsymbol{m}(\boldsymbol{x}_j)\|^2 + O(\|\boldsymbol{m}(\boldsymbol{x}_i) - \boldsymbol{m}(\boldsymbol{x}_j)\|^4), \tag{2.3}$$

where $\cos x = 1 - x^2/2 + O(x^4)$ was used for small angles $\sphericalangle(\boldsymbol{s}_i, \boldsymbol{s}_j) = \|\boldsymbol{m}(\boldsymbol{x}_i) - \boldsymbol{m}(\boldsymbol{x}_j)\| + O(\|\boldsymbol{m}(\boldsymbol{x}_i) - \boldsymbol{m}(\boldsymbol{x}_j)\|^3)$. Now the *first* order expansion in the position $\boldsymbol{x}$

$$\boldsymbol{m}(\boldsymbol{x}_j) \approx \boldsymbol{m}(\boldsymbol{x}_i) + J_{\boldsymbol{m}}(\boldsymbol{x}_i)(\boldsymbol{x}_j - \boldsymbol{x}_i), \tag{2.4}$$

with the Jacobian $J_{\boldsymbol{m}}(\boldsymbol{x}_i)$ yields

$$\|\boldsymbol{m}(\boldsymbol{x}_i) - \boldsymbol{m}(\boldsymbol{x}_j)\|^2 \approx \sum_{p=1}^{3} \left((\boldsymbol{x}_j - \boldsymbol{x}_i)^T \nabla m^{(p)}(\boldsymbol{x}_i) \right)^2 . \tag{2.5}$$

In analogy to (2.2) this all together gives an expression for the *'(classical) micromagnetic' exchange energy*

$$\mathcal{E}_{ex} = \mathcal{E}_{ex}^{ref} + JS^2 \sum_{i} \sum_{j \in \text{next-neighbor}(i)} \sum_{p=1}^{3} \left((\boldsymbol{x}_j - \boldsymbol{x}_i)^T \nabla m^{(p)}(\boldsymbol{x}_i) \right)^2 , \tag{2.6}$$

where $\mathcal{E}_{ex}^{ref}$ is the energy of the reference state where the magnetization is uniform. Usually this term is omitted, which means that the exchange energy is measured with respect to the uniform state. Also note that constants in energies are not interesting for determining equilibrium.

The continuous expression for the exchange energy of a cubic and isotropic ferromagnetic body Ω (which is usually taken in micromagnetics) is

$$E_{ex} = A \int_{\Omega} \sum_{p=1}^{3} \|\nabla m^{(p)}(\boldsymbol{x})\|^2 \, \mathrm{d}\boldsymbol{x}, \tag{2.7}$$

where A is called the exchange constant, which is determined experimentally for different materials. Eqn. (2.7) can be derived from (2.6) by assuming a cubic lattice with constant lattice spacings and changing the sum over i in (2.6) into an integral [8].

The energy (2.7) is minimized for uniform magnetization and, in the general case, is lower when the configuration varies only slowly.

2.1.2 Anisotropy Energy

The crystal structure of a ferromagnetic material causes certain preferred axes (*easy axes*) for the magnetization. The easy axes are *not* directed, hence an expression for the anisotropy energy has to give the same value for the configurations $\pm\boldsymbol{m}$ (even function). For uniaxial (only one easy axis) magnetic materials (e.g. $Nd_2Fe_{14}B$) a general expression of the anisotropy energy is given as

$$E_{an} = \int_{\Omega} K_1 \sin^2\theta + K_2 \sin^4\theta + \dots , \tag{2.8}$$

where θ is the angle between $\boldsymbol{m}$ and the easy direction $\boldsymbol{a}$ (unit vector) and the $K_1, K_2, \dots$ are called magnetocrystalline anisotropy constants. In the following only the second order expansion

$$E_{an} = \int_\Omega K_1(1 - \cos^2\theta) = \int_\Omega K_1(1 - (\boldsymbol{a}\cdot\boldsymbol{m})^2)\,\mathrm{d}\boldsymbol{x}, \tag{2.9}$$

is used.

2.1.3 Zeeman Energy

The energy of a ferromagnetic body in an applied/external field $\boldsymbol{H}_{ext}$, also called *Zeeman energy*, is given as

$$E_{ext} = -\int_\Omega \boldsymbol{J}\cdot\boldsymbol{H}_{ext}\,\mathrm{d}\boldsymbol{x}. \tag{2.10}$$

It is minimized if the magnetization is aligned along the external field. .

2.1.4 Stray Field and Energy

The stray field $\boldsymbol{H}_d$ can be derived from the magnetostatic Maxwell equations without current [10], i.e.

$$\begin{aligned} \nabla\cdot(\boldsymbol{H}_d + \boldsymbol{M}) &= 0 \\ \nabla\times\boldsymbol{H}_d &= \boldsymbol{0}. \end{aligned} \tag{2.11}$$

Let the magnetization $\boldsymbol{M}$ be defined inside a magnetic body Ω. From interface conditions [10, 11] one recognizes that the normal component of $\boldsymbol{H}_d$ at the boundary of Ω has a jump, whereas the tangential component is continuous, i.e. there holds on the boundary

$$\begin{aligned} (\boldsymbol{H}_d^{ext} - \boldsymbol{H}_d^{int})\cdot\boldsymbol{n} &= \boldsymbol{M}\cdot\boldsymbol{n} \\ (\boldsymbol{H}_d^{ext} - \boldsymbol{H}_d^{int})\times\boldsymbol{n} &= \boldsymbol{0}, \end{aligned} \tag{2.12}$$

where $\boldsymbol{n}$ is the outer unit normal. Moreover, the second equation in (2.11) gives rise to a scalar potential ϕ with $\boldsymbol{H}_d = -\nabla\phi$ and therefore (2.11) reduces to a set of equations for ϕ, see Sec. 2.3.1. The solution of this equations has the integral representation (2.23) given in Sec. 2.3.1. The stray field as a function of the magnetization is linear.
The stray field energy is

$$E_d(\boldsymbol{M}) = -\frac{\mu_0}{2}\int_\Omega \boldsymbol{M}\cdot\boldsymbol{H}_d(\boldsymbol{M})\,d\Omega. \tag{2.13}$$

This energy is lower the more the magnetization avoids volume and surface charges, cf. Sec. 2.4 and (2.23).

2.2 Reduced Energy and Units

The *magnetization* is denoted with $\boldsymbol{M} = M_s\boldsymbol{m}$, $\|\boldsymbol{m}\| = 1$, where M_s is the *saturation magnetization* and $\boldsymbol{m}$ the unit vector of the magnetization. In the following $\boldsymbol{m}$ is often also called magnetization where this does not lead to any confusion. In SI-units one has $[\boldsymbol{M}] = [M_s] = A/m$.
The *magnetic polarization* is $\boldsymbol{J} = \mu_0\boldsymbol{M} = \mu_0 M_s\boldsymbol{m} = J_s\boldsymbol{m}$, with SI-units $[J_s] = [\mu_0]A/m = J/(A^2m)A/m = J/(Am^2) = \text{Tesla}$, where $\mu_0 = 1.256637\text{ e-}06\, Tm/A$ is the *vacuum permeability*.
The *demagnetizing field* (stray field) $\boldsymbol{H}_d$ is a linear function of the magnetization $\boldsymbol{M}$, its units are A/m. It is convenient for future purpose to define the *scaled field* $\boldsymbol{h}_d = \boldsymbol{h}_d(\boldsymbol{m}) = \boldsymbol{H}_d(\boldsymbol{M}/M_s) = \boldsymbol{H}_d/M_s$ (dimensionless).
For the description of the total magnetic energy one may introduce the *exchange constant* A in units of J/m, and K_1, the *first magnetocrystalline anisotropy constant*, in units of J/m^3.
Let further $\Omega \subset \mathbb{R}^3$ be an open subset of the three-dimensional space.
The *magnetic Gibbs free energy* is the sum of *exchange-, demagnetizing-, (uniaxial/first order) anisotropy-* and *external energy*, given by

$$\begin{aligned} E_{tot}(\boldsymbol{m}) &= E_{ex}(\boldsymbol{m}) + E_d(\boldsymbol{m}) + E_{an}(\boldsymbol{m}) + E_{ext}(\boldsymbol{m}) \\ &= \int_\Omega A\left(\sum_{p=1}^{3}\left\|\nabla m^{(p)}\right\|^2\right) - \frac{1}{2}\boldsymbol{J}\cdot\boldsymbol{H}_d(\boldsymbol{M}) + K_1(1-(\boldsymbol{a}\cdot\boldsymbol{m})^2) - \boldsymbol{J}\cdot\boldsymbol{H}_{ext}\; d\Omega, \end{aligned} \tag{2.14}$$

where $\boldsymbol{a}$ is the unit vector parallel to the easy axis and $\boldsymbol{H}_{ext}$ the external field in units of A/m. Clearly, one has $[E_{tot}] = J$. For future purpose it is convenient to define the *scaled energy* $e_{tot}(\boldsymbol{m}) = E_{tot}(\boldsymbol{m})/(\mu_0 M_s^2)$, which then has units of a volume, i.e. $[e_{tot}] = m^3$,

$$e_{tot}(\boldsymbol{m}) = \int_\Omega \frac{A}{\mu_0 M_s^2}\left(\sum_{p=1}^{3}\left\|\nabla m^{(p)}\right\|^2\right) - \frac{1}{2}\boldsymbol{m}\cdot\boldsymbol{h}_d(\boldsymbol{m}) + \frac{K_1}{\mu_0 M_s^2}(1-(\boldsymbol{a}\cdot\boldsymbol{m})^2) - \frac{1}{M_s}\boldsymbol{m}\cdot\boldsymbol{H}_{ext}\; d\Omega. \tag{2.15}$$

One therefore defines the reduced units [12] with help of the energy density $K_m := \mu_0 M_s^2 = J_s^2/\mu_0$ (units of J/m^3):

$$Q := K_1/K_m, \quad \widetilde{A} := A/K_m, \tag{2.16}$$

where lengths are usually given in units of the *exchange length* $l_{\text{ex}} = \sqrt{2A/K_m} = \sqrt{2\widetilde{A}}$, e.g. if a cubic particle has the length L then the *reduced length* $\lambda = L/l_{\text{ex}}$ (dimensionless) is introduced. Below the exchange length twisting of the magnetization is energetically unfavorable, which imposes a constraint on the mesh size, i.e. discretization of the domain Ω should be finer than this critical length [13]. Usually for $Q > 1/2$ the magnetic material is called *hard magnetic*, whereas for $Q \leq 1/2$ one calls it *soft magnetic*. In the hard magnetic case the so-called *domain wall width* or *wall width parameter* $\sqrt{A/K_1} = \sqrt{A/(K_m Q)} \leq l_{ex}$ plays the important role and calls for an even finer discretization [14].

For the purpose of converting reduced units to SI units one has to choose M_s or J_s to calculate K_m.

The scaled energy from Eqn. (2.15) has the dimensionless form

$$\psi_t := \frac{e_{tot}}{|\Omega|} = \frac{1}{|\Omega|}\left(\widetilde{A}\, e_{ex} + e_d + Q\, e_{an} + e_{ext}\right), \tag{2.17}$$

with

$$e_{ex}(\boldsymbol{m}) := \int_\Omega \sum_{p=1}^{3} \left\|\nabla m^{(p)}\right\|^2 = -\int_\Omega \sum_{p=1}^{3} m^{(p)} \Delta m^{(p)}, \tag{2.18}$$

$$e_d(\boldsymbol{m}) := -\frac{1}{2}\int_\Omega \boldsymbol{m} \cdot \boldsymbol{h}_d(\boldsymbol{m}), \tag{2.19}$$

$$e_{an}(\boldsymbol{m}) := \int_\Omega (1 - (\boldsymbol{a} \cdot \boldsymbol{m})^2), \tag{2.20}$$

$$e_{ext}(\boldsymbol{m}) := -\int_\Omega \boldsymbol{m} \cdot \boldsymbol{h}_{ext}, \tag{2.21}$$

where $\boldsymbol{h}_{ext} = \boldsymbol{H}_{ext}/M_s$. The equality in (2.18) (cf. [13]) comes from the vector identity $\left\|\nabla m^{(p)}\right\|^2 = \nabla\cdot(m^{(p)}\nabla m^{(p)}) - m^{(p)}\Delta m^{(p)}$ together with the fact that $\|\boldsymbol{m}\| = 1$ implies $\sum_{p=1}^{3} m^{(p)}\nabla m^{(p)} = \boldsymbol{0}$.

In the special case of a cube with side-length L, i.e. $|\Omega| = L^3$, the constant in the exchange energy can be expressed via the relation $L = \lambda l_{\mathrm{ex}} = \lambda\sqrt{2\widetilde{A}}$. Nevertheless, from a numerical point of view, grid spacings used in finite differences for the exchange energy discretization should have orders of magnitudes near 1, instead of 1e-9 (nanometers). Thus, it is better to scale the domain, e.g. lengths could be measured as fractions of 1. In the case of a cube one would have the relation $1 = \lambda\, l'_{\mathrm{ex}}$, with $l'_{\mathrm{ex}} = l_{\mathrm{ex}}/L$ (dimensionless). In this case, the coefficient in front of e_{ex} is given as $\widetilde{A} = (l'_{\mathrm{ex}})^2/2 = 1/(2\lambda^2)$, whereas Ω is scaled to the unit cube.

2.3 Stray Field Computation

Portions of this section were previously submitted for publication as [15] and have been reproduced here with permission of the co-authors. Content which was not generated by the author of this thesis is explicitly denoted.

2.3.1 Problem Formulation

The micromagnetic *demagnetizing or stray field* is given as $\boldsymbol{h}_d = -\nabla\phi$, where the *scalar potential* ϕ for a given magnetization $\boldsymbol{m} \in (C^1(\Omega))^3 \cap (C^0(\overline{\Omega}))^3$ [1], $\Omega \subset \mathbb{R}^3$ bounded and open, fulfills the interface problem [7, 9–11] (also compare with Sec. 2.1.4)

$$\begin{aligned}
-\Delta\phi &= -\nabla\cdot\boldsymbol{m} && \text{in } \Omega, \\
-\Delta\phi &= 0 && \text{in } \overline{\Omega}^c, \\
[\phi] &= 0 && \text{on } \partial\Omega, \\
\left[\frac{\partial\phi}{\partial\boldsymbol{n}}\right] &= -\boldsymbol{m}\cdot\boldsymbol{n} && \text{on } \partial\Omega, \\
\phi(\boldsymbol{x}) &= \mathcal{O}(\tfrac{1}{\|\boldsymbol{x}\|}) \quad \text{as } \|\boldsymbol{x}\| \to \infty,
\end{aligned} \tag{2.22}$$

where $[f] := f^{ext} - f^{int}$ stands for the jump of a function f at the boundary. The asymptotic decay (regularity at infinity) is stated for the sake of solvability and uniqueness of the solution [8, 11]. The classical solution of (2.22) has to be determined in whole space and is at least two times continuously differentiable in Ω and the exterior region $\overline{\Omega}^c$.

For a so-called *weak solution* we further assume that Ω is a Lipschitz domain with polyhedral boundary $\partial\Omega$. One reformulates the set of equations (2.22), [16]:

For given $\boldsymbol{m} \in (H^1(\Omega))^3$ the *micromagnetic scalar potential* $\phi := (\phi^{int}, \phi^{ext}) \in H^1(\Omega) \times H^1_{loc}(\overline{\Omega}^c)$ is the solution to (2.22), where $-\Delta\phi^{int} = -\nabla\cdot\boldsymbol{m}$ in Ω and $-\Delta\phi^{ext} = 0$ in $\overline{\Omega}^c$ holds in a variational sense.

Hereby, $H^1(\Omega)$ denotes the usual *Sobolev space*, i.e. $H^1(\Omega) := \{u \in L^2(\Omega) \mid \text{weak derivatives } \partial_q u \in L^2(\Omega),\ q = x, y, z\}$ and $H^1_{loc}(\overline{\Omega}^c) := \{u \in H^1(C) \mid C \subset \overline{\Omega}^c \text{ compact}\}$. The jump [.] is determined by taking the corresponding trace operators. Within this setting the existence of a unique solution to (2.22) has been proved. For details the reader is referred to [16, 17] and references therein.

An integral representation of the scalar potential is given by [9, 10]

$$\phi(\boldsymbol{x}) = -\frac{1}{4\pi}\Big(\int_\Omega \frac{\nabla\cdot\boldsymbol{m}(\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|}\,\mathrm{d}\boldsymbol{y} - \int_{\partial\Omega} \frac{\boldsymbol{m}(\boldsymbol{y})\cdot\boldsymbol{n}(\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|}\,\mathrm{d}\sigma(\boldsymbol{y})\Big). \tag{2.23}$$

The expressions $\nabla\cdot\boldsymbol{m}$ and $\boldsymbol{m}\cdot\boldsymbol{n}$ are also called volume and surface charge density, respectively.

---

[1] Here $C^q(\Omega)$ is the space of q times continuously differentiable functions defined on Ω, $\overline{\Omega}$ means closure of Ω and $(.)^c$ stands for complement.

2.3.2 Overview of Existing Methods

In order to compare micromagnetic methods in terms of storage and complexity it is important to mention the possibility to precompute certain quantities that do not depend on the magnetization $\boldsymbol{m}$, e.g. only rely on the geometry of the problem, see for instance [18], where several methods were compared. Thus, in asymptotic operation counts, one neglects the effort for computing these steps and, instead, often refers to as *precomputation* or *setup phase*.

Several methods address the approximation of the solution to the set of equations (2.22) that define the scalar potential. Integral methods with a regular discretization of the domain Ω aim to directly compute the integral representation (2.23) of the solution inside the magnetic body Ω. A widely used method aims to calculate the stray field inside the magnetic region by using the (cell-averaged) gradient of (2.23), also referred to as *demagnetizing tensor method* [13, 19]. Discretization on an equispaced grid built of rectangular computational cells allows applying *fast Fourier transform* (FFT) techniques, making this methods quasi-optimal, i.e. the costs are $\mathcal{O}(N \log N)$ for N grid points [9, 18, 19].

Also the *fast multipole method* and combination with FFT has been applied to compute the magnetostatic field and energy [20–22], as well as a nonuniform grid (NG) algorithm [23].

Within the framework of integral approaches also tensor grid methods were developed, which make further assumptions on the representation of the magnetization field by tensor formats, but then can even gain sub-linear complexity [24–26]. Chapter 5 is dedicated to this topic.

A method that uses non-uniform FFT from [27] on the quadrature approximation of the integral representation (2.23) discretized on unstructured 2-dimensional FE grids has been reported in [28]. To the authors knowledge, this method would scale $\mathcal{O}(Q + N + n^d \log n)$ in the general case of d dimensions for $Q = qL$ quadrature points in total, where q quadrature points are used for each of the L computational domains (tetrahedrons for volume and triangles for surface integrals), N mesh-nodes and an auxiliary parameter n, which comes from the FFT. This auxiliary parameter is of the same order as in the BEM-NFFT method in section 8.5, which basically leads to the same complexity for a prescribed accuracy but requires only a pre-factor $q = 1$. This is achieved by the usage of *integrated window functions*, hence, performing the integration in a setup phase.

Moreover, *shell transformation techniques* on a finite element mesh containing Ω were applied to address unbounded problems like (2.22), [29]. The discrete formulation of (2.22) translates to only one sparse linear system, which, however, tends to be very ill-conditioned due to the transformation. Algebraic multigrid preconditioners were successfully applied to address this issue [18].

On the other hand, the well-known hybrid FEM-BEM coupling by the *ansatz of Fredkin and Koehler* [30] aims to solve (2.22) by the splitting $\phi = \phi_1 + \phi_2$, where ϕ_1 is determined by a Poisson equation with Neumann boundary conditions and ϕ_2 by a Laplace equation where the Dirichlet data are computed by the values of ϕ_1 through a boundary integral representation of the potential ϕ_2. Hereby, the calculation of the boundary values of ϕ_2 leads to a dense matrix-vector product which scales $O(N_b^2)$ for N_b boundary nodes. Compression techniques were introduced to reduce this complexity and storage requirements [31].

In chapter 8 a first order polynomial (P1) finite element method is presented that solves (2.22) by the *ansatz of García-Cervera and Roma* [32], where a fast evaluation technique for the single layer potential is developed [15].

2.4 Brown's Equilibrium Equations

The condition of zero first variation (see Sec. 3.1 for a definition) of the total energy (2.15) subject to the unit norm constraint leads to necessary conditions for a magnetization configuration in equilibrium [7]. These conditions are

$$\begin{aligned} \boldsymbol{m} \times \boldsymbol{h}_{eff} &= \boldsymbol{0} \qquad \text{in } \Omega \\ \boldsymbol{m} \times \partial_{\boldsymbol{n}} \boldsymbol{m} &= \boldsymbol{0} \qquad \text{on } \partial\Omega, \end{aligned} \tag{2.24}$$

where

$$\boldsymbol{h}_{eff} := -\frac{\delta e_{tot}}{\delta \boldsymbol{m}} = 2\widetilde{A}\,\Delta \boldsymbol{m} + \boldsymbol{h}_d + \boldsymbol{h}_{ext} - \frac{\delta e_{ani}}{\delta \boldsymbol{m}}, \tag{2.25}$$

is the (reduced) *effective field*. The second equation in (2.24) implies Neumann conditions $\partial_{\boldsymbol{n}} \boldsymbol{m} = \boldsymbol{0}$ due to the unit norm constraint.

The next chapter is dedicated to the numerical aspects of finding equilibrium magnetization configurations.

Chapter 3

Methods for Energy Minimization

Portions of this chapter were previously submitted for publication as [33] and have been reproduced here with permission of the co-authors. Content which was not generated by the author of this thesis is explicitly denoted.

Probably one of the first numerical methods for the purpose of minimizing the total Gibbs free energy was introduced by LaBonte [34]. This method was supposed to solve Brown's equilibrium equation, i.e. $\boldsymbol{h}_{eff} \times \boldsymbol{m} = \mathbf{0}$, by a finite difference scheme for the Gibbs free energy of a Bloch-wall in a ferromagnetic film. For that reason, in each step of an iterative procedure the normalized effective field defines the new magnetization. This corresponds to a steepest descent step with certain step length, cf. section 3.3. Kosavisutte and Hayashi [35] showed that, in analogy with the SOR (Successive Over-Relaxation) method, LaBonte's method can be accelerated.

Cohen and co-workers [36] introduced a projected non-linear conjugate gradient method for the computation of molecular orientation of liquid crystals. The orientation vectors in liquid crystals have a fixed lengths similar to the magnetization in ferromagnets. In micromagnetics a similarly conjugate gradient method has been applied, [37, 38]. Further, Alouges and co-workers [39] computed equilibrium configurations and switching fields of small ferromagnetic particles.

In the following the micromagnetic energy minimization problem is defined and the Karush-Kuhn-Tucker (KKT) conditions are derived. In the remaining chapters several numerical methods for calculating local solutions are introduced.

3.1 Problem Formulation in the Continuous Setting

Brown's equilibrium equation is equivalent to the problem of finding a magnetization configuration $\boldsymbol{m}$ in such a way that the variational derivative of $h := \psi_t \circ g$ with $g : \boldsymbol{m} \mapsto \boldsymbol{m}/\left\|\boldsymbol{m}\right\|$ (the map onto the unit sphere) is zero:
The variational (or functional) derivative of h is given as

$$\begin{aligned} \frac{\delta}{\delta \boldsymbol{m}} h(\boldsymbol{m}) &= J_g^T(\boldsymbol{m}) \frac{\delta}{\delta \boldsymbol{m}} \psi_t(g(\boldsymbol{m})) \\ &= \frac{\delta}{\delta \boldsymbol{m}} \psi_t(\boldsymbol{m}) - \left(\boldsymbol{m} \cdot \frac{\delta}{\delta \boldsymbol{m}} \psi_t(\boldsymbol{m})\right)\boldsymbol{m}, \end{aligned} \tag{3.1}$$

where $J_g(\boldsymbol{m})$ is the Jacobian of g at $\boldsymbol{m}$.
Remember that the variational (or functional) derivative of a functional h, denoted by $\frac{\delta}{\delta \boldsymbol{m}} h(\boldsymbol{m})$, is defined by

$$\delta h(\boldsymbol{m})(\delta \boldsymbol{m}) = \int \frac{\delta}{\delta \boldsymbol{m}} h(\boldsymbol{m})\, \delta \boldsymbol{m} := \lim_{\epsilon \to 0} \frac{h(\boldsymbol{m} + \epsilon\, \delta \boldsymbol{m}) - h(\boldsymbol{m})}{\epsilon}, \tag{3.2}$$

where the first equality holds in terms of distributions. The differential $\delta h(\boldsymbol{m})(\delta \boldsymbol{m})$ is also denoted as *first variation of h*. If the functional h is defined on a Banach space, then the definition of the variational derivative coincides with that of a *Fréchet derivative*. In most micromagnetic settings the magnetization is either assumed to be smooth up to e.g. second derivatives or element of a Sobolev space. Hence, in those cases one would deal with Fréchet derivatives of functionals defined on Banach or even Hilbert spaces.
For normalized $\boldsymbol{m}$ Eqn. (3.1) corresponds to the orthogonal projection of $\frac{\delta}{\delta \boldsymbol{m}} \psi_t(\boldsymbol{m})$ onto the orthogonal complement of $\boldsymbol{m}$, as can be easily seen by multiplication with $\boldsymbol{m}$. Incidentally, a generalization to non-normalized $\boldsymbol{m}$ would therefore be

$$\frac{\delta}{\delta \boldsymbol{m}} \psi_t(\boldsymbol{m}) - \frac{\boldsymbol{m} \cdot \frac{\delta}{\delta \boldsymbol{m}} \psi_t(\boldsymbol{m})}{\left\|\boldsymbol{m}\right\|^2} \boldsymbol{m}. \tag{3.3}$$

Now, Brown's equilibrium equation $\boldsymbol{h}_{eff} \times \boldsymbol{m} = 0$, cf. Sec. 2.4, means that an equilibrium state $\boldsymbol{m}$ is parallel to the effective field, i.e.

$$-\frac{\delta}{\delta \boldsymbol{m}} \psi_t = \lambda \boldsymbol{m}, \tag{3.4}$$

where multiplying with $\boldsymbol{m}$ and the constraint $\left\|\boldsymbol{m}\right\| = 1$ gives for the multiplier $\lambda = -\delta\psi_t/\delta \boldsymbol{m} \cdot \boldsymbol{m}$ and hence

$$-\frac{\delta}{\delta \boldsymbol{m}} \psi_t - \left(-\frac{\delta}{\delta \boldsymbol{m}} \psi_t \cdot \boldsymbol{m}\right)\boldsymbol{m} = 0. \tag{3.5}$$

Comparing Eqn. (3.5) with (3.1) yields

$$\frac{\delta}{\delta \boldsymbol{m}} h(\boldsymbol{m}) = 0. \tag{3.6}$$

Thus, configurations $\boldsymbol{m}$ which fulfill $\boldsymbol{h}_{eff} \times \boldsymbol{m} = 0$ and $\|\boldsymbol{m}\| = 1$ are critical points of h and vice versa.
Since the micromagnetic energy minimization problem

$$\min \psi_t(\boldsymbol{m}) \quad \text{subject to} \quad \|\boldsymbol{m}\| = 1, \tag{3.7}$$

has the unconstrained form

$$\min h(\boldsymbol{w}) = (\psi_t \circ g)(\boldsymbol{w}), \tag{3.8}$$

the necessary first order optimality condition for normalized magnetization $\boldsymbol{m} := g(\boldsymbol{w})$ is given by Eqn. (3.6), where in the following the quantity

$$\frac{\delta}{\delta \boldsymbol{m}} h(\boldsymbol{m}) = \frac{\delta}{\delta \boldsymbol{m}} \psi_t(\boldsymbol{m}) - \left(\boldsymbol{m} \cdot \frac{\delta}{\delta \boldsymbol{m}} \psi_t(\boldsymbol{m})\right)\boldsymbol{m} = \boldsymbol{m} \times (-\boldsymbol{m} \times \frac{\delta}{\delta \boldsymbol{m}} \psi_t(\boldsymbol{m})), \tag{3.9}$$

is denoted as *projected variational derivative of the energy*, where *Lagrange's formula*

$$\boldsymbol{a} \times (\boldsymbol{b} \times \boldsymbol{c}) = (\boldsymbol{a} \cdot \boldsymbol{c})\boldsymbol{b} - (\boldsymbol{a} \cdot \boldsymbol{b})\boldsymbol{c}, \tag{3.10}$$

was used.

3.2 Problem Formulation in the Discrete Setting

Consider now a discrete setting, i.e. a discretized version of the total magnetic Gibbs free energy, which we denote with capital letter, i.e. Ψ_t shall be a discrete realization of ψ_t cf. (2.17). If not explicitly denoted as tensor, the magnetization is assumed as discrete (mesh) vector $\boldsymbol{m} = (\boldsymbol{m}^{(1)^T}, \boldsymbol{m}^{(2)^T}, \boldsymbol{m}^{(3)^T})^T \in \mathbb{R}^{3N\times 1}$, where $\boldsymbol{m}^{(p)} \in \mathbb{R}^{N\times 1}, p = 1, 2, 3$ are the vectors of x, y and z components with N degrees of freedom. For instance, imagine a Cartesian grid with $N = n_1 n_2 n_3$ nodes and magnetization defined on the centers of a computational cell, compare with section 5.1.4. Alternatively, the reader can also think of a finite element discretization where the degree of freedom N depends on the order of elements and the mesh size. For P1 (first order) N is equal to the number of nodes in the FE mesh.

The problem of discussion is

$$\min_{\boldsymbol{m}\in\mathbb{R}^{3N}} \Psi_t(\boldsymbol{m}) \quad \text{subject to} \quad \left\|\boldsymbol{m}_j\right\|_2^2 := (m_j^{(1)})^2 + (m_j^{(2)})^2 + (m_j^{(3)})^2 = 1,\; j = 1\dots N \tag{3.11}$$

where the (discretized) total energy Ψ_t consists of stray field, exchange, anisotropy and external energy. Note that in general only local solutions to (3.11) can be derived numerically, which is, however, enough for fulfilling Brown's equilibrium condition, cf. Sec. 3.1.
In the following the index in $\|.\|_2$ is omitted. By the definition

$$\boldsymbol{a}_j := (a_j^{(1)}, a_j^{(2)}, a_j^{(3)}), \tag{3.12}$$

for a (mesh) vector $\boldsymbol{a} = (\boldsymbol{a}^{(1)}, \boldsymbol{a}^{(2)}, \boldsymbol{a}^{(3)})^T \in \mathbb{R}^{3N\times 1}$, the quantity $\nabla_\Pi \Psi_t(\boldsymbol{m}) \in \mathbb{R}^{3N\times 1}$ with entries

$$\left(\nabla_\Pi \Psi_t(\boldsymbol{m})\right)_j := \left(\nabla\Psi_t(\boldsymbol{m})\right)_j - \left(\boldsymbol{m}_j \cdot (\nabla\Psi_t(\boldsymbol{m}))_j\right)\boldsymbol{m}_j = \boldsymbol{m}_j \times \left(-\boldsymbol{m}_j \times (\nabla\Psi_t(\boldsymbol{m}))_j\right), \tag{3.13}$$

for $\left\|\boldsymbol{m}_j\right\| = 1$ (feasible), is the discrete analogue of the projected first variation of the energy (3.9), and can therefore be denoted as *projected gradient of the (discretized) energy*. The discrete version of the equilibrium condition (3.6) is therefore

$$\nabla_\Pi \Psi_t(\boldsymbol{m}) = \boldsymbol{0} \text{ for } \boldsymbol{m} \text{ feasible.} \tag{3.14}$$

Also note that Eqn. (3.14) follows from the Lagrange multiplier theorem applied to problem (3.11): By defining the *Lagrangian function*

$$\mathcal{L}(\boldsymbol{m}; \boldsymbol{\lambda}) := \Psi_t(\boldsymbol{m}) - \boldsymbol{\lambda}^T \boldsymbol{c}(\boldsymbol{m}), \tag{3.15}$$

with the *multiplier vector* $\boldsymbol{\lambda} \in \mathbb{R}^{N\times 1}$ and $\boldsymbol{c}(\boldsymbol{m}) \in \mathbb{R}^{N\times 1}$ with $c_j(\boldsymbol{m}) = \frac{1}{2}((m_j^{(1)})^2 + (m_j^{(2)})^2 + (m_j^{(3)})^2 - 1)$, the first order (KKT) conditions

$$\begin{aligned} \nabla_{\boldsymbol{m}} \mathcal{L}(\boldsymbol{m}; \boldsymbol{\lambda}) &= \boldsymbol{0}, \\ \nabla_{\lambda} \mathcal{L}(\boldsymbol{m}; \boldsymbol{\lambda}) &= \boldsymbol{0}, \end{aligned} \tag{3.16}$$

yield $\lambda_j = \boldsymbol{m}_j^T (\nabla\Psi_t(\boldsymbol{m}))_j$, which again substituted into the first equation in (3.16) gives the condition (3.14). The *Karush-Kuhn-Tucker (KKT) conditions* for problem (3.11) are therefore given by Eqn. (3.14), namely *the configuration has to be feasible and the projected gradient at this configuration has to be zero*.

3.2.1 Discretization of the Energy and Gradients

In this section and for the most part in this thesis the domain Ω is cubic and discretized on a Cartesian grid with $N = n^3$ nodes (more general $N = \prod_{p=1}^{3} n_p$) and the magnetization supposed to be defined on the centers of a computational cell, also compare with section 5.1.4.
The exchange energy part is (cf. section 2.2)

$$e_{ex} = -\int_{\Omega} \sum_{p=1}^{3} m^{(p)} \Delta m^{(p)}. \tag{3.17}$$

In this thesis the second derivatives in (3.17) are discretized by symmetric finite differences, i.e.

$$f''(x) = \frac{1}{h^2}(f(x-h) - 2f(x) + f(x+h)) + O(h^2), \tag{3.18}$$

for $f \in C^4[x-h, x+h]$ and Neumann boundary conditions are considered, i.e.

$$f''(x_k) = \frac{1}{h^2}(f(x_k+h) - f(x_k)) + O(h) \tag{3.19}$$

for a node x_k at the boundary. Together with midpoint integration for the integral in (3.17), this yields a second order approximation on regular Cartesian grids to the exchange energy term [40].
Since for the tensor grid methods (except the stray field method introduced in chapter 8) the magnetization and the stray field (cf. chapter 5) are assumed to be constant in each cell (or are defined in the centers of computational cells, respectively), midpoint integration of this terms also yields second order approximations. In this context a term like the stray field energy gets a sum of inner products of mesh vectors or tensors (cf. with the previous section and Eqn. (5.28) in chapter 5)

$$e_{\mathrm{d}} \approx -\frac{1}{2n_1n_2n_3} \sum_{p=1}^{3} (\boldsymbol{m}^{(p)})^T \boldsymbol{h}_d^{(p)}, \tag{3.20}$$

and similar for external and uniaxial anisotropy energies.
The gradients of the discretized energies can be derived by standard calculus. Depending on the underlying discretization scheme (rectangular grid with midpoint integration with/without tensor quantities used; finite elements cf. section 8.8, etc.) also compact (tensor) matrix expressions can be derived. Alternatively, the continuous (integral) form of an energy can be investigated in terms of variational derivatives. For example, the *reciprocity theorem* gives for

the first variation of the demagnetizing energy [7]

$$\delta e_{\mathrm{d}}(\boldsymbol{m})(\delta \boldsymbol{m}) = -\frac{1}{2} \int \boldsymbol{h}_d(\boldsymbol{m}) \cdot \delta \boldsymbol{m} + \boldsymbol{h}_d(\delta \boldsymbol{m}) \cdot \boldsymbol{m} = -\int \boldsymbol{h}_d(\boldsymbol{m}) \cdot \delta \boldsymbol{m}. \tag{3.21}$$

For a certain discretization the first variation becomes an expression of the *total differential* of the discretized energy. E.g. in the case of cell-wise constant magnetization $\boldsymbol{m}_j$ and stray field $(\boldsymbol{h}_d)_j$ one gets

$$\mathrm{d}e_{\mathrm{d}} = \sum_j -|\Omega_j| \, (\boldsymbol{h}_d)_j \cdot d\boldsymbol{m}_j, \tag{3.22}$$

where Ω_j is the j–th discretization cell. Hence, in this case, the gradient of the demagnetizing energy $\nabla e_d(\boldsymbol{m}) \in \mathbb{R}^{3N \times 1}$ is simply given as

$$(\nabla e_d(\boldsymbol{m}))_j := \frac{\partial e_{\mathrm{d}}}{\partial \boldsymbol{m}_j} = -|\Omega_j| \, (\boldsymbol{h}_d)_j. \tag{3.23}$$

The same result can be derived from Eqn. (3.20) if the discrete stray field is assumed to be calculated from a symmetric linear operator $\mathcal{H}$ (e.g. the demagnetizing tensor [13]), i.e. $\boldsymbol{h}_d = \mathcal{H}\boldsymbol{m}$.

3.3 Steepest Descent Method and Variations

Assume the non-linear optimization problem (3.11).
By the definition

$$(H(\boldsymbol{m}))_j := -\boldsymbol{m}_j \times (\nabla \Psi_t(\boldsymbol{m}))_j, \tag{3.24}$$

a steepest descent method would calculate a new iteration $\boldsymbol{m}^{n+1}$ from a given normalized approximation $\boldsymbol{m}^n$ by

$$\boldsymbol{m}_j^{n+1} = \boldsymbol{m}_j^n - \tau_n \, \boldsymbol{m}_j^n \times (H(\boldsymbol{m}^n))_j, \; j = 1 \dots N \tag{3.25}$$

for a certain step size τ_n.
Note, that the steepest descent scheme (3.25) can be seen as the explicit Euler rule for the flow equation

$$\partial_t \boldsymbol{m} = -\boldsymbol{m} \times (\boldsymbol{m} \times \boldsymbol{h}_{eff}(\boldsymbol{m})), \tag{3.26}$$

which is, up to constants, the Landau-Lifshitz equation (2.1) with only damping, since the (discretized) effective field is proportional to $\nabla\Psi_t(\boldsymbol{m})$.

The new approximation $\boldsymbol{m}_j^{n+1}$ in (3.25) is not normalized to 1. Therefore, one can introduce a renormalization step, i.e.

$$\boldsymbol{m}_j^{n+1}(\tau_n) = \frac{1}{\sqrt{1 + \tau_n^2 A_{j,n}^2}} \Big(\boldsymbol{m}_j^n - \tau_n\, \boldsymbol{m}_j^n \times (H(\boldsymbol{m}^n))_j\Big),\ j = 1 \ldots N, \tag{3.27}$$

where $\left\|\boldsymbol{m}_j^{n+1}\right\| = \sqrt{1 + \tau_n^2 A_{j,n}^2},\ A_{j,n} := \left\|(H(\boldsymbol{m}^n))_j\right\|$ was used. This is equivalent to ($\widetilde{\tau}_n := \tau_n / \sqrt{1 + \tau_n^2 A_{j,n}^2}$)

$$\boldsymbol{m}_j^{n+1}(\widetilde{\tau}) = \sqrt{1 - \widetilde{\tau}_n^2 A_{j,n}^2}\, \boldsymbol{m}_j^n - \widetilde{\tau}_n\, \boldsymbol{m}_j^n \times (H(\boldsymbol{m}^n))_j,\ j = 1 \ldots N. \tag{3.28}$$

For determining the step τ_n or $\widetilde{\tau}_n$ one could approximately solve either the one-dimensional problem

$$\min_{\tau_n > 0} \Psi_t\Big(\boldsymbol{m}_j^{n+1}(\tau_n)\Big), \tag{3.29}$$

or

$$\min_{\widetilde{\tau}_n \in [0, 1/A_{j,n}]} \Psi_t\Big(\boldsymbol{m}_j^{n+1}(\widetilde{\tau}_n)\Big), \tag{3.30}$$

by inexact line search, see e.g. [41] or by the so-called *Barzilai-Borwein* (BB) rule [42].

Nevertheless, steepest descent performs very poor in practice. Therefore, two alternatives are introduced in the following, which can be viewed as significant improvements over steepest descent.

LaBonte's original method replaced the magnetization by the normalized gradient, i.e. for the new iterate holds

$$(\nabla\Psi_t(\boldsymbol{m}^n))_j \times \boldsymbol{m}_j^{n+1} = 0,\ \ j = 1 \ldots N. \tag{3.31}$$

Locally (fixed j) this is a steepest descent step with step length $\tau = -\Big((\boldsymbol{m}_j^n)^T (\nabla\Psi_t(\boldsymbol{m}^n))_j\Big)^{-1}$ (if $\boldsymbol{m}_j^n$ and $(\nabla\Psi_t(\boldsymbol{m}^n))_j$ not perpendicular), as can be recalculated for $\boldsymbol{m}_j^{n+1} = \boldsymbol{m}_j^n - \tau\, \boldsymbol{m}_j^n \times (H(\boldsymbol{m}^n))_j$.

3.3.1 A Semi-Implicit Scheme

Replace the steepest descent direction by

$$-\frac{\boldsymbol{m}_j^n + \boldsymbol{m}_j^{n+1}}{2} \times (H(\boldsymbol{m}^n))_j, \tag{3.32}$$

which yields the iteration scheme

$$\boldsymbol{m}_j^{n+1}(\tau) = \boldsymbol{m}_j^n - \tau\frac{\boldsymbol{m}_j^n + \boldsymbol{m}_j^{n+1}}{2} \times (H(\boldsymbol{m}^n))_j\, j = 1\ldots N. \tag{3.33}$$

This update scheme preserves the length of the iterates, i.e. $\left\|\boldsymbol{m}_j^{n+1}\right\| = \left\|\boldsymbol{m}_j^n\right\|$, which can be easily checked by multiplying Eqn. (3.33) by $\boldsymbol{m}_j^n + \boldsymbol{m}_j^{n+1}$. An advantage is the fact that the new state $\boldsymbol{m}_j^{n+1}$ can be computed by explicit formulas [43], so no system of equations has to be solved each step, i.e.

$$\begin{aligned}
d &= 4u + 4\tau bw - 4\tau cv - \tau^2 b^2 u + \tau^2 a^2 u - \tau^2 c^2 u + 2\tau^2 abv + 2\tau^2 acw,\\
e &= 4v + 4\tau cu - 4\tau aw + \tau^2 b^2 v - \tau^2 a^2 v - \tau^2 c^2 v + 2\tau^2 cbw + 2\tau^2 abu,\\
f &= 4w + 4\tau av - 4\tau bu - \tau^2 b^2 w - \tau^2 a^2 w + \tau^2 c^2 w + 2\tau^2 bcv + 2\tau^2 acu,\\
\boldsymbol{m}_j^{n+1}(\tau) &= (d, e, f)^T / N;
\end{aligned} \tag{3.34}$$

where the abbreviations $(H(\boldsymbol{m}^n))_j = (a, b, c)^T$ and $\boldsymbol{m}_j^n = (u, v, w)^T$ are used and $N := 4 + \tau^2(a^2 + b^2 + c^2)$.

The update scheme (3.33) can be seen as an implicit integration rule for the flow equation (3.26). In [33] this method was compared to a state-of-the-art adaptive time integration scheme for (3.26), where for a hard magnetic sample the scheme based on (3.33) outperformed the time integrator.

The first step size τ_0 is calculated by an inexact line search and all subsequent steps τ_n by the so-called *Barzilai-Borwein* (BB) rule [42]:

Define

$$\begin{aligned}
\boldsymbol{g}^n &:= \nabla h(\boldsymbol{m}^n) = \boldsymbol{m}^n \times (-\boldsymbol{m}^n \times \nabla\Psi_t(\boldsymbol{m}^n))\\
\boldsymbol{s}^{n-1} &:= \boldsymbol{m}^n - \boldsymbol{m}^{n-1}\\
\boldsymbol{y}^{n-1} &:= \boldsymbol{g}^n - \boldsymbol{g}^{n-1}.
\end{aligned} \tag{3.35}$$

The step size τ_n is determined such that $D^n := \tau_n^{-1} I$ is an approximation of the Hessian of h at $\boldsymbol{m}^n$, i.e. the *secant equation* $D^n \boldsymbol{s}^{n-1} = \boldsymbol{y}^{n-1}$ holds. By multiplying this equation with $(\boldsymbol{s}^{k-1})^T$ and $(\boldsymbol{y}^{k-1})^T$ one finds the two possible solutions

$$\tau_n^1 = \frac{(\boldsymbol{s}^{n-1})^T \boldsymbol{s}^{n-1}}{(\boldsymbol{s}^{n-1})^T \boldsymbol{y}^{n-1}}, \qquad \tau_n^2 = \frac{(\boldsymbol{s}^{n-1})^T \boldsymbol{y}^{n-1}}{(\boldsymbol{y}^{n-1})^T \boldsymbol{y}^{n-1}}. \tag{3.36}$$

Indeed, in one dimension the condition $D^n \boldsymbol{s}^{n-1} = \boldsymbol{y}^{n-1}$ yields the 'secant approximation' to the second derivative, i.e. $1/\tau_k = (h'_k - h'_{k-1})/(m_k - m_{k-1}) \approx h''_k$.
One possibility is to alternately switch between τ_n^1 and τ_n^2. However, in the algorithm presented in Alg. 2 the more elaborate strategy proposed for step selection in gradient projection methods [44] is used, see Alg. 1. In the numerical tests τ_{max} = 1e+6 and $\tau_{min} \sim$ *stopping tolerance* was used.
Note that the BB rule yields a non-monotonic method, thus, as globalization strategy, an inexact line search (simple backtracking) [41] is used if the new computed energy is still larger than the maximum of the previous (e.g.) 20 energies. Alg. 2 summarizes the method where a $p-$norm is used for the stopping criteria, e.g. $p = \infty$ for maximum norm, or $p = 2$ for l^2-norm (Euclidean norm)

Algorithm 1 Barzilai-Borwein switch; `BBswitch`$(\tau_{min}, \tau_{max}, \tau_n^1, \tau_n^2, \alpha)$

```
Require: $\tau_{min}, \tau_{max} > 0, \tau_n^1, \tau_n^2, \alpha \in [0, 1]$
Ensure: $\tau, \alpha$
if $\tau_n^1 \leq 0$ then                                  ▷ $(\boldsymbol{s}^{n-1})^T \boldsymbol{y}^{n-1} \leq 0$
    $\tau \leftarrow \tau_{max}$
else
    $\tau_1 \leftarrow \max(\tau_{min}, \min(\tau_n^1, \tau_{max}))$
    $\tau_2(n \mod 2) \leftarrow \max(\tau_{min}, \min(\tau_n^2, \tau_{max}))$
    if $\tau_1/\tau_2 \leq \alpha$ then
        $\tau \leftarrow \min \tau_2$
        $\alpha \leftarrow 0.9\alpha$
    else
        $\tau \leftarrow \tau_1$
        $\alpha \leftarrow 1.1\alpha$
    end if
end if
```

3.3.2 Limited Memory Quasi Newton Scheme

Limited memory quasi Newton methods are efficient methods for unconstrained numerical optimization. The Hessian of the objective function is approximated using the vectors (3.35) from a few previous iterations, i.e. the Hessian is approximated by low rank updates (dyadic products), see [41] for a detailed description and analysis of limited memory quasi Newton methods. This yields in the n-th iteration the quasi Newton step

$$\boldsymbol{p}^n = -\boldsymbol{Q}_n^{-1} \boldsymbol{g}^n, \tag{3.37}$$

where $\boldsymbol{Q}_n$ is the approximate Hessian and $\boldsymbol{g}^n$ the gradient of the objective at the current iterate $\boldsymbol{m}^n$. The limited memory variant of the BFGS formulas (lBFGS) allow the direct calculation of

Algorithm 2 Semi-implicit steepest descent scheme; `SemiImplSD(`$\boldsymbol{m}^0$`,tol)`

```
Require: $\boldsymbol{m}^0 \in \mathbb{R}^{3N}$, tol$> 0$
Ensure: $\boldsymbol{m} \in \mathbb{R}^{3N}$
$\boldsymbol{m} \leftarrow$ Calculate steepest descent step from initial state $\boldsymbol{m}^0$ with (inexact) line search ensuring (at least) (3.39)
Set $n \leftarrow 1, \alpha \leftarrow 0.5$
while $\|\nabla_{\Pi}\Psi_t(\boldsymbol{m})\|_p >$ tol do
    if $\Psi_t(\boldsymbol{m}^n) \leq e_{max}$ then        ▷ $e_{max}$ maximum value of energy of last (e.g.) 20 iterations
        $\tau_n^1, \tau_n^2 \leftarrow$ calculate BB steps from (3.36)
        $\tau, \alpha \leftarrow$ BBswitch($\tau_{min}, \tau_{max}, \tau_n^1, \tau_n^2, \alpha$)
        $\boldsymbol{m} \leftarrow \boldsymbol{m}(\tau)$ calculated using formulas (3.34)
    else
        $\tau \leftarrow \arg\min_\tau \Psi_t(\boldsymbol{m}(\tau))$ calculated by (inexact) line search ensuring (at least) (3.39)
        $\boldsymbol{m} \leftarrow \boldsymbol{m}(\tau)$
    end if
    $n \leftarrow n + 1$
end while
```

the quasi-Newton direction $\boldsymbol{p}^n$ by a 2−loop recursion without inversion of $\boldsymbol{Q}_n$ [41]. In their limited memory variants they have about the same computational costs as well as storage demands as the steepest descent method of the previous section. In addition, quasi Newton methods also carry information of the curvature of the problem.

The problem which is suited for lBFGS is the unconstrained version of problem (3.11)

$$\min h(\boldsymbol{m}), \tag{3.38}$$

with the gradient (3.13) if the $\boldsymbol{m}_j$ are normalized. As a step length selection, Alg. 3 uses simple backtracking (like in Alg. 2), which ensures the *sufficient decrease condition*, i.e.

$$h(\boldsymbol{m}^n + \tau\boldsymbol{p}^n) \leq h(\boldsymbol{m}^n) + c_1\tau\nabla(h(\boldsymbol{m}^n))^T\boldsymbol{p}^n, \tag{3.39}$$

with $c_1 = 1\text{e-}4$ and $\tau_0 = 1$ is tried first. A more elaborate line search strategy would also try to ensure a curvature condition, but this is not considered here, see [41] for more information.

3.3.3 Comparison

A good starting point for comparison of the performance of the two methods Alg. 2 and Alg. 3 is the calculation of a flower state in a soft ferromagnetic cube near the single domain limit, i.e. $L \sim 8l_{ex}$. More precise the parameters $\lambda = 8.4515$ and $Q = 0.05$ are chosen. Both methods measure the maximum norm of the projected gradient. Table 3.1 shows the number of iterations and function evaluations (energy and gradient are evaluated simultaneously) for different

Algorithm 3 Limited memory Quasi Newton; `lBFGS(`$\boldsymbol{m}^0$`,tol)`

```
Require: $\boldsymbol{m}^0 \in \mathbb{R}^{3N}$, tol$> 0$
Ensure: $\boldsymbol{m} \in \mathbb{R}^{3N}$
$\boldsymbol{m} \leftarrow$ Calculate steepest descent step from initial state $\boldsymbol{m}^0$ with (inexact) line search ensuring (at least) (3.39)
Set $n \leftarrow 1$
while $\|\nabla_\Pi \Psi_t(\boldsymbol{m})\|_p >$ tol do
    $\boldsymbol{p}^n \leftarrow$ lBFGS quasi Newton direction from 2-loop recursion [41] using limited memory parameter $M$.
    $\tau \leftarrow \arg\min_\tau h(\boldsymbol{m} + \tau \boldsymbol{p}^n)$ calculated by (inexact) line search ensuring (at least) (3.39)
    $\boldsymbol{m} \leftarrow \boldsymbol{m} + \tau \boldsymbol{p}^n$
    $\boldsymbol{m}_j \leftarrow \boldsymbol{m}_j / \left\|\boldsymbol{m}_j\right\|$
    $n \leftarrow n + 1$
end while
```

Table 3.1: The number of iterations and function evaluations (energy and gradient are evaluated simultaneously) for different mesh-sizes ($N = n^3$) and tolerance `tol`= 1e-7 of Alg. 2 and Alg. 3 for cube with $\lambda = 8.4515$ and $Q = 0.05$. Initial state is always uniform magnetization in z-direction. M denotes the limited memory parameter in Alg. 3.

	Alg.	evaluations	iterations
$n = 30$	`SemiImplSD`	87	81
	`lBFGS` ($M = 16$)	42	41
	`lBFGS` ($M = 8$)	42	41
$n = 50$	`SemiImplSD`	182	167
	`lBFGS` ($M = 16$)	59	58
	`lBFGS` ($M = 8$)	68	66

mesh-sizes and tolerance 1e-7. Initial state is always uniform magnetization in z-direction. The limited memory Quasi Newton solver outperforms the (semi-implicit) steepest descent solver in this test, especially for larger grids. Increasing the number of dyadic products in the low-rank approximation of the Hessian (i.e. limited memory parameter) improves the performance in terms of needed iterations and function evaluations. Nevertheless, the storage requirements increase linearly with M. The second test is the calculation of a demagnetizing curve of a 70nm $Nd_2Fe_{14}B$ cube. The material parameters are chosen as following: uniaxial anisotropy axis in z-direction, $J_s = 1.61$T, $K_1 = 4.3\text{e+}6\ J/m^3$ and $A = 7.3\text{e-}12 J/m$. The mesh size is $n = 50$, which yields a grid spacing of about the wall width parameter. The field is applied parallel to the $(1, 0, 1)-$axis and is decreased by steps of $\Delta h = 5\text{e-}3$ from 1 to -2, when the relative l^2-norm ($\|.\|_2 /(3N)$) of the projected gradient is smaller than 1e-11. The previous final (relaxed) magnetization state is taken as the new initial state; the first initial state is parallel to the field axis. Hence, the test involves 601 subsequent minimization problems. Tab. 3.2

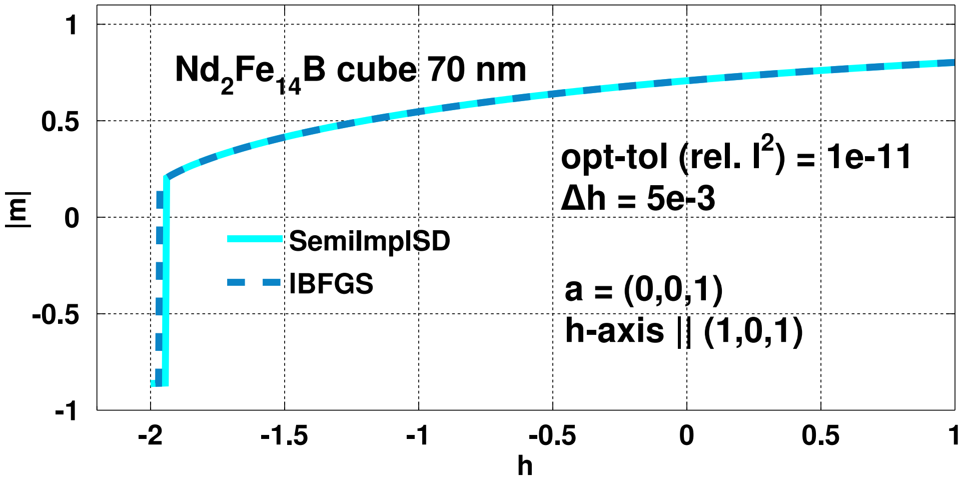

Figure 3.1: Demagnetizing curve of 70nm $Nd_2Fe_{14}B$ cube, i.e. projection of magnetization onto field axis $(1, 0, 1)^T$ ($|m|$) as function of external field h. Uniaxial anisotropy axis is $(0, 0, 1)^T$, $\Delta h = 0.005$, opt-tol (rel. l^2) = 1e-11.

Table 3.2: The number of function evaluations (energy and gradient are evaluated simultaneously) of Alg. 2 and Alg. 3 for calculation of the demagnetizing curve in Fig. 3.1.

Alg.	evaluations
`SemiImplSD`	6162
`lBFGS` ($M = 8$)	3555

shows the number of function evaluations of the two algorithms for the whole simulation.

3.4 Miscellaneous Approaches for Micromagnetic Energy Minimization

The variations of the steepest descent method, introduced in the previous section, represent fast and storage-friendly numerical methods for standard spacial discretizations of the total energy (finite differences, finite elements). However, within the scope of this work, particular tensor structured magnetization configurations will be considered. In this case, the generalization of Alg. 2 and 3 demands for truncation of certain tensor representations whose storage requirements would increase significantly otherwise. However, these generalizations of standard algorithms via so-called approximate tensor arithmetics turns out to yield serious limitations in practice due to possible rapid storage (and computational) demands, cf. Ch. 6 and 7. Alternatively, the micromagnetic energy minimization problem subject to structured tensors can be formulated in a *penalty framework*, which incorporates the constraint into the problem formulation, cf. Sec. 7.4. In the following such *inexact* methods are formulated for micromagnetics. The chapter starts with penalty methods and then describes Newton's method on the KKT conditions. Alouge's method for energy minimization [39] can be considered as a special case of the latter one.

3.4.1 Penalty Methods

The quadratic (l^2-) penalty function for problem (3.11) is

$$Q_\mu(\boldsymbol{m}) := \Psi_t(\boldsymbol{m}) + \frac{\mu}{2}\sum_{j=1}^{N}\big(\underbrace{\tfrac{1}{2}(\|\boldsymbol{m}_j\|^2 - 1)}_{=:c_j(\boldsymbol{m})}\big)^2 = \Psi_t(\boldsymbol{m}) + \frac{\mu}{2}\|\boldsymbol{c}(\boldsymbol{m})\|^2\,, \tag{3.40}$$

where $\mu > 0$ is the *penalty parameter*. In general, a *penalty term* [squared norm term in (3.40)] is a function of the constraints that is greater than zero iff the constraints are *not* fulfilled and therefore 'penalizes' this violation. Penalty methods are in the class of so-called *infeasible*, *inexact* or *exterior methods* because iterates do not fulfill the constraints. Instead, the aim is to approach an optimal solution for the constrained problem from the infeasible region. A general penalty method framework, cf. [41], is summarized in Alg. 4. The penalty parameter has to tend to infinity. A practical strategy is to leave μ unchanged if the previous minimization caused a sufficient decrease of the constraint violation, and increase it otherwise. Hence, in the description of Alg. 4 the scaling factor M_n is assumed to be determined accordingly on run time, e.g. $M_n = 1$ or 5. The final termination condition in Alg. 4 has do be safeguarded by e.g. a maximum allowed iteration number or some condition that checks if the iteration is converging to a stationary point of $\|\boldsymbol{c}(.)\|^2$, see the remark after Th. 1 for a short discussion in the case of the unit-norm constraint. The reason is the following theorem:

Algorithm 4 Quadratic penalty method; `qPM(`$\boldsymbol{m}^0$`,tol,`μ_0`,`$\tau_n \to 0$`)`

```
Require: $\boldsymbol{m}^0 \in \mathbb{R}^{3N}$, tol> 0, $\mu_0 > 0$,$\tau_n \to 0$ (nonnegative)
Ensure: $\boldsymbol{m} \in \mathbb{R}^{3N}$
$\boldsymbol{m} \leftarrow \boldsymbol{m}^0$, $\mu \leftarrow \mu_0$, $n \leftarrow 0$
while $\left\|\nabla_{\boldsymbol{m}} Q_\mu(\boldsymbol{m})\right\|_p >$ tol do
    Find approximate solution $\boldsymbol{x}$ to the sub-problem $\min Q_\mu(.)$ starting at $\boldsymbol{m}$ and ensure $\left\|\nabla_{\boldsymbol{m}} Q_\mu(\boldsymbol{x})\right\| \leq \tau_n$
    $\mu \leftarrow M_n \mu \, (M_n \geq 1)$
    $\boldsymbol{m} \leftarrow \boldsymbol{x}$
    $n \leftarrow n + 1$
end while
```

Theorem 1 ([41] Th.17.2)**.** *Let $\mu \to \infty$ and $\tau_n \to 0$ in Alg. 4. Then for a limit point $\boldsymbol{m}^*$ holds:*

- *If $\boldsymbol{m}^*$ is infeasible (constraint is violated), it is a stationary point of $\|\boldsymbol{c}(.)\|^2$.*

- *If* $\boldsymbol{m}^*$ *is feasible , then* $\boldsymbol{m}^*$ *is a KKT point for problem* (3.11)*, i.e.* (3.16) *holds.*

Remark 1. *In the original theorem from [41] in the feasible case of* $\boldsymbol{m}^*$ *the additional condition that* $\boldsymbol{D}(\boldsymbol{m}^*) := [\nabla \boldsymbol{c}_1(\boldsymbol{m}^*), \ldots, \nabla \boldsymbol{c}_N(\boldsymbol{m}^*)]$ *has full rank has to be satisfied.*
Since for the micromagnetic side constraints holds

$$\boldsymbol{D}(\boldsymbol{m}) = \begin{bmatrix} \boldsymbol{m}_1 & & \\ & \ddots & \\ & & \boldsymbol{m}_N \end{bmatrix} \in \mathbb{R}^{3N \times N}, \tag{3.41}$$

the required condition is fulfilled automatically whenever $\mathbf{m}$ *is non-degenerate (particularly the case if* $\mathbf{m}$ *is feasible).*
On the other hand, if $\boldsymbol{m}^*$ *in Th. 1 is infeasible (*$\mathbf{c}(\boldsymbol{m}_j^*) \neq \mathbf{0}$*) it follows that* $\mathbf{c}(\boldsymbol{m}^*) \in ker(\boldsymbol{D}(\boldsymbol{m}^*))$*, i.e.*

$$\boldsymbol{D}(\boldsymbol{m}^*)\boldsymbol{c}(\boldsymbol{m}^*) = \mathbf{0}. \tag{3.42}$$

This is equivalent to the 'complementarity' condition

$$c_j(\boldsymbol{m}^*)\boldsymbol{m}_j^* = \mathbf{0} \quad \textit{for all } j = 1 \ldots N. \tag{3.43}$$

Hence, the first case in Th. 1 can be identified by checking the condition (3.43)*, e.g. if* $\left| \|\boldsymbol{m}_j\|^2 - 1 \right|$ *is greater some tolerance* `tol`$_2 := 1 - \varepsilon > 0$ *check if* $\|\boldsymbol{m}_j\|^2 < \varepsilon$. □

An inherent problem of the quadratic penalty method is the ill-conditioning in the Hessian $\nabla^2_{\boldsymbol{mm}} Q_\mu$ as μ increases: The Hessian is given as

$$\nabla^2_{\boldsymbol{mm}} Q_\mu(\boldsymbol{m}) = \nabla^2 \Psi_t + \sum_{j=1}^{N} \mu c_j(\boldsymbol{m}) \nabla^2 c_j(\boldsymbol{m}) + \mu \boldsymbol{D}(\boldsymbol{m}) \boldsymbol{D}(\boldsymbol{m})^T. \tag{3.44}$$

From Th.17.2 in [41] (feasible case) one gets near the minimizer

$$\nabla^2_{\boldsymbol{mm}} Q_\mu(\boldsymbol{m}) \approx \nabla^2_{\boldsymbol{mm}} \mathcal{L}(\lambda^*) + \mu \boldsymbol{D}(\boldsymbol{m}) \boldsymbol{D}(\boldsymbol{m})^T, \tag{3.45}$$

which is a sum of a part that is independent of μ (the first one of (3.45) which contains the Hessian of the Lagrangian function $\mathcal{L}$) and a rank N matrix whose nonzero eigenvalues are of order μ. Hence, the Hessian of the penalty function has some eigenvalues approaching a constant, while others are of order μ. Since $\mu \to \infty$, the condition number $\kappa(\nabla^2_{\boldsymbol{mm}} Q_\mu(\boldsymbol{m})) = |\lambda_{max}/\lambda_{min}| \to \infty$. Nevertheless, there exists a well-conditioned reformulation of the linear system which determines a Newton step in the unconstrained subproblems of Alg. 4, [41].

Moreover, so-called *exact penalty methods* do not require the penalty parameter to approach infinity due to the fact that there exists a finite value μ^* for which the local minima of the penalty functions with $\mu > \mu^*$ coincide with the solution (KKT point) of the constrained problem [41]. Such penalty functions use for instance the l^1-norm, l^2-norm (not squared) or l^∞-norm, which are non-smooth functions.

Another penalty-like method is the *augmented Lagrangian method*, which consists of unconstrained subproblems for the *augmented Lagrangian function*

$$\mathcal{L}_A(\boldsymbol{m}, \boldsymbol{\lambda}, \mu) := Q_\mu(\boldsymbol{m}) - \boldsymbol{\lambda}^T \boldsymbol{c}(\boldsymbol{m}), \tag{3.46}$$

where the penalty parameter μ and the vector $\boldsymbol{\lambda}$ are fixed. Near the solution one gets $\boldsymbol{\lambda}^* \approx \boldsymbol{\lambda} - \mu\boldsymbol{c}(\boldsymbol{m})$, which gives rise to the updating scheme in the n-th iteration: $\boldsymbol{\lambda}^{n+1} = \boldsymbol{\lambda}^n - \mu^n \boldsymbol{c}(\boldsymbol{m}^n)$. Conditions for convergence of the augmented Lagrangian method [41] do not require μ to approach infinity, which makes ill-conditioning of this method less a problem than for the quadratic penalty method.

The l^2-penalty method (as well as the augmented Lagrangian method) needs several hundred function evaluations for the flower state example of Sec. 3.3.3 and can be considered to be inferior to the previous introduced methods. Nevertheless, the l^2-penalty method represents a suitable way to treat the point-wise unit norm constraints when dealing with structured tensors as magnetization, Ch. 7.

3.4.2 Newton's Method on the KKT Conditions

The KKT conditions (3.16) are a set of $4N$ non-linear equations

$$\boldsymbol{F}(\boldsymbol{m}, \boldsymbol{\lambda}) = \begin{bmatrix} \nabla\Psi_t(\boldsymbol{m}) - \boldsymbol{D}(\boldsymbol{m})\boldsymbol{\lambda} \\ \boldsymbol{c}(\boldsymbol{m}) \end{bmatrix} = \boldsymbol{0}, \tag{3.47}$$

where $\boldsymbol{D}$ is defined in (3.41). Solving this set of equations by Newton's method requires the Jacobian matrix of $\boldsymbol{F}$

$$\boldsymbol{F}'(\boldsymbol{m}, \boldsymbol{\lambda}) = \begin{bmatrix} \nabla^2\Psi_t - \boldsymbol{\Lambda} & -\boldsymbol{D}(\boldsymbol{m}) \\ \boldsymbol{D}(\boldsymbol{m})^T & \boldsymbol{0} \end{bmatrix}, \tag{3.48}$$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{3N\times 3N}$ is a diagonal matrix consisting of the diagonal blocks $\boldsymbol{\Lambda}_j = \lambda_j I_3 \in \mathbb{R}^{3\times 3}$. Alg. 5 summarizes the Newton procedure for (3.47). The linear system (3.49) has a unique solution if the Hessian matrix $\nabla^2_{\boldsymbol{mm}}\mathcal{L}(\boldsymbol{\lambda}) = \nabla^2\Psi_t - \boldsymbol{\Lambda}$ is positive definite on, at least, the tangent space of the unit norm constraints, that is $\boldsymbol{d}^T\nabla^2_{\boldsymbol{mm}}\mathcal{L}(\boldsymbol{\lambda})\boldsymbol{d} > 0$ for all $\boldsymbol{d} \neq \boldsymbol{0}$ with $\boldsymbol{D}(\boldsymbol{m})^T\boldsymbol{d} = \boldsymbol{0}$ (means $\boldsymbol{d}_j \perp \boldsymbol{m}_j$, $j = 1\ldots N$), cf. Lemma 16.1 [41]. Under this condition Alg. 5 converges

Algorithm 5 Newton's method for KKT system; `NKKT(`$\boldsymbol{m}^0$`,`$\boldsymbol{\lambda}^0$`,tol)`

```
Require: $\boldsymbol{m}^0 \in \mathbb{R}^{3N}, \boldsymbol{\lambda}^0 \in \mathbb{R}^N$, tol $> 0$
Ensure: $\boldsymbol{m} \in \mathbb{R}^{3N}$
$\boldsymbol{m} \leftarrow \boldsymbol{m}^0, \boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda}^0$
while $\|\boldsymbol{F}(\boldsymbol{m}, \boldsymbol{\lambda})\|_p >$ tol do
    Solve the linear system
        $\boldsymbol{F}'(\boldsymbol{m}, \boldsymbol{\lambda})[\delta\boldsymbol{m}^T, \delta\boldsymbol{\lambda}^T]^T = -\boldsymbol{F}(\boldsymbol{m}, \boldsymbol{\lambda}).$   (3.49)
    $[\boldsymbol{m}^T, \boldsymbol{\lambda}^T]^T \leftarrow [\boldsymbol{m}^T, \boldsymbol{\lambda}^T]^T + [\delta\boldsymbol{m}^T, \delta\boldsymbol{\lambda}^T]^T$
end while
```

quadratically provided that the initial guess is close enough to the solution.

The linear system (3.49) can also be seen as the KKT conditions of a quadratic program, i.e. quadratic expansion of the Lagrangian function at the current iterate $\boldsymbol{m}$ subject to linearized constraints, [41]. This allows one to apply ideas from quadratic programming on the subproblems (3.49) (so-called *sequential quadratic programming* (SQP)), e.g. Null-space methods, projected CG, (reduced Hessian) quasi-Newton, etc. [41].

As a numerical test the flower state example of Sec. 3.3.3 is chosen. The subproblems in Alg. 5 are solved with a GMRES algorithm where the action of the Hessian $\nabla^2\Psi_t$ is computed with the help of the gradient, i.e. $(\nabla^2\Psi_t)\boldsymbol{x} = \nabla\Psi_t(\boldsymbol{x}) + 1/n^3\boldsymbol{h}_{ext}$. As the stopping criterion the projected gradient is measured in the maximum norm like in Sec. 3.3.3. For the case $n = 30$ the number of function evaluations were 67 (those for the Hessian evaluation included) and 6 iterations in the outer loop, which is comparable with the corresponding amount of evaluations that were needed for the gradient based methods in Sec. 3.3.3. However, further numerical tests show that the Newton-KKT (NKKT) algorithm Alg. 5 strongly depends on the initial state. Further, iterative solution of the systems (3.49) would need preconditioning for large system sizes. In reference [45] a block preconditioner is suggested for the reduced Newton system arising from liquid crystal modeling which is a similar problem like the micromagnetic energy minimization problem.

Relation to Alouge's method:

Alouge and co-workers proposed an update scheme [39] (for a finite element discretization of the energy) where in each step a quadratic program has to be solved, i.e.

$$\min_{\delta\boldsymbol{m}} \Psi_t(\boldsymbol{m} + \delta\boldsymbol{m}) \quad \text{s.t.} \quad \boldsymbol{D}(\boldsymbol{m})^T\delta\boldsymbol{m} = \boldsymbol{0}. \tag{3.50}$$

Since Ψ_t is quadratic the objective in (3.50) is

$$\Psi_t(\boldsymbol{m} + \delta\boldsymbol{m}) = \Psi_t(\boldsymbol{m}) + (\nabla\Psi_t(\boldsymbol{m}))^T\delta\boldsymbol{m} + \tfrac{1}{2}\delta\boldsymbol{m}^T\nabla^2\Psi_t\,\delta\boldsymbol{m}. \tag{3.51}$$

The first order optimality conditions for (3.50) lead to a linear system like (3.49) but with $\Lambda = \mathbf{0}$ and $\boldsymbol{c}(\boldsymbol{m}) = \mathbf{0}$. The updated solution in the algorithm proposed in [39], i.e. $\boldsymbol{m}+\tau_n\delta\boldsymbol{m}$ for suitable τ_n, is normalized afterwards, hence, $\boldsymbol{m}$ is always normalized. This justifies $\boldsymbol{c}(\boldsymbol{m}) = \mathbf{0}$ in the linear system. However, the choice $\Lambda = \mathbf{0}$ is somehow questionable. Numerical tests also show slow convergence, respectively, serious problems near a solution. Moreover, the linear system is solvable if $\nabla^2\Psi_t$ is positive definite which does not have to be the case. Nevertheless, since $\boldsymbol{m}$ is normalized any matrix of the type of Λ can be added to the Hessian of Ψ_t, e.g. constructed from an estimate of the Lagrange multiplier vector (to the original unit norm constraint) $\boldsymbol{\lambda} = \boldsymbol{D}(\boldsymbol{m})^T\nabla\Psi_t(\boldsymbol{m})$ cf. (3.16).

Chapter 4

Tensor Formats

Portions of this chapter were previously published as [25] or submitted for publication as [26] and have been reproduced here with permission of the co-authors. Content which was not generated by the author of this thesis is explicitly denoted.

Most of the computational schemes developed in this thesis rely on, or at least give the possibility to make use of, structured tensor representation, so called *tensor formats*. The following chapter is dedicated to data sparse formats for multidimensional data.

4.1 Motivation

In the following the term *tensor* is understood as *multidimensional array* ('high dimensional matrix'). Most of the time three dimensions are enough for the purpose of the following discussion.

Let the set of (order-3) tensors [1] with mode sizes $\boldsymbol{n} = (n_1, n_2, n_3)$ over the field $K = \mathbb{R}$ or $\mathbb{C}$ be denoted with $\bigotimes_{p=1}^{3} \mathbb{K}^{n_p}$ and the set of matrices of size $n_1 \times n_2$ as usual with $\mathbb{K}^{n_1 \times n_2}$. The tensor space $\bigotimes_{p=1}^{3} \mathbb{K}^{n_p}$ is the vector space generated by linear combinations of so-called rank-1 tensors (*elementary tensors*), e.g. §3.2.6.1 in reference [47] and Sec. 4.2.

In the last few years much research was done on problems and applications where tensors appear as data or solutions. There are two basic distinctions in principle:

First, there are many problems where tensors arise as multidimensional (observed) data sets, e.g. *psychometrics*, *data mining*, *neuroscience*, *image compression and classification*, see [46] references therein.

Secondly, those problems where tensors come from an underlying multivariate function, like for instance if one discretizes a Poisson equation $-\Delta u = f$ on $\Omega = [0, 1]^3$ on a regular Cartesian grid (tensor grid) in, e.g., three spacial dimensions. A tensor grid method would aim to approx-

---

[1] Another common notation is $\mathbb{K}^{\boldsymbol{I}}$ where $\boldsymbol{I} = I_1 \times I_2 \times I_3$ and $I_p = \{1 \ldots n_p\}$, see [46]

imate the solution $u^* = u^*(x_1, x_2, x_3)$ on a tensor grid of size $\boldsymbol{n} = (n_1, n_2, n_3)$ (mode-sizes) by a tensor $\mathcal{U} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$ with entries $u_{i_1 i_2 i_3} \approx u^*(X(i_1), Y(i_2), Z(i_3))$, where X, Y, Z is meant to be the lists of x, y and z coordinates of grid points, respectively. Thus, the tensor $\mathcal{U}$ contains the (approximate) function values of the solution sampled on the grid. In this context, $\mathcal{U}$ is called a *function-related tensor*.

In general, the explicit storage of the $\prod_{q=1}^{d} n_q$ (complex) numbers in the case of $d \gg 2$ dimensions or large mode-sizes is not possible anymore, similar difficulties arise in the context of computational cost; from Beylkin-Mohlenkamp [48]:

'When an algorithm in dimension one is extended to dimension d, in nearly every case its computational cost is taken to the power d. This fundamental difficulty is the single greatest impediment to solving many important problems and has been dubbed the curse of dimensionality.'

To break this curse, an approach based on separated representation of functions and operators in higher dimensions was discussed [48], i.e.

$$f(x_1, \ldots, x_d) = \sum_{l=1}^{r} \alpha_l \, f_l^{(1)}(x_1) \, f_l^{(2)}(x_2) \ldots f_l^{(d)}(x_d). \tag{4.1}$$

The discrete representation of the function in (4.1) on a d-dimensional rectangular domain (tensor product grid) is called a *canonical tensor*, see section 4.2. Also approximation schemes for compressed format of dense discrete (tensor) representation of multidimensional data or solutions/operators of high dimensional problems, e.g. by low-rank decomposition, were developed, see for example [46, 47] or [6] for a literature survey on low-rank tensor approximation techniques.

The following sections are brief introductions into the widely used tensor formats, i.e. *canonical tensors*, *Tucker tensors* and *Tensor Trains*, but also aspects like (best) approximation from a theoretical and practical point of view are discussed. For an extensive review on structured tensors and some algorithms to compute structured tensor approximations the reader is referred to the work [46] and references therein. The most common arithmetic operations on Tucker and canonical tensors are presented in [49]. Those for tensor trains can be found in [50].

The following definitions are fundamental for the further discussion.

Let $\mathcal{X}, \mathcal{Y} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$. The *Frobenius norm* is defined as

$$\|\mathcal{X}\|_F := \sqrt{\sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \sum_{i_3=1}^{n_3} x_{i_1 i_2 i_3}^2}, \tag{4.2}$$

which is associated with a *scalar product*

$$\langle \mathcal{X}, \mathcal{Y} \rangle := \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \sum_{i_3=1}^{n_3} x_{i_1 i_2 i_3}\, y_{i_1 i_2 i_3}, \tag{4.3}$$

with $\|\mathcal{X}\|_F^2 = \langle \mathcal{X}, \mathcal{X} \rangle$.

For a matrix $\boldsymbol{U} \in \mathbb{R}^{m \times n_j}$ the *j-mode matrix product* $\mathcal{X} \times_j \boldsymbol{U}$ of the tensor $\mathcal{X}$ with $\boldsymbol{U}$ is defined element-wise in the following way. E.g. for $j = 1$,

$$(\mathcal{X} \times_1 \boldsymbol{U})_{i_1\, i_2\, i_3} := \sum_{i'=1}^{n_1} x_{i'\, i_2\, i_3}\, u_{i_1\, i'}, \tag{4.4}$$

i.e., the resulting tensor $\mathcal{X} \times_1 \boldsymbol{U} \in \mathbb{R}^{m \times n_2 \times n_3}$ is obtained by right-multiplication of the mode-1 fibers (columns) of $\mathcal{X}$ by $\boldsymbol{U}$. Analogously for $j = 2, 3$; the cost for the computation of $\mathcal{X} \times_j \boldsymbol{U}$ is $O(m \prod_{q=1}^{3} n_q)$ operations in general. In the forthcoming text the following definitions are useful.

Definition 1 (Tensor outer product)**.** *Let* $\boldsymbol{a}^{(p)} \in \mathbb{R}^{n_p}, p = 1 \ldots d$. *Then the outer product of the vectors* $\boldsymbol{a}^{(p)}$ *is a tensor* $\mathcal{X} \in \bigotimes_{p=1}^{d} \mathbb{R}^{n_p}$ *given via*

$$\begin{aligned} \mathcal{X} &= \boldsymbol{a}^{(1)} \circ \boldsymbol{a}^{(2)} \circ \ldots \circ \boldsymbol{a}^{(d)}, \\ x_{i_1 i_2 \ldots i_d} &= a_{i_1}^{(1)} a_{i_2}^{(2)} \ldots a_{i_d}^{(d)}. \end{aligned} \tag{4.5}$$

Definition 2 (Kronecker product)**.** *For two matrices* $\boldsymbol{A} \in \mathbb{R}^{I \times J}$ *and* $\boldsymbol{B} \in \mathbb{R}^{K \times L}$ *the Kronecker product is defined by*

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & a_{12}\boldsymbol{B} & \ldots & a_{1J}\boldsymbol{B} \\ a_{21}\boldsymbol{B} & a_{22}\boldsymbol{B} & \ldots & a_{2J}\boldsymbol{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\boldsymbol{B} & a_{I2}\boldsymbol{B} & \ldots & a_{IJ}\boldsymbol{B} \end{bmatrix} \in \mathbb{R}^{IK \times JL}. \tag{4.6}$$

Definition 3 (Khatri-Rao product)**.** *Given two matrices* $\boldsymbol{A} \in \mathbb{R}^{I \times K}$ *and* $\boldsymbol{B} \in \mathbb{R}^{J \times K}$*, their Khatri-Rao product is defined by*

$$\boldsymbol{A} \odot \boldsymbol{B} = [\boldsymbol{a}_1 \otimes \boldsymbol{b}_1 \quad \boldsymbol{a}_2 \otimes \boldsymbol{b}_2 \quad \ldots \quad \boldsymbol{a}_K \otimes \boldsymbol{b}_K] \in \mathbb{R}^{IJ \times K}. \tag{4.7}$$

4.2 Canonical Tensors

Definition 4. *A tensor $\mathcal{X} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$ is said to be in canonical format [CANDECOMP/PARAFAC (CP) decomposition] with (outer product) rank R, if*

$$\mathcal{X} = \sum_{r=1}^{R} \lambda_r \, \boldsymbol{u}_r^{(1)} \circ \boldsymbol{u}_r^{(2)} \circ \boldsymbol{u}_r^{(3)}, \tag{4.8}$$

with $\lambda_r \in \mathbb{R}$, and (unit) vectors $\boldsymbol{u}_r^{(j)} \in \mathbb{R}^{n_j}$.

Abbreviating notation as in [46], a tensor in CP format is written as

$$\mathcal{X} \equiv [\![\boldsymbol{\lambda}; \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!], \tag{4.9}$$

with weight vector $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_R] \in \mathbb{R}^R$ and (factor) matrices $\boldsymbol{U}^{(j)} = [\, \boldsymbol{u}_1^{(j)} \,|\, \ldots \,|\, \boldsymbol{u}_R^{(j)} \,] \in \mathbb{R}^{n_j \times R}$. The storage requirement for the canonical tensor format amounts to $R \sum_{j=1}^{3} n_j$.
In the following $\mathcal{C}_{\mathbf{n},r}$ denotes the set of canonical tensors with mode sizes $\mathbf{n} = (n_1, n_2, n_3)$ and rank r, and simple $\mathcal{C}_{n,r}$, when the mode sizes are equal.
The *inner product* for two canonical tensors $\mathcal{A} \in \mathcal{C}_{\mathbf{n},r_1}$ and $\mathcal{B} \in \mathcal{C}_{\mathbf{n},r_2}$, as well as other operations, can be performed with reduced complexity [for the inner product operation it amounts to $\mathcal{O}(r_1 r_2 \sum_q n_q)$], see e.g. [49].

In general, element-wise operations (element-wise multiplication, evaluation of a function on the entries, etc.) for structured mathematical objects (matrices, tensors) are difficult to perform within the format in terms of preserving the structure and not destroy it. The element-wise product of CP tensors is a good example for this issue; it can be performed within the format but increases the rank:
For two canonical tensors $\mathcal{X} = [\![\, \boldsymbol{\lambda}; \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)} \,]\!] \in \mathcal{C}_{\boldsymbol{n},r}$ and $\mathcal{Y} = [\![\, \boldsymbol{\mu}; \boldsymbol{V}^{(1)}, \boldsymbol{V}^{(2)}, \boldsymbol{V}^{(3)} \,]\!] \in \mathcal{C}_{\boldsymbol{n},r'}$ of equal mode sizes the *Hadamard product* (element-wise product) is

$$(\mathcal{X} \bullet \mathcal{Y})_{IJK} = \sum_i \sum_j \lambda_i \mu_j \, (u_{Ii}^{(1)} v_{Ij}^{(1)}) \, (u_{Ji}^{(2)} v_{Jj}^{(2)}) \, (u_{Ki}^{(3)} v_{Kj}^{(3)}) \equiv [\![\, \boldsymbol{\nu}; \boldsymbol{W}^{(1)}, \boldsymbol{W}^{(2)}, \boldsymbol{W}^{(3)} \,]\!] \in \mathcal{C}_{\boldsymbol{n},rr'}, \tag{4.10}$$

where $\boldsymbol{\nu} = [\boldsymbol{\lambda}, \boldsymbol{\mu}]$ and $\boldsymbol{W}^{(p)} = \left[\boldsymbol{u}_1^{(p)} \bullet \boldsymbol{v}_1^{(p)} \,|\, \ldots \,|\, \boldsymbol{u}_1^{(p)} \bullet \boldsymbol{v}_{r'}^{(p)} \,|\, \ldots \,|\, \boldsymbol{u}_r^{(p)} \bullet \boldsymbol{v}_{r'}^{(p)} \right] \in \mathbb{R}^{n_p \times rr'}$. The cost for forming (4.10) is of order $\mathcal{O}(rr' \sum_{p=1}^{3} n_p)$. The new CP tensor has rank $r\,r'$. Direct recompression can be considered e.g. by optimization [51].

The *tensor (outer-) product rank*, i.e. the minimal number of rank-1 terms in a representation like (4.8) for a tensor $\mathcal{X}$, is an analogue of the matrix rank. However there are major differences between those two [46]. The product rank of a tensor might be different over $\mathbb{R}$ and $\mathbb{C}$; in principle there is no easy algorithm to determine the tensor rank since this is an NP-complete problem [52]. In fact, there are several specific examples of tensors where only bounds exist for their ranks. Moreover the tensor rank is not *upper semi-continuous*, e.g. there exist sequences of tensors of rank $\leq r$ converging to a tensor of rank greater than r [53]. There is no *Eckart-Young Theorem* available, i.e. a CP decomposition can not be computed via the *singular value decomposition* (SVD), indeed, it is possible that the best rank-r approximation of a tensor of order greater than two may not even exist for the case $r > 1$, [53]. Hence, the set $C_{\boldsymbol{n},r} \subset \bigotimes_{p=1}^{d} \mathbb{R}^{n_p}$ is *not* closed for $r > 1$ and $d > 2$. In fact, the best approximation problem $\min_{\mathcal{X}\in C_{\boldsymbol{n},r}} \|\mathcal{X} - \mathcal{Y}\|$ is unsolvable if and only if infimum sequences are unstable, i.e. their rank-1 terms get unbounded, see §9.4 and §9.5.3 in reference [47]. Hence, closed subsets of canonical tensors are those with terms bounded by a constant $c > 0$, i.e.

$$C^{c}_{\boldsymbol{n},r} := \{\mathcal{X} = \sum_{j=1}^{r} \boldsymbol{u}_r^{(1)} \circ \boldsymbol{u}_r^{(2)} \circ \boldsymbol{u}_r^{(3)} \in C_{\boldsymbol{n},r} : \sum_{j=1}^{r} \left\|\boldsymbol{u}_r^{(1)} \circ \boldsymbol{u}_r^{(2)} \circ \boldsymbol{u}_r^{(3)}\right\|^2 = \sum_{j=1}^{r} \prod_{q=1}^{3} \left\|\boldsymbol{u}_r^{(q)}\right\|^2 \leq c\}. \tag{4.11}$$

Nevertheless, a broad community uses canonical decomposition, e.g. *psychometrics*, *data mining*, *neuroscience*, *image compression and classification*, see [46] references therein.
Algorithms for computing canonical decompositions are mostly based on optimization, e.g. *alternating least squares* [46], *gradient based* or *nonlinear least squares methods* [51] or *Gauss-Newton* [54].
Also *black-box approximation* using fibre-crosses was considered [55, 56]. This method is also based on an optimization approach but the cost function only requires the evaluation of the original tensor on small sets of indices.
Approximation of *operators* like the *multi-particle Schrödinger operator* [48] or Newton potential [57] can also be achieved by using the canonical tensor format in order to overcome the *curse of dimensionality*. For matrices $\boldsymbol{A} \in \mathbb{R}^{\left(\prod_{i=1}^{3} n_i\right)\times\left(\prod_{i=1}^{3} n_i\right)}$, typically arising from discretized operators, the canonical format is usually given in *Kronecker product form* [58]

$$\boldsymbol{A} = \sum_{j=1}^{r} \alpha_j \boldsymbol{U}_j^{(1)} \otimes \boldsymbol{U}_j^{(2)} \otimes \boldsymbol{U}_j^{(3)}, \tag{4.12}$$

with matrices $\boldsymbol{U}_j^{(q)} \in \mathbb{R}^{n_q \times n_q}$, scalars α_j and $\otimes$ equals *Kronecker product*. Due to the relation $\mathrm{vec}(\mathrm{vec}(\boldsymbol{U}^{(3)}) \circ \mathrm{vec}(\boldsymbol{U}^{(2)}) \circ \mathrm{vec}(\boldsymbol{U}^{(1)})) = \mathrm{vec}(\boldsymbol{U}^{(1)}) \otimes \mathrm{vec}(\boldsymbol{U}^{(2)}) \otimes \mathrm{vec}(\boldsymbol{U}^{(3)})$, the form (4.12) can be identified with (4.8), where the *vectorization* vec(.) is understood as in [46].
Storage and tensor operations for the canonical format scale linearly in the dimension d, rather than exponentially as for dense tensors. However, the above mentioned drawbacks (instability, lack of robust algorithms) have led to the development of other (stable) formats that scale linearly or polynomially in the dimension, such as *$\mathcal{H}$-Tucker* [59] which relies on hierarchical tree structure, or the *Tensor Train format* [50], which is briefly discussed in Sec. 4.4.

4.3 Tucker Tensors and Quasi-Best Approximation

Definition 5. *A Tensor $\mathcal{X} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$ is said to be represented in Tucker format if*

$$\mathcal{X} = \mathcal{C} \times_1 \boldsymbol{U}^{(1)} \times_2 \boldsymbol{U}^{(2)} \times_3 \boldsymbol{U}^{(3)} \equiv [\![\mathcal{C};\, \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!], \tag{4.13}$$

with the so-called core tensor $\mathcal{C} \in \bigotimes_{p=1}^{3} \mathbb{R}^{r_p}$ and (factor) matrices $\boldsymbol{U}^{(j)} \in \mathbb{R}^{n_j \times r_j}$.

The storage requirement for (4.13) is $\prod_{j=1}^{3} r_j + \sum_{j=1}^{3} n_j r_j$, which is smaller than $\prod_{j=1}^{3} n_j$ if $r_j \ll n_j$.

The j-*rank* of a tensor $\mathcal{X}$ is the rank of the unfolding (j-mode matricization) $\mathcal{X}_{(j)}$, [46]. By setting $r_j = \mathrm{rank}(\mathcal{X}_{(j)})$, the tensor $\mathcal{X}$ is usually referred to as *rank-*(r_1, r_2, r_3) *tensor*. Of course $r_j \leq n_j$ holds.

In the following the set of Tucker tensors with mode sizes $\boldsymbol{n} = (n_1, n_2, n_3)$ and (j-) ranks $\boldsymbol{r} = (r_1, r_2, r_3)$ is denoted with $\mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$ and with $\mathcal{T}_{\boldsymbol{n},r}$ if $r \equiv r_j$. In fact, $\mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$ contains all tensors with mode-size $\boldsymbol{n}$ and j-ranks smaller or equal r_j, [47]. There also holds $\mathcal{C}_{\boldsymbol{n},r} \subseteq \mathcal{T}_{\boldsymbol{n},r}$ since a canonical tensor can be identified with a Tucker tensor which has a diagonal core tensor. Hence, the 'Tucker-rank' is smaller or equal the 'CP-rank'.

Many algebraic operations for Tucker tensors, like inner product, mode-multiplication, etc., can be performed with reduced complexity, see e.g. [49]. Again, the elemet-wise evaluation of Tucker tensors is non-trivial in general. The element-wise product is described in detail:

For two Tucker tensors $\mathcal{X} = [\![\mathcal{C};\, \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!] \in \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$ and $\mathcal{Y} = [\![\mathcal{D};\, \boldsymbol{V}^{(1)}, \boldsymbol{V}^{(2)}, \boldsymbol{V}^{(3)}]\!] \in \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}'}$ of equal mode sizes the *Hadamard product* (element-wise product) is

$$(\mathcal{X} \bullet \mathcal{Y})_{IJK} = \sum_{i,j,k} \sum_{l,m,n} c_{ijk}\, d_{lmn}\, (u^{(1)}_{Ii} v^{(1)}_{Il})\, (u^{(2)}_{Jj} v^{(2)}_{Jm})\, (u^{(3)}_{Kk} v^{(3)}_{Kn}). \tag{4.14}$$

The Hadamard product (4.14) can be written in compact form with the *Khatri-Rao product*. It is straight forward to show

$$\mathcal{X} \bullet \mathcal{Y} = [\![\mathcal{E};\, (\boldsymbol{V}^{(1)^T} \odot \boldsymbol{U}^{(1)^T})^T, (\boldsymbol{V}^{(2)^T} \odot \boldsymbol{U}^{(2)^T})^T, (\boldsymbol{V}^{(3)^T} \odot \boldsymbol{U}^{(3)^T})^T]\!] \in \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}\bullet\boldsymbol{r}'}, \tag{4.15}$$

where $\mathcal{E}$ is the reshaped tensor product of $\mathcal{C}$ and $\mathcal{D}$, i.e. $e_{(il)(jm)(km)} = c_{ijk}\, d_{lmn}$. The costs for computing (4.15) are therefore $\mathcal{O}(\sum_{p=1}^{3} n_p r_p r'_p + \prod_{p=1}^{3} r_p r'_p)$. The new Tucker tensor has ranks $\boldsymbol{r}\bullet\boldsymbol{r}'$. It is therefore practical to re-compress the original cores before building the tensor-product; also re-compression of (4.15) is highly recommended if further operations with $\mathcal{X} \bullet \mathcal{Y}$ are planned. Indeed, in practice often very effective re-compression of the core $\mathcal{E}$ and/or Hadamard product itself is observed. Nevertheless, compression of the cores beforehand may lead to loss of accuracy.

Another possibility to compute element-wise operations on Tucker tensors is the use of *approx-*

imate iterations [60, 61]. For instance, the element-wise reciprocal of each entry of a tensor could be calculated by a *Newton-Schultz iteration* [converges in exact arithmetics for entries in $(0,2)$]

$$\mathcal{Y}_{j+1} = \mathcal{T}\big(\mathcal{Y}_j + \mathcal{T}(\mathcal{Y}_j \bullet (\mathbf{1} - \mathcal{X} \bullet \mathcal{Y}_j))\big), \tag{4.16}$$

with $\mathcal{Y}_0 = \mathbf{1}$ the tensor with all entries equal to one, and $\mathcal{T}$ a *truncation operator* realized e.g. by the HOOI algorithm (Alg. 6) if $\mathcal{X}$ is in Tucker format.

In contrast to canonical tensors the existence of a solution to the Tucker tensor approximation problem can be ensured:

Definition 6. *Let (V, d) be a metric space and $\emptyset \neq U \subseteq V$. A best approximation of $v \in V$ in U is an element $u^* \in U$ such that*

$$d(u^*, v) = d(U, v) := \inf\{d(u, v) :\ u \in U\}, \tag{4.17}$$

i.e. the infimum is attained in U.

If V is a normed vector space the condition (4.17) reads: $\| v - u^* \| \leq \| v - u \|$ for all $u \in U$.

Lemma 2. *Let $(V, \| \,.\, \|)$ be a normed vector space with dim$(V) < \infty$ and $\emptyset \neq U \subseteq V$ a closed subset. Then, for all $v \in V$ there exists an element $u_{best} \in U$ with $\| v - u_{best} \| \leq \| v - u \|$ for all $u \in U$.*

Proof. *Let $v \in V$ and $\widetilde{u} \in U$ and define $D := U \cap \{u \in V :\ \|v - u\| \leq \| v - \widetilde{u} \|\} \subseteq U$. Since D is non-empty ($\widetilde{u} \in D$), bounded (triangle inequality) and closed (D is intersection of two closed sets) and hence compact, the continuity of the function $f : D \to \mathbb{R}$, $f(u) := \| v - u \|$ together with the extreme value theorem*[2] *ensures that f attains its minimal value at $u_{best} \in D \subseteq U$.* $\square$

The set $\mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$ is closed [3] due to the fact that the set of matrices of rank $\leq r$ is closed, i.e. for a sequence of matrices $(\boldsymbol{U}_n)_{n\in\mathbb{N}}$ with ranks $\leq r$ one has $\lim_{n\to\infty} \boldsymbol{U}_n =: \boldsymbol{U}$ has rank $\leq r$. Thus, if a sequence $(\mathcal{X}_n)_{n\in\mathbb{N}} \subset \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}} \subset \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$ is assumed, one gets $\lim_{n\to\infty} \operatorname{rank}(\mathcal{X}_{n(j)}) \leq r_j$.
In finite dimension, the closedness of the subset $\mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$ of a normed vector space, e.g. $(\bigotimes_{p=1}^{3} \mathbb{R}^{n_p}, \| \,.\, \|_F)$, ensures the existence of a *best-approximation* $\mathcal{X}_{\text{best}} \in \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$ of an element in $\mathcal{X} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$, i.e.

$$\| \mathcal{X} - \mathcal{X}_{\text{best}} \|_F \leq \| \mathcal{X} - \mathcal{Y} \|_F \quad \forall \mathcal{Y} \in \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}, \tag{4.18}$$

see Lemma 2.

An approximation of a given tensor to prescribed j-ranks $\boldsymbol{r}$ was investigated in [62], where

---

[2] The image of compact sets under continuous functions is compact.

[3] This also holds for order-d tensors where $d > 3$ and can be proved along the same lines.

the described algorithm to compute such an approximation (*Higher Order Singular Value Decomposition* [HOSVD]) works by truncating the SVD of the j-mode unfoldings. The resulting tensor is an approximation to the best-approximation in $\mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$. Indeed, due to Property 10 in [62] one has for the HOSVD approximation $\mathcal{X}_{\text{HOSVD}}$ of a tensor $\mathcal{X}$ with j-ranks R_j and the descending ordered singular values of the j-th unfolding of $\mathcal{X}$ denoted with $\sigma_k^{(j)}$, $k = 1 \dots R_j$ (here formulated for order-3 tensors)

$$\| \mathcal{X} - \mathcal{X}_{\text{HOSVD}} \|_F \leq \sqrt{\sum_{j=1}^{3} \sum_{k=r_j+1}^{R_j} \sigma_k^{(j)2}} \leq \sqrt{3} \, \| \mathcal{X} - \mathcal{X}_{\text{best}} \|_F \, , \tag{4.19}$$

where $\mathcal{X}_{\text{best}}$ is the best approximation of $\mathcal{X}$ in $\mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$, cf. [59].

Existence of the Tucker approximation was also investigated in [63], as well as alternating least squares methods (ALS) for the fitting problem

$$\min_{\mathcal{C},\boldsymbol{U}^{(1)},\boldsymbol{U}^{(2)},\boldsymbol{U}^{(3)}} \left\| \mathcal{X} - [\![\mathcal{C}; \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!] \right\|_F^2 \quad \text{s.t.} \quad \boldsymbol{U}^{(p)} \text{ column-wise orthonormal,} \tag{4.20}$$

where $\mathcal{X} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$ is given and $\mathcal{C} \in \bigotimes_{p=1}^{3} \mathbb{R}^{r_p}$, $\boldsymbol{U}^{(p)} \in \mathbb{R}^{n_p \times r_p}$ to be computed.

For three dimensions Tucker approximation by a black-box cross 3d algorithm was reported in [64]. There it was shown that the reconstruction of Tucker tensors with low core rank can be achieved in linear time, i.e. $O(nr)$, and with complexity $O(nr^3)$ in many other cases.

Another algorithm is the so-called *higher order orthogonal iteration* (HOOI), an ALS algorithm, [46], which can also be used for the purpose of *re-compression* (namely computing a quasi-optimal Tucker approximation of lower rank to a given Tucker tensor). Algorithmic variants of the HOOI were investigated in [65].

The HOOI is based on the following reformulation of the optimization problem (4.20):

Using the orthonormality of the factor matrices and rewriting the objective in (4.20) gives

$$\left\| \mathcal{X} - [\![\mathcal{C}; \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!] \right\|_F^2 = \|\mathcal{X}\|_F^2 + \|\mathcal{C}\|_F^2 - 2 \left\langle \mathcal{X} \times_1 \boldsymbol{U}^{(1)^T} \times_2 \boldsymbol{U}^{(2)^T} \times_3 \boldsymbol{U}^{(3)^T} , \mathcal{C} \right\rangle . \tag{4.21}$$

Its gradient w.r.t. to $\mathcal{C}$ is given as

$$\partial_{\mathcal{C}} \left\| \mathcal{X} - [\![\mathcal{C}; \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!] \right\|_F^2 = 2(\mathcal{C} - \mathcal{X} \times_1 \boldsymbol{U}^{(1)^T} \times_2 \boldsymbol{U}^{(2)^T} \times_3 \boldsymbol{U}^{(3)^T}), \tag{4.22}$$

which attains zero for

$$\mathcal{C} = \mathcal{X} \times_1 \boldsymbol{U}^{(1)^T} \times_2 \boldsymbol{U}^{(2)^T} \times_3 \boldsymbol{U}^{(3)^T}, \tag{4.23}$$

giving a necessary condition for the optimal choice of the core.

By inserting (4.23) into (4.21) problem (4.20) can be re-casted as a maximization problem [46], [66], i.e.

$$\max_{\boldsymbol{U}^{(1)},\boldsymbol{U}^{(2)},\boldsymbol{U}^{(3)}} \left\| \mathcal{X} \times_1 \boldsymbol{U}^{(1)^T} \times_2 \boldsymbol{U}^{(2)^T} \times_3 \boldsymbol{U}^{(3)^T} \right\|_F^2 \quad \text{s.t.} \quad \boldsymbol{U}^{(p)} \text{ column-wise orthonormal.} \tag{4.24}$$

An alternating least squares approach for solving (4.24) can easily be derived by alternately fixing all but one factor matrix and solving for the remaining by an SVD-approach, see [46] and Alg. 6. In the description of Alg. 6 assume the (common) convention of descending ordered singular values. It is enough to compute the so-called *economic sized* svd, namely, in the case $\prod_{i\neq p} r_i < n_p$ only the first $\prod_{i\neq p} r_i$ columns of $\boldsymbol{U}$ have to be computed and $\boldsymbol{\Sigma} \in \mathbb{R}^{\prod_{i\neq p} r_i \times \prod_{i\neq p} r_i}$; analog for the case $\prod_{i\neq p} r_i \geq n_p$.

Algorithm 6 HOOI (ALS); `HOOI`$(\mathcal{X}, \boldsymbol{r}_0, \epsilon)$

```
Require: $\mathcal{X} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$, $\epsilon > 0$, $r_{p,0} \in \mathbb{N}_{>1}$, $p = 1 \dots 3$ (initial guesses for $j$-ranks)
Ensure: $\mathcal{C} \in \bigotimes_{p=1}^{3} \mathbb{R}^{r_p}$, $\boldsymbol{U}^{(p)} \in \mathbb{R}^{n_p \times r_p}$, $p = 1 \dots 3$
Initialize $\boldsymbol{U}^{(p)} \in \mathbb{R}^{n_p \times r_{p0}}$, $p = 1 \dots 3$ e.g. by HOSVD or random
repeat
    for $p = 1 \dots 3$ do
        $\mathcal{Y} \leftarrow \mathcal{X} \times_{\substack{j=1 \\ j\neq p}}^{3} \boldsymbol{U}^{(j)^T}$
        $(\boldsymbol{U}, \Sigma) \leftarrow \text{svd}(\mathcal{Y}_{(p)})$   (where $\mathbb{R}^{n_p \times \prod_{i\neq p} r_i} \ni \mathcal{Y}_{(p)} = \boldsymbol{U\Sigma V}$)
        $r_p \leftarrow \min\{r \mid \sqrt{\sum_{i>r} \sigma_i^2} < \frac{\epsilon}{\sqrt{3}} \|\Sigma\|_F\}$
        $\boldsymbol{U}^{(p)} \leftarrow \boldsymbol{U}(:, 1 : r_p) = [\boldsymbol{u}_1, \dots, \boldsymbol{u}_{r_p}]$
    end for
until fit ceases to improve
$\mathcal{C} \leftarrow \mathcal{X} \times_1 \boldsymbol{U}^{(1)^T} \times_2 \boldsymbol{U}^{(2)^T} \times_3 \boldsymbol{U}^{(3)^T}$
```

The SVD is truncated in such a way that the relative error in the Frobenius norm is smaller than the given tolerance of $\epsilon/\sqrt{3}$.

The mode multiplications in Alg. 6, namely $\mathcal{X} \times_{\substack{j=1 \\ j\neq p}}^{3} \boldsymbol{U}^{(j)^T}$, need most of the computing time, since they scale with $\mathcal{O}(rn^3)$, whereas the SVD needs $\mathcal{O}(r^2 n)$ (here $n_q \equiv n$ and $r_q \equiv r$ is assumed).

HOOI converges to a solution where the objective function of (4.21) ceases to decrease; in fact, the convergence of HOOI to a global optimum is not guaranteed, not even to stationary points, [66],[63]. Nevertheless, to the author's experience Alg. 6 works well in practice and most of the time yields better results than HOSVD (even for random initialization).

An efficient generalization of Alg. 6 to re-compression, i.e. the case where $\mathcal{X}$ is already in Tucker format, is straight forward and can be realized by using the formulas for mode-multiplication and matricization within the Tucker format from [49].

HOOI can also be used for approximate addition of Tucker tensors. Assume $\mathcal{X}$ is a sum of two Tucker tensors with equal mode sizes (Block CP [BCP], [46],[67]), i.e.

$$\mathcal{X} = [\![\, \mathcal{D};\, \boldsymbol{V}^{(1)}, \boldsymbol{V}^{(2)}, \boldsymbol{V}^{(3)} \,]\!] + [\![\, \mathcal{E};\, \boldsymbol{W}^{(1)}, \boldsymbol{W}^{(2)}, \boldsymbol{W}^{(3)} \,]\!]. \tag{4.25}$$

Mode multiplications and matricization have to be performed with respect to the BCP format, which can be done element-wise (summand by summand).

The main advantage of the Tucker decomposition over the CP format is the available SVD-based quasi best approximation. At least in three dimensions the exponential scaling of the dimension in the storage of the core tensor plays a secondary role. If, though, a canonical decomposition is needed, the 2-stage approximation 'Tucker to CP' might be considered: Assume the core tensor in (4.13) is approximated in the canonical tensor format, i.e.

$$\mathcal{C} = [\![\lambda;\, \boldsymbol{V}^{(1)}, \boldsymbol{V}^{(2)}, \boldsymbol{V}^{(3)}]\!] \in \mathcal{C}_{\boldsymbol{r},R}, \tag{4.26}$$

with $\boldsymbol{V}^{(j)} \in \mathbb{R}^{r_j \times R}$. Then, one can easily transform Eq. (4.13) into CP format for a cost of $O(R\, \sum_{j=1}^{3} r_j\, n_j)$ operations by

$$\mathbf{A} = [\![\lambda;\, \boldsymbol{U}^{(1)}\boldsymbol{V}^{(1)}, \boldsymbol{U}^{(2)}\boldsymbol{V}^{(2)}, \boldsymbol{U}^{(3)}\boldsymbol{V}^{(3)}]\!] \in \mathcal{C}_{\boldsymbol{n},R}. \tag{4.27}$$

4.4 Relation between Tucker Tensors and Tensor Trains (TT) in 3 Dimensions

Definition 7. *A tensor $\mathcal{X} \in \bigotimes_{p=1}^{d} \mathbb{R}^{n_p}$ is called a Tensor Train (TT) [50] if its entries are given as*

$$x_{i_1 i_2 \ldots i_d} = \boldsymbol{G}_1(i_1)\boldsymbol{G}_2(i_2)\ldots\boldsymbol{G}_d(i_d), \tag{4.28}$$

where $\boldsymbol{G}_k(i_k)$ is an $r_{k-1} \times r_k$ matrix with $r_0 = r_d = 1$.

The $\boldsymbol{G}_k$ are actually three dimensional arrays with sizes $r_{k-1} \times n_k \times r_k$. Writing out the products in (4.28) leads to

$$x_{i_1 i_2 \ldots i_d} = \sum_{\alpha_1,\ldots,\alpha_{d-1}} \boldsymbol{G}_1(i_1, \alpha_1)\, \boldsymbol{G}_2(\alpha_1, i_2, \alpha_2)\, \ldots\, \boldsymbol{G}_{d-1}(\alpha_{d-2}, i_{d-1}, \alpha_{d-1})\, \boldsymbol{G}_d(\alpha_{d-1}, i_d). \tag{4.29}$$

For dense tensors $\mathcal{X}$ there exists quasi-best approximation in TT format due to best rank-r_k approximation of the unfolding matrices of $\mathcal{X}$ [50]. Re-compression (*rounding*) and 'canonical tensor to tensor train' (CP2TT) is also possible [50], as well as black-box approximation by

ACA type algorithms [68].

In three dimensions (4.29) reduces to

$$x_{i_1 i_2 i_3} = \sum_{\alpha_1,\alpha_2} \boldsymbol{G}_1(i_1,\alpha_1)\boldsymbol{G}_2(\alpha_1,i_2,\alpha_2)\boldsymbol{G}_3(\alpha_2,i_3). \tag{4.30}$$

From (4.30) one concludes

$$\mathcal{X} = \mathcal{G} \times_1 \boldsymbol{G}_1 \times_3 \boldsymbol{G}_3^T \equiv [\![\mathcal{G}; \boldsymbol{G}_1, \mathrm{id}, \boldsymbol{G}_3^T]\!], \tag{4.31}$$

where $\mathcal{G}$ denotes the 3-tensor $\mathcal{G}_2 \in \bigotimes_{p=1}^{3} \mathbb{R}^{m_p}$, where $m_p = r_p$, $p \neq 2$ and $m_2 = n_2$. Recompression of (4.31) leads to a (r_1', r_2', r_3')-Tucker tensor. The representation (4.31) is also known as *Tucker2* decomposition of an order three tensor [46].

On the other hand a Tucker tensor can also be easily converted to a tensor train, i.e. by mode-multiplication of one factor matrix (e.g. the second factor) one immediately gets the form (4.31). Re-compression by TT-rounding can be done afterwards. The Hadamard product (as well as many other arithmetic operations) can be carried out efficiently in TT-format, [50].

4.5 Discrete Fourier Transform for Structured Tensors

Again, the following is presented for the case of order-3 tensors, although everything is also valid for the general case of order-d.

For a given tensor $\mathcal{X} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$ the 3-d *discrete Fourier Transform* (DFT) results in the complex tensor $\widehat{\mathcal{X}} \in \bigotimes_{p=1}^{3} \mathbb{C}^{n_p}$ and is defined as

$$\widehat{x}_{k_1 k_2 k_3} = \sum_{j_1=1}^{n_1} \omega_{n_1}^{(k_1-1)(j_1-1)} \sum_{j_2=1}^{n_2} \omega_{n_2}^{(k_2-1)(j_2-1)} \sum_{j_3=1}^{n_3} \omega_{n_3}^{(k_3-1)(j_3-1)} x_{j_1 j_2 j_3}, \tag{4.32}$$

where $\omega_{n_p}^{k_p j_p} = e^{-2\pi i \frac{k_p j_p}{n_p}}$.

The inverse discrete Fourier Transformation (IDFT) is given by

$$x_{j_1 j_2 j_3} = \frac{1}{n_1 n_2 n_3} \sum_{k_1=1}^{n_1} \omega_{n_1}^{-(k_1-1)(j_1-1)} \sum_{k_2=1}^{n_2} \omega_{n_2}^{-(k_2-1)(j_2-1)} \sum_{k_3=1}^{n_3} \omega_{n_3}^{-(k_3-1)(j_3-1)} \widehat{x}_{k_1 k_2 k_3}. \tag{4.33}$$

FFT variants are commonly used because of their almost linear scaling, i.e. $O(\sum_{p=1}^{3} \log(n_p) \prod_{q=1}^{3} n_q)$. The convolution theorem holds, i.e. in multi-index notation and using the operator symbols $\mathcal{F}$ and $\mathcal{F}^{-1}$ for the DFT and IDFT respectively

$$\mathcal{X} * \mathcal{K}_{\mathbf{n}} = \mathcal{F}^{-1}(\mathcal{F}(\mathcal{X}) \bullet \mathcal{F}(\mathcal{K})), \tag{4.34}$$

where the subscript denotes the **n**- shift and $\bullet$ the element-wise product (*Hadamard product*). The following is straight forward to show.

Lemma 3. *For a canonical tensor* $\mathcal{X} = [\![\boldsymbol{\lambda};\ \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!] \in \mathcal{C}_{\boldsymbol{n},r}$, $x_{j_1j_2j_3} = \sum_{l=1}^{r} \lambda_j\, u^{(1)}_{j_1l}\, u^{(2)}_{j_2l}\, u^{(3)}_{j_3l}$ *the DFT is given by*

$$\widehat{\mathcal{X}} = [\![\boldsymbol{\lambda};\ \mathcal{F}_{1d}(\boldsymbol{U}^{(1)}), \mathcal{F}_{1d}(\boldsymbol{U}^{(2)}), \mathcal{F}_{1d}(\boldsymbol{U}^{(3)})]\!], \tag{4.35}$$

where the (1-d) Fourier transform $\mathcal{F}_{1d}$ *is only taken along each column of a factor matrix. The IDFT is given as*

$$\mathcal{X} = [\![\boldsymbol{\lambda};\ \mathcal{F}_{1d}^{-1}(\boldsymbol{U}^{(1)}), \mathcal{F}_{1d}^{-1}(\boldsymbol{U}^{(2)}), \mathcal{F}_{1d}^{-1}(\boldsymbol{U}^{(3)})]\!]. \tag{4.36}$$

Proof.

$$\widehat{x}_{k_1k_2k_3} = \sum_{l=1}^{r} \lambda_r \Big(\sum_{j_1=1}^{n_1} \omega_{n_1}^{(k_1-1)(j_1-1)}\, u^{(1)}_{j_1l} \Big) \Big(\sum_{j_2=1}^{n_2} \omega_{n_2}^{(k_2-1)(j_2-1)}\, u^{(2)}_{j_2l} \Big) \Big(\sum_{j_3=1}^{n_3} \omega_{n_3}^{(k_3-1)(j_3-1)}\, u^{(3)}_{j_3l} \Big) \tag{4.37}$$

and analog for the IDFT. □

Lemma 4. *For a Tucker tensor* $\mathcal{X} = [\![\boldsymbol{C};\ \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!] \in \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$, $x_{j_1j_2j_3} = \sum_{j_1',j_2',j_3'=1}^{r_1,r_2,r_3} c_{j_1'j_2'j_3'}\, u^{(1)}_{j_1j_1'}\, u^{(2)}_{j_2j_2'}\, u^{(3)}_{j_3j_3'}$ *the DFT is given by*

$$\widehat{\mathcal{X}} = [\![\boldsymbol{C};\ \mathcal{F}_{1d}(\boldsymbol{U}^{(1)}), \mathcal{F}_{1d}(\boldsymbol{U}^{(2)}), \mathcal{F}_{1d}(\boldsymbol{U}^{(3)})]\!], \tag{4.38}$$

where the (1-d) Fourier transform $\mathcal{F}_{1d}$ *is only taken along each column of a factor matrix. The IDFT is given as*

$$\mathcal{X} = [\![\boldsymbol{C};\ \mathcal{F}_{1d}^{-1}(\boldsymbol{U}^{(1)}), \mathcal{F}_{1d}^{-1}(\boldsymbol{U}^{(2)}), \mathcal{F}_{1d}^{-1}(\boldsymbol{U}^{(3)})]\!]. \tag{4.39}$$

Proof.

$$\widehat{x}_{k_1k_2k_3} = \sum_{j_1',j_2',j_3'=1}^{r_1,r_2,r_3} c_{j_1'j_2'j_3'} \Big(\sum_{j_1=1}^{n_1} \omega_{n_1}^{(k_1-1)(j_1-1)}\, u^{(1)}_{j_1j_1'} \Big) \Big(\sum_{j_2=1}^{n_2} \omega_{n_2}^{(k_2-1)(j_2-1)}\, u^{(2)}_{j_2j_2'} \Big) \Big(\sum_{j_3=1}^{n_3} \omega_{n_3}^{(k_3-1)(j_3-1)}\, u^{(3)}_{j_3j_3'} \Big) \tag{4.40}$$

and analog for the IDFT. □

The DFT as well as the IDFT for structured tensors is therefore basically 1-dimensional, more precise, assuming FFT and IFFT implementations, the costs are $\mathcal{O}(R \sum_p n_p \log n_p)$ for canonical tensors and $\mathcal{O}(\sum_p r_p n_p \log n_p)$ for Tucker tensors respectively. This matches asymptotically the costs for 1-dimensional (I)FFT times the rank constants.

4.6 Approximation of Disturbed Data

Assume a tensor $\mathcal{R} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$ that is composed by a Tucker tensor $\mathcal{X} \in \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$ and an additive noise $\mathcal{N} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$, i.e.

$$\mathcal{R} = \mathcal{X} + \mathcal{N}. \tag{4.41}$$

The problem of approximating $\mathcal{X}$ in Tucker format is also referred to as *'tensor filtering problem'* [69]. Note that, in general, the disturbed data tensor $\mathcal{R}$ can have full mode ranks, although $\mathcal{X}$ has ranks $\boldsymbol{r} <_c \boldsymbol{n}$. A method for Tucker approximation which seeks an approximation with a prescribed error bound of magnitude of the noise, like for instance the HOOI Alg. 6, would approximate the full-rank tensor $\mathcal{R}$, i.e. one would fail to derive a lower rank approximation. Here, an algorithm is introduced that first filters the noise and then approximates the filtered tensor, i.e. a low rank approximation of the given tensor $\mathcal{R}$ is achieved with accuracy in the order of the noise. The method uses the Fourier approach of the previous section and has the same complexity as Alg. 6. To the author's knowledge the proposed method has not been introduced in the literature yet.

For the sake of simplicity, *Gaussian functions* serve as filters for the method, i.e.

$$g_b^{(q)}(x) = \frac{1}{\sqrt{\pi b}} \exp\Big(-\frac{(x - n_q/2)^2}{b}\Big), \quad b := \frac{2am}{\pi(2a-1)}, \; m \geq 1, \; a \in \mathbb{N}\,(a \geq 2) \tag{4.42}$$

where a is an over-sampling factor and m a cut-off parameter. A rank-1 tensor is now defined by the tensor product of g, i.e.

$$\mathcal{G}_b := \boldsymbol{g}_b^{(1)} \circ \boldsymbol{g}_b^{(2)} \circ \boldsymbol{g}_b^{(3)}, \tag{4.43}$$

where $\boldsymbol{g}^{(q)} = (g^q(i))_{i=1\dots n_q}$. Also other fast decaying functions could be considered, e.g. *Kaiser-Bessel functions*, see section 8.5.2.

The simple idea is now summarized in Alg. 7. Due to the fast decaying Gaussians, the convolution in Alg. 7 can be performed locally by only using neighboring data points, see section 8.5.1 for a detailed description of the more general case of gridding unstructured data onto a tensor grid. The complexity of this step is therefore linear in the tensor grid size. The 1-d FFTs are performed with zero-padding to the size an. Also note that the Hadamard product with the rank-1 tensor $\widehat{\mathcal{G}}^{-1}$ can be performed without increasing the rank of $\widetilde{\mathcal{R}}$. Overall, the dominating complexity of Alg. 7 is the `HOOI` part, see Alg. 6.

Tab. 4.1 shows results for Alg. 7 for the case of $\max_j |n_j| =: \|\mathcal{N}\|_\infty$ = 1e-06 and ϵ = 1e-08. $\mathcal{X}$ was chosen randomly with ranks $r_q \equiv 10$, whereas $n_q \equiv 40$. A direct approximation by `HOOI` leads to full ranks, i.e. $r_q \equiv 40$; `HOOId`, on the other hand, yields lower rank approximations.

Algorithm 7 HOOI (ALS) for disturbed data; `HOOId`$(\mathcal{R}, \boldsymbol{r}_0, \epsilon, b)$

```
Require: $\mathcal{R} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$,  $\epsilon > 0$,  $r_{p,0} \in \mathbb{N}_{>1}$,  $p = 1 \dots 3$;  $b > 0$
Ensure: $\widetilde{\mathcal{R}} \in \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$
    Precompute:
    $\widehat{\mathcal{G}}_b \leftarrow$ FFT of $\mathcal{G}_b$ using Lemma 3
    Compute:
        • Convolve (locally): $\widetilde{\mathcal{R}} \leftarrow \mathcal{R} * \mathcal{G}_b$
        • Approximate: $\widetilde{\mathcal{R}} \leftarrow$ HOOI$(\mathcal{R}, \boldsymbol{r}_0, \epsilon)$
        • Compute the FFT: $\widetilde{\mathcal{R}} \leftarrow \mathcal{F}(\widetilde{\mathcal{R}})$ using Lemma 4
        • Deconvolve: $\widetilde{\mathcal{R}} \leftarrow \widetilde{\mathcal{R}} \bullet \widehat{\mathcal{G}}_b^{-1}$
        • Compute IFFT: $\widetilde{\mathcal{R}} \leftarrow \mathcal{F}^{-1}(\widetilde{\mathcal{R}})$ using Lemma 4
```

Table 4.1: Results for Alg. 7 for the case of $\max_j |n_j| =: \|\mathcal{N}\|_\infty = 1\text{e-}06$ and $\epsilon = 1\text{e-}08$. $\mathcal{X}$ was chosen randomly with ranks $r_q \equiv 10$, whereas $n_q \equiv 40$. The over-sampling parameter was $a = 2$. The first column gives the parameter m from (4.42).

m	(r_1, r_2, r_3)	$\left\|\widetilde{\mathcal{R}} - \mathcal{R}\right\|_F / \|\mathcal{R}\|_F$	$\left\|\widetilde{\mathcal{R}} - \mathcal{R}\right\|_\infty$
3	$(26, 30, 31)$	1.0e-7	2.0e-6
5	$(19, 21, 21)$	1.3e-7	3.0e-6
8	$(14, 15, 16)$	1.6e-7	4.9e-6
12	$(11, 12, 11)$	2.6e-7	9.4e-6
18	$(10, 10, 10)$	5.5e-7	3.2e-5

Chapter 5

Stray Field Computation on Tensor Grids

Portions of this chapter were previously published as [25] or submitted for publication as [26] and have been reproduced here with permission of the co-authors. Content which was not generated by the author of this thesis is explicitly denoted.

In this chapter a method is introduced and mathematically analyzed that allows computing the stray field for given magnetization configuration on a tensor grid. The described method is capable to efficiently use/profit from structured tensor representation of the magnetization components, e.g. CP, Tucker or TT formats, see section 4. The resulting complexity strongly depends on the format and, within this, of the ranks of the representation. Also it makes a difference whether the result itself is stored in a (data sparse) structured format or as full/dense tensor. In the tensor structured case, the method scales below linear in the tensor grid size $N = n^3$ with a constant that depends on the format and rank(s). In the case of re-conversion into full format the method scales linear in the grid size. A Fast Fourier transform (FFT) variant that even accelerates the method is also discussed. In the degenerated case of dense magnetization component tensors the costs are $\mathcal{O}(N^{4/3})$, slightly above the optimal linear scaling.
Section 5.1 is meant to be an easily readable description of the method, which is numerically tested in section 5.2. Moreover, the method is mathematically analyzed in detail in section 5.3. The FFT version is presented in section 5.4.

5.1 Description of the Method

It follows the principal description of the stray field method, while a detailed mathematical analysis is performed in section 5.3.

5.1.1 Separable Representation of Multivariate Functions

Assume a multivariate function $f = f(\rho) : \mathbb{R}^d \to \mathbb{R}$ where $\rho = \|\boldsymbol{x}\|^2 \in [a, b] \subset \mathbb{R}$ and the integral representation

$$f(\rho) = \int_{\mathbb{R}} g(\tau)\, e^{\rho h(\tau)}\, \mathrm{d}\tau. \tag{5.1}$$

If quadrature with nodes and weights (t_k, ω_k), $k = 1 \dots R$ is applied to (5.1), one obtains a separable representation of f, i.e.

$$f(\rho) \approx \sum_{k=1}^{R} \omega_k\, g(t_k)\, e^{\rho h(t_k)} = \sum_{k=1}^{R} \omega_k\, g(t_k) \prod_{p=1}^{d} e^{x^{(p)2} h(t_k)}. \tag{5.2}$$

The quadrature order R refers to the *separation rank* of (5.2).

The following methods makes use of this observation. So called *sinc-quadrature* is used, see section 5.1.3 and 5.3.3.

5.1.2 Analytical Preparations

Integration by parts of the integral representation of the scalar potential, see section 2.3.1 Eqn. (2.23), yields

$$\phi(\boldsymbol{x}) = -\frac{1}{4\pi} \int_{\Omega} \boldsymbol{m}(\boldsymbol{y}) \cdot \frac{\boldsymbol{x} - \boldsymbol{y}}{\|\boldsymbol{x} - \boldsymbol{y}\|^3} \mathrm{d}\boldsymbol{y}. \tag{5.3}$$

This expression makes sense in micromagnetics due to the constraint $\|\boldsymbol{m}\| = 1$ a.e. in Ω [7], which implies $\boldsymbol{m} \in L^\infty(\Omega)$. Since the kernels $\kappa^{(q)}(\boldsymbol{x}) := x^{(q)} / \|\boldsymbol{x}\|^3 \in L^p(B_R(\boldsymbol{0}))$ for balls $B_R(\boldsymbol{0})$ centered in $\boldsymbol{0}$ with $R > 0$ and $p \in [1, 3/2)$, Hölder's inequality ensures that (5.3) is well-defined.

Next, denote the three volume integrals in Eqn. (5.3) by

$$I^{(p)}(\boldsymbol{x}) = \int_{\Omega} m^{(p)}(\boldsymbol{y})\, \frac{x^{(p)} - y^{(p)}}{\|\boldsymbol{x} - \boldsymbol{y}\|^3}\, \mathrm{d}\boldsymbol{y}, \tag{5.4}$$

for each of the components $m^{(p)}$, $p = 1 \dots 3$, of the magnetization $\boldsymbol{m}$. Thus, Eqn. (5.3) reads

$$\phi(\boldsymbol{x}) = -\frac{1}{4\pi} \sum_{p=1}^{3} I^{(p)}(\boldsymbol{x}). \tag{5.5}$$

The crucial step is to represent the integral kernel in (5.4) as an integral of a *Gaussian function* by the formula

$$\frac{1}{\rho^{\frac{3}{2}}} = \frac{2}{\sqrt{\pi}} \int_{\mathbb{R}} \tau^2 \, e^{-\tau^2 \rho} \, d\tau. \tag{5.6}$$

So one has, for $\rho = \|\boldsymbol{x} - \boldsymbol{y}\|^2$,

$$I^{(p)}(\boldsymbol{x}) = \frac{2}{\sqrt{\pi}} \int_{\mathbb{R}} \tau^2 \int_{\Omega} e^{-\tau^2 \|\boldsymbol{x}-\boldsymbol{y}\|^2} m^{(p)}(\boldsymbol{y})(x^{(p)} - y^{(p)}) \, \mathrm{d}\boldsymbol{y} \, \mathrm{d}\tau. \tag{5.7}$$

Eqn. (5.7) reduces the computation to independent integrals along each principal direction, because the integrand is a separable function, i.e.

$$e^{-\tau^2 \|\boldsymbol{x}-\boldsymbol{y}\|^2} = \prod_{q=1}^{3} e^{-\tau^2 \, (x^{(q)} - y^{(q)})^2}. \tag{5.8}$$

As will be seen later, this results in a reduction of computational effort from $O(N^2)$ to $O(N^{4/3})$, where $N = n^3$.

However, in order to be able to use Eqn. (5.7), one first has to apply stable quadrature for the $\tau-$integration.

Remark 2. *In general, there holds the formula*

$$\int_0^{\infty} x^n e^{-\rho x^2} \, dx = \frac{\Gamma(\frac{n+1}{2})}{2\rho^{\frac{n+1}{2}}}, \; \rho > 0, \; n > -1. \tag{5.9}$$

Thus, for the kernel $1/\|\boldsymbol{x}\|^k$, $k \in \mathbb{N} \cup \{0\}$ *the choice* $n = k-1$ *leads to a separable representation after the substitution* $\rho = \|\boldsymbol{x} - \boldsymbol{y}\|^2$.

5.1.3 τ-Integration

Although the singularity is gone, the numerical treatment of (5.7) is not straightforward. Small values of ρ have a dispersive effect on the integrand in (5.6), so one has to distribute the quadrature nodes over a wide range for accurate approximation of the kernel function $1/\rho^{3/2}$. These values of ρ correspond to a small grid size, which is again essential for an accurate approximation of the magnetic scalar potential. Therefore one has to use a quadrature rule that is robust with respect to small values of ρ.

For *Gaussian quadrature*, weights and nodes can be computed by solving an eigenvalue problem of symmetric tridiagonal type. For a larger number of quadrature points one uses the QR algorithm, which scales linearly in the number of quadrature terms. On the other hand, using Gaussian quadrature formulas on a finite subinterval $[0, A]$, or *Gauss-Laguerre quadrature* over the infinite interval in (5.6), fail in terms of achieving a sufficiently good representation of the kernel function for small values of ρ.

In [70] an integral representation for the Newton potential $1/\|\boldsymbol{x}\|$ was used to compute the electrostatic potential for given Gaussian distributed charges in whole space. The τ - integration was performed using the Gauss-Legendre formula on logarithmically scaled blocks of the interval $[0, 10^4]$ using a total of 120 quadrature points. Due to geometrical refinement against zero, the functional $1/\rho$ is well described in this region.

Here we use the exponentially convergent *Sinc-quadrature* for numerical integration of the integral [57]. This method shows better approximation properties than the Gauss-Legendre formula for much fewer quadrature terms. See Fig. 5.1, where Sinc-, Gauss-Laguerre- and Gauss-Legendre - quadrature are compared for the integration of (5.6).

Abbreviating notation and exploiting symmetry in τ, the $I^{(p)}$ take the form

$$I^{(p)}(\boldsymbol{x}) = \frac{2}{\sqrt{\pi}} \int_0^\infty \tau^2\, 2\, F^{(p)}(\boldsymbol{x}, \tau)\, d\tau, \tag{5.10}$$

where $F^{(p)}$ stands for the Ω - integral in (5.7). In order to guarantee the above mentioned exponential convergence rate one has to perform an integral transform (also see section 5.3) on Eq. (5.10), i.e. $\tau = \sinh(t)$, and apply numerical integration afterwards, which gives

$$\int_0^\infty \tau^2\, 2\, F^{(p)}(\boldsymbol{x}, \tau)\, d\tau \approx \sum_{l=1}^{R} \omega_l\, \sinh(t_l)^2\, G^{(p)}(\boldsymbol{x}, t_l), \tag{5.11}$$

where (t_l, ω_l) are the nodes and weights of the underlying quadrature, and

$$G^{(p)}(\vec{x}, t) = 2\, F^{(p)}(\boldsymbol{x}, \sinh(t)). \tag{5.12}$$

To apply *Sinc-quadrature* one uses the $R + 1$ nodes and weights given by

$$t_l = l h_R \tag{5.13}$$

and

$$\omega_l = \begin{cases} h_R & l = 0, \\ 2\, h_R \cosh(t_l) & l > 0, \end{cases} \tag{5.14}$$

Table 5.1: Average abs./rel. errors of sinc-quadrature on ρ-interval $[5e-05, 1e-02]$. Left: c_0-dependence of approximation of (5.6) ($R = 50$). Right: R-dependence of approximation of (5.6) ($c_0 = 1.85$).

c_0	abs. error	rel. error	R	abs. error	rel. error
1.70	$8.3\,e{-}01$	$3.4\,e{-}07$	35	$1.6\,e{+}00$	$6.5\,e{-}07$
1.75	$8.4\,e{-}03$	$3.3\,e{-}09$	40	$7.4\,e{-}03$	$2.9\,e{-}09$
1.80	$8.5\,e{-}06$	$3.3\,e{-}12$	45	$4.7\,e{-}06$	$3.2\,e{-}12$
1.85	$3.4\,e{-}09$	$1.2\,e{-}13$	50	$3.4\,e{-}09$	$1.2\,e{-}13$
1.90	$7.8\,e{-}09$	$3.0\,e{-}13$	55	$2.8\,e{-}10$	$1.1\,e{-}14$
1.95	$1.7\,e{-}08$	$6.7\,e{-}13$	60	$2.5\,e{-}11$	$9.5\,e{-}16$
2.00	$3.9\,e{-}08$	$1.5\,e{-}12$	65	$3.3\,e{-}12$	$1.2\,e{-}16$
2.05	$8.2\,e{-}08$	$3.0\,e{-}12$	70	$4.4\,e{-}12$	$8.9\,e{-}17$
2.10	$1.7\,e{-}07$	$6.3\,e{-}12$	75	$2.3\,e{-}12$	$9.3\,e{-}17$
2.15	$3.2\,e{-}07$	$1.2\,e{-}11$	80	$2.4\,e{-}12$	$9.1\,e{-}17$

with $h_R = c_0 \ln(R)/R$ for some appropriate c_0, see theorem 6 in section 5.3. For the sake of completeness, the algorithm for Sinc-quadrature for approximation of an integral

$$I(f) := \int_{\mathbb{R}_+} f(\xi)\, d\xi = \int_{\mathbb{R}_+} \cosh(\xi) f(\sinh(\xi))\, d\xi \tag{5.15}$$

is summarized in Alg. 8, i.e.

$$I(f) \approx \sum_{j=0}^{R} \omega_j f(\sinh(\tau_j)). \tag{5.16}$$

Tab. 5.1 shows the average absolute and relative errors due to Sinc-quadrature approximation

Algorithm 8 Sinc-Quadrature for (5.15); `sinc(R, c0)`

```
Require: $R \in \mathbb{N}$, $c_0 > 0$
Ensure: $\tau, \omega \in \mathbb{R}_+^{R+1}$
h ← $c_0 \ln(R)/R$
for $j = 0 \dots R$ do
    $\tau_j \leftarrow jh$
    $\omega_j \leftarrow 2h\cosh(jh)$
end for
$\omega_0 \leftarrow h$
```

of the functional (5.6) for 10^5 equidistantly chosen ρ-values of the interval $[5e-05, 1e-02]$. The left three columns show accuracy for different values of the parameter c_0 and $R = 50$, right columns show dependence of the number of quadrature terms and $c_0 = 1.85$.

For the numerical experiments in section 5.2, $c_0 = 1.85$ was used, which gives a sufficiently

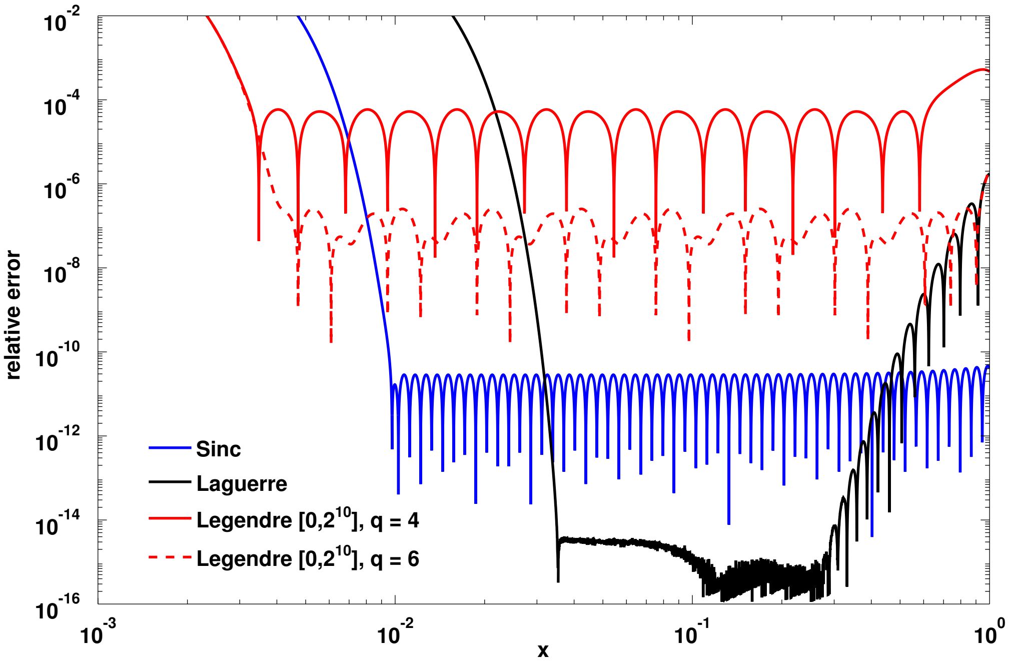

Figure 5.1: Comparison of three quadrature methods for Eq. (5.6) with $\rho = x^2$: Block-Gauss-Legendre on 10 logarithmically scaled subintervals of the interval $[0, 2^{10}] = [0, 2] \cup [2, 4] \cup \ldots \cup [2^9, 2^{10}]$ with $q = 4$ and 6 quadrature nodes each, respectively; the *Sinc*-quadrature with a total of 40 nodes and $c_0 = 1.85$ and Gauss-Laguerre quadrature with 40 nodes.

good description of the functional (5.6).

5.1.4 Description of the Magnetization on a Tensor Grid

Assume for instance a $200nm \times 100nm \times 10nm$ magnet and a grid that consists of $100 \times 80 \times 20$ cuboids, that arise from a (not necessarily equidistant) partition of the x-, y- and z-axis respectively. In general one has $n_1 \times n_2 \times n_3$ cuboids that define the grid together with three vectors $\boldsymbol{h}_p \in \mathbb{R}^{n_p}, p = 1 \ldots 3$ which consist of the grid spacings along each axis (see Fig. 5.2). The tensor grid is described by the vectors $\boldsymbol{h}_p$ in the general case or by the *grid-size vector* $\boldsymbol{n} = (n_1, n_2, n_3)$ in the equispaced case.

For the proposed method the magnetization is assumed to 'live' on the center points of the cuboids, which means the magnetization is assumed to be constant in each cell. Thus, the magnetization is given as three 3-tensors (one for each component of the magnetization), i.e.

$$\boldsymbol{m}^{(p)} = \sum_{\mathbf{j}} m_{\mathbf{j}}^{(p)} \chi_{\Omega_{\mathbf{j}}}, \quad \mathcal{M}^{(p)} \equiv (m_{\mathbf{j}}^{(p)})_{j \in \times_{q=1}^{3} \{1,\ldots,n_q\}} \in \bigotimes_{q=1}^{3} \mathbb{R}^{n_q}, \quad p = 1 \ldots 3, \tag{5.17}$$

where $\chi_{\Omega_{\mathbf{j}}} = \chi_{\Omega_{j_1}} \chi_{\Omega_{j_2}} \chi_{\Omega_{j_3}}$ is the 3-d characteristic function[1] of the subcube $\Omega_{\mathbf{j}} = \Omega_{j_1} \times \Omega_{j_2} \times \Omega_{j_3}$ and $m_{\mathbf{j}}^{(p)}$ the components of the *p-component magnetization tensor* $\mathcal{M}^{(p)}$.

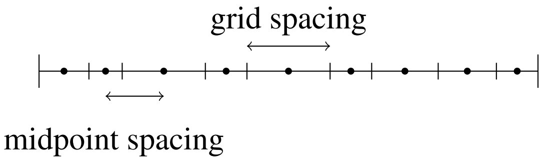

Figure 5.2: Vector of tensor grid spacings, i.e. $\boldsymbol{h}_p$

5.1.5 Computation on a Tensor Grid

The computational realization of the quadrature approximation (5.11) to Eq. (5.7) requires evaluation of the scalar potential at the center point $\boldsymbol{x}_{\mathbf{i}}^c = (x_{i_1}^c, x_{i_2}^c, x_{i_3}^c)$ of each field cell. To this end one substitutes Eqn. (5.17) into the function $G^{(p)}$ of Eq. (5.11). This leads to

$$G^{(p)}(\boldsymbol{x}_{\mathbf{i}}^c, t_l) = \sum_{\mathbf{j}} m_{\mathbf{j}}^{(p)} \prod_{q=1}^{3} \int_{\Omega} g^{(q)}(x_{i_q}^c, x', t_l) \chi_{\Omega_{j_q}}(x')\, dx', \tag{5.18}$$

where

$$g^{(q)}(\alpha, \alpha', \tau) := \begin{cases} \exp(-\sinh(\tau)^2 (\alpha - \alpha')^2) & q \neq p, \\ (\alpha - \alpha') \exp(-\sinh(\tau)^2 (\alpha - \alpha')^2) & q = p. \end{cases} \tag{5.19}$$

The three integrals in Eq. (5.18) define $(n_q \times n_q)$ - matrices, i.e.

$$\begin{aligned} d_{i_q, j_q}^l &:= \int_{\Omega_{j_q}} g^{(q)}(x_{i_q}^c, x', t_l)\, dx', \\ \boldsymbol{D}_q^l &:= (d_{i_q\, j_q}^l). \end{aligned} \tag{5.20}$$

Thus, one has a Tucker representation of the function $G^{(p)}$ (cf. section 4.3), i.e.,

$$\begin{aligned} G^{(p)}(\boldsymbol{x}_{\mathbf{i}}^c, t_l) &= \sum_{\mathbf{j}} m_{j_1\, j_2\, j_3}^{(p)}\, d_{i_1\, j_1}^l\, d_{i_2\, j_2}^l\, d_{i_3\, j_3}^l \\ &= \mathcal{M}^{(p)} \times_1 \boldsymbol{D}_1^l \times_2 \boldsymbol{D}_2^l \times_3 \boldsymbol{D}_3^l, \end{aligned} \tag{5.21}$$

with the core tensor $\mathcal{M}^{(p)}$.

---

[1] Also other tensor product basis functions could be used here.

Alg. 9 summarizes the computation of the 'Gaussian matrices' in (5.20), where Alg. 10 gives the computational scheme for computing the scalar potential for dense (unstructured) magnetization.

Algorithm 9 Gaussian matrices; `gaussmat`$(\tau_l, p, h_1, h_2, h_3)$

```
Require: $\tau_l \in \mathbb{R},\ p \in \{1, 2, 3\},\ h_p \in \mathbb{R}^{n_p}\ (p = 1 \dots 3)$
Ensure: $\boldsymbol{D}_q \in \mathbb{R}^{n_q \times n_q}\ (q = 1 \dots 3)$
Preallocate $D_q \in \mathbb{R}^{n_q \times n_q}$ for $q = 1 \dots 3$
for $q = 1 \dots 3$ do
    for $i = 1 \dots N_q$ do
        for $j = 1 \dots N_q$ do
            if $p = q$ then
                $(d_q)_{i,j} \leftarrow$ approximation of $\int_{\Omega_{j_q}} g^{(q)}(x_i^c, x', \tau_l)\, dx'$ using second expression in (5.19) via Gauss-Legendre quadrature or analytical formulas
            else
                $(d_q)_{i,j} \leftarrow$ approximation of $\int_{\Omega_{j_q}} g^{(q)}(x_i^c, x', \tau_l)\, dx'$ using first expression in (5.19) via Gauss-Legendre quadrature or analytical formulas
            end if
        end for
    end for
end for
```

For full micromagnetic simulations (e.g. energy minimization) the Gaussian matrices have to be stored only once ($3R$ matrices), since they only carry geometrical information. Instead of line 5 of Alg. 10 (computation of the Gaussian matrices in the inner loop) it is therefore better to compute them separately in the beginning of the simulation and use them as input in Alg. 10.

Algorithm 10 Scalar potential; `scpot`$(\mathcal{M}^{(1)}, \mathcal{M}^{(2)}, \mathcal{M}^{(3)}, h_1, h_2, h_3, R, c_0)$

```
Require: $\mathcal{M}^{(p)} \in \bigotimes_{q=1}^{d} \mathbb{R}^{n_q},\ p = 1 \dots 3,\ h_p \in \mathbb{R}^{n_p}\ (p = 1 \dots 3),\ R,\ c_0 > 0$
Ensure: $\Phi \in \bigotimes_{q=1}^{d} \mathbb{R}^{n_q}$
1: $(\tau, \omega) \leftarrow$ sinc$(R, c_0)$
2: Preallocate $\Phi$ with zero entries
3: for $p = 1 \dots 3$ do
4:     for $l = 1 \dots R$ do
5:         $\boldsymbol{D}_1, \boldsymbol{D}_2, \boldsymbol{D}_3 \leftarrow$ gaussmat$(\tau_l, p, h_1, h_2, h_3)$        ▹ Alg. 9
6:         $\Phi \leftarrow \Phi + \omega_l \sinh(\tau_l)^2\, \mathcal{M}^{(p)} \times_1 \boldsymbol{D}_1 \times_2 \boldsymbol{D}_2 \times_3 \boldsymbol{D}_3$
7:     end for
8: end for
9: $\Phi \leftarrow -1/(2\pi^{3/2})\Phi$
```

The complexity of Alg. 10, assuming the Gaussian matrices are already precomputed, is that of

R mode multiplications with the tensors $\mathcal{M}^{(p)}$ for $p = 1 \ldots 3$, followed by tensor addition, i.e.

$$\mathcal{O}\Big(3RN(1 + \sum_{p=1}^{3} n_p)\Big), \tag{5.22}$$

where $N = \prod_{q=1}^{3} n_q$.

If magnetization components are given in a particular tensor format, line 6 in Alg. 10 changes accordingly. In the case of the Tucker format, i.e. $\mathcal{M}^{(p)} = [\![C^{(p)}; \boldsymbol{U}_1^{(p)}, \boldsymbol{U}_2^{(p)}, \boldsymbol{U}_3^{(p)}]\!] \in \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$, line 6 reads

$$6: \quad \Phi \leftarrow \Phi + \omega_l \sinh(\tau_l)^2 C^{(p)} \times_1 \boldsymbol{D}_1 \boldsymbol{U}_1^{(p)} \times_2 \boldsymbol{D}_2 \boldsymbol{U}_2^{(p)} \times_3 \boldsymbol{D}_3 \boldsymbol{U}_3^{(p)}. \tag{5.23}$$

The complexity for the matrix products $\boldsymbol{V}_q^{(p)} = \boldsymbol{D}_q \boldsymbol{U}_q^{(p)} \in \mathbb{R}^{n_q \times r_q}$ is $\mathcal{O}(r_q n_q^2)$, whereas the mode-multiplications $C^{(p)} \times_1 \boldsymbol{V}_1 \times_2 \boldsymbol{V}_2 \times_3 \boldsymbol{V}_3$ costs $\mathcal{O}(n_1 r_1 r_2 r_3 + n_1 n_2 r_2 r_3 + n_1 n_2 n_3 r_3)$. Note that the mode-multiplications can also be performed in a different order, which might be advantageous if there is a great difference in the magnitude between the mode sizes or the ranks.

Alternatively, subsequent summands in line 6 of Alg. 10 or all summands at once can be added approximately in Tucker format using Alg. 6 from section 4.3. This reduces the costs (the mode multiplications in Alg. 6 can be carried out efficiently within the Tucker format), whereas the result Φ is a Tucker tensor. Fig. 5.3 compares those two variants of dealing with magnetization in Tucker format. Approximate summation is done with accuracy 1e-8, which yields an error in the Frobenius norm of less than 5e-7 for the scalar potential in all computations of the experiment. One observes linear scaling in $N = n^3$ for the full computation, whereas the approximate computation does not depend on n in this test. However, one can expect a scaling of at least n^2 for larger mode sizes, where the costs for mode multiplication in (5.23) (n^2-scaling) get significant in comparison to the costs of Alg. 6 (basically linear in n). Moreover, this experiment was done for constant ranks of the magnetization component tensors. In realistic simulations the ranks would also (more or less) depend on the grid size. Furthermore, the separation rank R was constant, rather than the more realistic assumption of *weak dependence* on the mesh parameter $h = 1/n$, i.e. $R = \mathcal{O}(\log h^{-1})$, compare with Fig. 5.5 in section 5.3.5 and also note that the considerations in section 8.7 are related to this discussion.

Furthermore, in the case where the magnetization components are tensor trains computational considerations like in the Tucker case are valid. Also, representation as Tucker tensor is possible, see section 4.4.

Finally, if the magnetization is given as canonical tensor, line 6 of Alg. 10 also reduces to multiplications of the factor matrices. This is again of n^2-complexity, see Fig. 1 in [25]. The

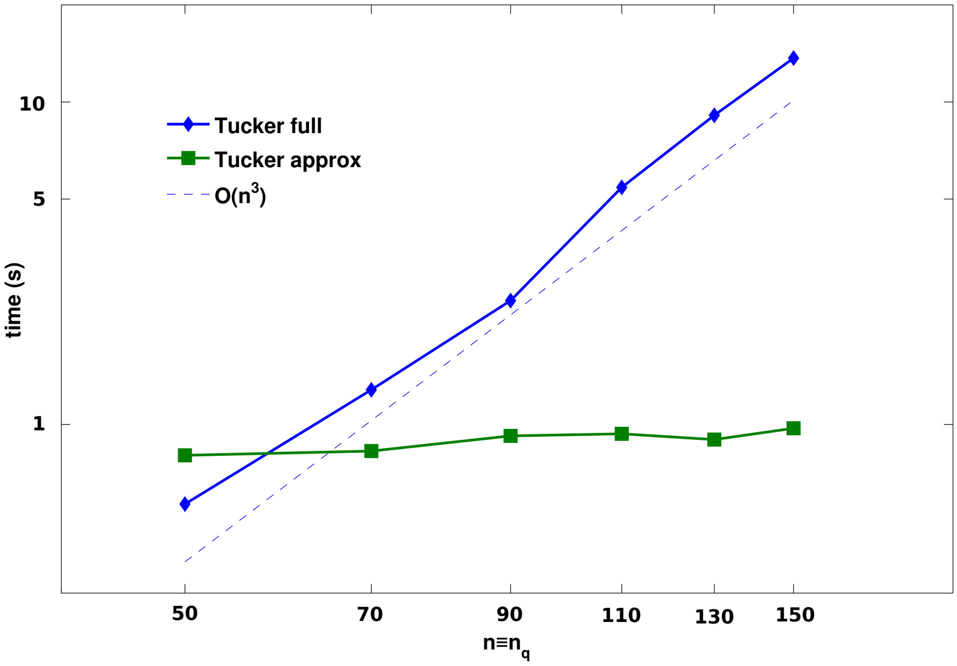

Figure 5.3: Computation time for the scalar potential by Alg. 10 for given magnetization in Tucker format (ranks constant ≡ 10); $R = 30$. 'Tucker full' indicates full ('exact') computation for Eqn. (5.23), whereas 'Tucker approx' are the times for approximately calculating Eqn. (5.23) within the Tucker format using Alg. 6 with `eps`= 1e-8 for approximate summation. Precomputation of the 'Gaussian matrices' is excluded.

resulting scalar potential can be stored as canonical tensor with rank $R' = R\sum_{p=1}^{3} r_p$, where r_p, $p = 1\ldots3$ are the ranks of the canonical representation of the magnetization components. If the full tensor is formed, additional $R'N$ operations have to be performed. An approximate summation, like in the Tucker or TT case, is not advisable, since there are no stable algorithms with error control available.

Once the potential has been computed, one has to perform discrete differentiation to obtain the field $-\nabla\phi$. This can be done by *q-mode sparse matrix multiplication*, which scales linearly in $N = n^3$ for dense potential and below linear (n^2) for tensor structured potential. Here a sparse finite-difference matrix is used which corresponds to a three-point finite-difference approximation of order 2 for the first derivative. Assuming a (not necessarily uniform) mesh in one spatial direction p, e.g., with mesh sizes h_j, $j = 1\ldots n$, one has to use general finite-difference approximations. Let $\tilde{h}_j := (h_j + h_{j+1})/2,\ j = 1\ldots n-1$ be the distance between successive midpoints. For interior points the corresponding second order centered finite-difference approximations are given by

$$\begin{aligned} &\alpha_0^{k,l} f(x-\tilde{h}_k) + \alpha_1^{k,l} f(x) + \alpha_2^{k,l} f(x+\tilde{h}_l) = f'(x) + O(\tilde{h}_k\,\tilde{h}_l), \quad \text{with} \\ &\alpha_0^{k,l} = -\frac{\tilde{h}_l}{\tilde{h}_k(\tilde{h}_k+\tilde{h}_l)}, \quad \alpha_1^{k,l} = \frac{\tilde{h}_l-\tilde{h}_k}{\tilde{h}_k\,\tilde{h}_l}, \quad \alpha_2^{k,l} = \frac{\tilde{h}_k}{\tilde{h}_l(\tilde{h}_k+\tilde{h}_l)}, \end{aligned} \tag{5.24}$$

where $\tilde{h}_k, \tilde{h}_l$ are the distances to the left and right neighbor of a midpoint x, respectively. For the boundaries the analogous one-sided scheme is used:

$$\begin{aligned} &\beta_0^{k,l} f(x) + \beta_1^{k,l} f(x+\tilde{h}_k) + \beta_2^{k,l} f(x+\tilde{h}_k+\tilde{h}_l) = f'(x) + O(\tilde{h}_k(\tilde{h}_k+\tilde{h}_l)), \quad \text{with} \\ &\beta_0^{k,l} = -\frac{2\,\tilde{h}_k+\tilde{h}_l}{\tilde{h}_k(\tilde{h}_k+\tilde{h}_l)}, \quad \beta_1^{k,l} = \frac{\tilde{h}_k+\tilde{h}_l}{\tilde{h}_k\,\tilde{h}_l}, \quad \beta_2^{k,l} = -\frac{\tilde{h}_k}{\tilde{h}_l(\tilde{h}_k+\tilde{h}_l)}. \end{aligned} \tag{5.25}$$

The resulting finite-difference matrix with respect to the p - th spatial direction is then given by

$$J_n^p := \begin{pmatrix} \beta_0^{1,2} & \beta_1^{1,2} & \beta_2^{1,2} & & & \\ \alpha_0^{1,2} & \alpha_1^{1,2} & \alpha_2^{1,2} & & & \\ & \alpha_0^{2,3} & \alpha_1^{2,3} & \alpha_2^{2,3} & & \\ & \ddots & \ddots & \ddots & & \\ & & \alpha_0^{n-3,n-2} & \alpha_1^{n-3,n-2} & \alpha_2^{n-3,n-2} & \\ & & & \alpha_0^{n-2,n-1} & \alpha_1^{n-2,n-1} & \alpha_2^{n-2,n-1} \\ & & & -\beta_2^{n-1,n-2} & -\beta_1^{n-1,n-2} & -\beta_0^{n-1,n-2} \end{pmatrix} \in \mathbb{R}^{n\times n}. \tag{5.26}$$

The tensor Φ representing the scalar potential on the center points of the field cells, is given by the entries $\phi_{i_1\,i_2\,i_3} \approx \phi(\boldsymbol{x}^c_{\mathbf{i}})$, and the stray field can now be computed by evaluating the 3-component tensor

$$\mathcal{H}_d = (\mathcal{H}_d^{(1)}, \mathcal{H}_d^{(2)}, \mathcal{H}_d^{(3)})^T = (-\Phi \times_1 J^1_{n_1}, -\Phi \times_2 J^2_{n_2}, -\Phi \times_3 J^3_{n_3})^T. \tag{5.27}$$

Furthermore, the demagnetizing energy can be approximated (second order) using midpoint integration

$$e_{\mathrm{d}} = -\frac{1}{2}\sum_{p=1}^{3} \langle \mathcal{V} \cdot \mathcal{M}^{(p)}, \mathcal{H}_d^{(p)} \rangle. \tag{5.28}$$

where the rank-1 tensor $\mathcal{V} = [\![h_1, h_2, h_3]\!] \in C_{\boldsymbol{n},1}$ contains the volumes of the computational cells. In the tensor structured case the element-wise multiplication $\mathcal{V} \cdot \mathcal{M}^{(p)}$ can be done efficiently without increasing the ranks, see section 4. Moreover, the inner products in Eqn. (5.28),

Table 5.2: Comparison of accuracy of Eqs. (5.29) and Alg. 10 ($R = 50$). Absolute ($Err_{10/50}$) and relative ($Relerr_{10/50}$) l^2-errors for a $N = 10^3$ (exact errors) and $N = 50^3$ (errors for 200 randomly chosen mesh-points) tensor product grid, g^{ten} indicates the order of Gaussian quadrature used for the integrals in (5.20), the integrals in (5.29) are evaluated exactly.

g^{ten}	Err_{10}	$Relerr_{10}$	Err_{50}	$Relerr_{50}$
4	$2.31\,e{-}04$	$4.52\,e{-}04$	$1.59\,e{-}05$	$4.45\,e{-}04$
8	$1.13\,e{-}07$	$2.21\,e{-}07$	$7.77\,e{-}09$	$2.17\,e{-}07$
16	$4.34\,e{-}14$	$8.55\,e{-}14$	$5.85\,e{-}14$	$1.64\,e{-}12$
32	$1.29\,e{-}14$	$2.52\,e{-}14$	$5.86\,e{-}14$	$1.64\,e{-}12$

i.e. $\langle \mathcal{V} \cdot \mathcal{M}^{(p)}, \mathcal{H}_d^{(p)} \rangle$, can be computed efficiently for structured tensors [49].

5.2 Accuracy Tests

First Alg. 10 is compared to direct integration, i.e.

$$\phi(\boldsymbol{x}) = -\frac{1}{4\pi} \sum_{p=1}^{3} \sum_{\mathbf{j}} m_{\mathbf{j}}^{(p)} \int_{\Omega_{\mathbf{j}}} \frac{x^{(p)} - y^{(p)}}{\|\boldsymbol{x} - \boldsymbol{y}\|^3} \, \mathrm{d}\boldsymbol{y}. \tag{5.29}$$

The absolute and relative l^2-errors are used as measurements for the accuracy, i.e.

$$Err_n = \sqrt{\sum_{\mathbf{j}} (\phi_{\mathbf{j}}^{\text{dir}} - \phi_{\mathbf{j}}^{\text{ten}})^2} \equiv \left\| \Phi^{\text{dir}} - \Phi^{\text{ten}} \right\|_{\mathrm{F}}, \tag{5.30}$$

$$Relerr_n = \left\| \Phi^{\text{dir}} - \Phi^{\text{ten}} \right\|_{\mathrm{F}} / \left\| \Phi^{\text{dir}} \right\|_{\mathrm{F}}, \tag{5.31}$$

where Φ^{dir} and Φ^{ten} are the tensors representing the potential at the center points of the computational cells obtained by the direct formula (5.29) and the tensor approach in Alg. 10, respectively.

Tab. 5.2 shows the relative and absolute errors for a $N = 10^3$ and $N = 50^3$ grid, where the integrals in (5.29) were evaluated exactly by using the formulas from [13].

Now, the proposed tensor scheme of this section is compared to the finite element simulation package FEMME [71] in the case of uniform magnetization where no discretization error for the magnetization arises. The used FEM/BEM implemention solves the weak formulation of the magnetostatic Poisson equation. Dense boundary element matrices are approximated in the *H-matrix format*, see [72].

From Tab. 5.3 one can see that the FEM/BEM algorithm approximates, in the case of uniform

Table 5.3: Absolute error in the energy between results for direct tensor integration algorithm and FEM/BEM for uniform magnetization distribution in the unit cube. N indicates the number of total nodes in the mesh. In the fourth column the rel. deviation of the energy values computed by the two numerical schemes is given; percentage is based on the true value, i.e. $e_d = 1/6\,[\mu_0^{-1} M_s^{-2}]$.

N	error (tensor)	error (FEM/BEM)	deviation [%]
15^3	$1.38\,e{-}04$	$1.55\,e{-}03$	$8.50\,e{-}1$
30^3	$8.19\,e{-}05$	$4.51\,e{-}04$	$3.20\,e{-}1$
60^3	$3.98\,e{-}05$	$3.47\,e{-}04$	$2.32\,e{-}1$

Table 5.4: Absolute and relative deviation in the energy between results for direct tensor integration algorithm and finite element reference-value for magnetization distribution of a vortex in the unit cube given by Eqs. (5.32). In order to resolve the vortex, a non-uniform grid (geometrically refined towards the center of the cube) is used. The columns to the right show the minimal grid size, i.e. $\min h_j$, in the center of the cube and respectively the maximal value, i.e. $\max h_j$, next to the boundaries.

n	abs. deviation	rel. deviation [%]	grid-min	grid-max
10	$2.02\,e{-}04$	$9.32\,e{-}01$	$8.9\,e{-}02$	$1.1\,e{-}01$
20	$1.61\,e{-}04$	$7.42\,e{-}01$	$3.3\,e{-}02$	$7.2\,e{-}02$
30	$5.58\,e{-}05$	$2.58\,e{-}01$	$1.3\,e{-}02$	$6.6\,e{-}02$
40	$6.65\,e{-}06$	$3.07\,e{-}02$	$5.5\,e{-}03$	$6.6\,e{-}02$
50	$1.78\,e{-}06$	$8.23\,e{-}03$	$2.8\,e{-}03$	$6.4\,e{-}02$

magnetization configuration, about one order of magnitude worse than the direct tensor integration algorithm.

Now, a vortex-like state in a $200\,\mathrm{nm}^3$ - cube is used, which is described by the model in [73], i.e.

$$\begin{aligned} M^x(r) &= -\frac{y}{r}\,(1-\exp(-4\tfrac{r^2}{r_c^2}))^{\frac{1}{2}}, \\ M^y(r) &= \frac{x}{r}\,(1-\exp(-4\tfrac{r^2}{r_c^2}))^{\frac{1}{2}}, \\ M^z(r) &= \exp(-2\tfrac{r^2}{r_c^2}), \end{aligned} \tag{5.32}$$

where $r = \sqrt{x^2+y^2}$. The radius of the vortex core is chosen to be $r_c = 28\,\mathrm{nm}$. The vortex center coincides with the center of the cube, and the magnetization is assumed to be rotationally symmetric about the $x = y = 100\,\mathrm{nm}$ axis and translationally invariant along the z - axis.

Table 5.5: Absolute and relative deviation between results for direct tensor integration algorithm and finite element reference-value for magnetization distribution in the unit cube given by Eqs. (5.33).

n	abs. deviation	rel. deviation [%]
20	$3.42\,e{-}04$	$2.24\,e{-}01$
30	$2.83\,e{-}04$	$1.85\,e{-}01$
40	$2.43\,e{-}04$	$1.59\,e{-}01$
50	$2.18\,e{-}04$	$1.43\,e{-}01$
80	$1.83\,e{-}04$	$1.20\,e{-}01$

For the above configuration (5.32) and an amount of 50^3 nodes and about $5 \cdot 50^3$ tetrahedral elements, FEMME finds $e_d/\mu_0 = 2.16185\,e{-}02$, which is taken as reference value. This value is compared with computations using the direct tensor integration algorithm of Alg. 10 on an adaptive mesh refined geometrically, in each spatial direction, towards the vortex center of the cube, see Tab. 5.4.

Finally, the algorithm is compared with FEMME for a flower-like state of the cube in the previous example. The main magnetization direction is taken to be along the z - axis, and the flower is obtained through an in-plane perturbation along the y - axis and an out-of-plane perturbation along the x - axis. Assuming polynomial expressions for the perturbations, as in [74], the flower is the normalized version of

$$\begin{aligned} M^x(r) &= \tfrac{1}{a}(x - x_m)(z - z_m), \\ M^y(r) &= \tfrac{1}{c}(y - y_m)(z - z_m) + \tfrac{1}{b^3}(y - y_m)^3(z - z_m)^3, \\ M^z(r) &= 1, \end{aligned} \tag{5.33}$$

where x_m, y_m and z_m are the coordinates of the center of the cube. For the experiment the parameters in Eqn.(5.33) are set $a = c = 1$ and $b = 2$. Using the same finite element mesh as in the previous example, FEMME now finds $e_d/\mu_0 = 1.52653\,e{-}01$. Tab. 5.5 shows absolute and relative deviations, in this case on a uniform grid used for the direct tensor integration algorithm. One can observe a similar difference like in Tab. 5.3.
More accuracy tests were performed in [18, 25].

5.3 Mathematical Analysis of the Method

A detailed mathematical analysis of the method of the previous sections will lead to a so-called *Kronecker product approximation* [cf. (4.12)] of the 'potential operator' (5.3).
The underlying collocation scheme is proved to be quadratically convergent, see Lemma 5. Furthermore, exponential convergence in the separation rank of the approximation is shown, see Corr. 8.

5.3.1 Notation

Let $\Omega = \bigtimes_{p=1}^{3} \Omega^{(p)} \subset \mathbb{R}^3$ with $\Omega^{(p)} = [\alpha_p, \beta_p] \subset \mathbb{R}$ and assume for $p = 1 \dots 3$ a partition of $\Omega^{(p)}$ into n_p sub-intervals $I_i^{(p)}$, $i = 1 \dots n_p$. On the resulting tensor grid $T := W^{(1)} \times W^{(2)} \times W^{(3)}$ of Ω where $W^{(p)} = \bigtimes_{i=1}^{n_p} I_i^{(p)}$ sets of collocation points $\{\xi_i^{(p)} \in I_i^{(p)} : i = 1 \dots n_p\}$, $p = 1 \dots 3$ are defined (for ease of presentation, one collocation point per sub-interval, e.g. midpoints).
Further, the number of collocation points $\boldsymbol{\xi}_{\boldsymbol{i}} = (\xi_{i_1}^{(1)}, \xi_{i_2}^{(2)}, \xi_{i_3}^{(3)})$, $\boldsymbol{i} = (i_1, i_2, i_3)$ is denoted with $N := \prod_p n_p$.

5.3.2 The Collocation Scheme

Using the tensor product basis functions (e.g. $\psi_i^{(p)} := \chi_{I_i^{(p)}}$ indicator function of sub-interval $I_i^{(p)}$)

$$\psi_{i_1 i_2 i_3}(\boldsymbol{x}) = \prod_{p=1}^{3} \psi_{i_p}^{(p)}(x^{(p)}), \tag{5.34}$$

and the ansatz cf. [25] [$m_{\boldsymbol{j}}^{(q)} = m^{(q)}(\boldsymbol{\xi}_{\boldsymbol{j}})$, $\boldsymbol{j} = (j_1, j_2, j_3)$]

$$\boldsymbol{m}^{(q)}(\boldsymbol{x}) = \sum_{\boldsymbol{j}} m_{\boldsymbol{j}}^{(q)} \psi_{\boldsymbol{j}}(\boldsymbol{x}) \tag{5.35}$$

the collocation scheme for (5.3) takes the form [$\boldsymbol{i} = (i_1, i_2, i_3)$, $\boldsymbol{j} = (j_1, j_2, j_3)$]

$$\phi_{\boldsymbol{i}} = -\frac{1}{4\pi} \sum_{q=1}^{3} \sum_{\boldsymbol{j}} m_{\boldsymbol{j}}^{(q)} \int_{\Omega} g^{(q)}(\boldsymbol{\xi}_{\boldsymbol{i}}, \boldsymbol{y})\, \psi_{\boldsymbol{j}}(\boldsymbol{y})\, \mathrm{d}\boldsymbol{y}, \tag{5.36}$$

where

$$g^{(q)}(\boldsymbol{x}, \boldsymbol{y}) := \frac{x^{(q)} - y^{(q)}}{\|\boldsymbol{x} - \boldsymbol{y}\|^3}. \tag{5.37}$$

Lemma 5. *Let* $\boldsymbol{m} \in C^2(\Omega)$*. Then the collocation scheme* (5.36)*, where* $\xi_i^{(p)}$ *are the midpoints of* $I_i^{(p)}$ $i = 1 \dots n_p$*, converges quadratically.*

Proof. *Assume (w.l.o.g) uniform spacings in each dimension, i.e.* $h_p := 1/n_p$*, and use the notation* $h := \max_{p=1\dots3} h_p$*. Furthermore, let* $\Omega_{\boldsymbol{j}} := \bigtimes_{p=1}^{3} I_{j_p}^{(p)}$*.*
The local error $e(\boldsymbol{\xi_i}) = |\phi_{\boldsymbol{i}} - \phi(\boldsymbol{\xi_i})|$ *(cf.* (5.36) *and* (5.3)*) will be estimated for each fixed* $\boldsymbol{i}$ *and* q *in* (5.36) *separately, i.e. define*

$$\begin{aligned} e_q(\boldsymbol{\xi_i}) &:= \left| \sum_{\boldsymbol{j}} m_{\boldsymbol{j}}^{(q)} \int_\Omega g^{(q)}(\boldsymbol{\xi_i}, \boldsymbol{y})\, \psi_{\boldsymbol{j}}(\boldsymbol{y})\, d\boldsymbol{y} - \sum_{\boldsymbol{j}} \int_\Omega m^{(q)}(\boldsymbol{y})\, g^{(q)}(\boldsymbol{\xi_i}, \boldsymbol{y})\, \psi_{\boldsymbol{j}}(\boldsymbol{y})\, d\boldsymbol{y} \right| \\ &= \left| \sum_{\boldsymbol{j}} \int_\Omega (m_{\boldsymbol{j}}^{(q)} - m^{(q)}(\boldsymbol{y}))\, g^{(q)}(\boldsymbol{\xi_i}, \boldsymbol{y})\, \psi_{\boldsymbol{j}}(\boldsymbol{y})\, d\boldsymbol{y} \right|. \end{aligned} \tag{5.38}$$

By using Taylor expansion for $m^{(q)}$ *at 'source' points* $\boldsymbol{\xi_j}$*, i.e.*

$$m^{(q)}(\boldsymbol{y}) = m^{(q)}(\boldsymbol{\xi_j}) + \left\langle \nabla m^{(q)}(\boldsymbol{\xi_j}), \boldsymbol{y} - \boldsymbol{\xi_j} \right\rangle + O(\left\|\boldsymbol{y} - \boldsymbol{\xi_j}\right\|^2), \tag{5.39}$$

one obtains ($C_1 > 0$ *independent of* h*)*

$$e_q(\boldsymbol{\xi_i}) \leq \sum_{\boldsymbol{j}} \left| \int_\Omega \left\langle \nabla m^{(q)}(\boldsymbol{\xi_j}), \boldsymbol{y} - \boldsymbol{\xi_j} \right\rangle g^{(q)}(\boldsymbol{\xi_i}, \boldsymbol{y})\, \psi_{\boldsymbol{j}}(\boldsymbol{y})\, d\boldsymbol{y} \right| + C_1 \left| \int_\Omega \left\|\boldsymbol{y} - \boldsymbol{\xi_j}\right\|^2 g^{(q)}(\boldsymbol{\xi_i}, \boldsymbol{y})\, \psi_{\boldsymbol{j}}(\boldsymbol{y})\, d\boldsymbol{y} \right|. \tag{5.40}$$

It can easily be seen that $g^{(q)}(\boldsymbol{\xi_i}, .) \in L^p(\Omega_{\boldsymbol{j}})$ *for all* $\boldsymbol{j}$ *and* $1 \leq p < 3/2$*.*
Hence, the second term in (5.40) *allows the estimate*

$$\begin{aligned} \sum_{\boldsymbol{j}} \left| \int_\Omega \left\|\boldsymbol{y} - \boldsymbol{\xi_j}\right\|^2 g^{(q)}(\boldsymbol{\xi_i}, \boldsymbol{y})\, \psi_{\boldsymbol{j}}(\boldsymbol{y})\, d\boldsymbol{y} \right| &\leq \sum_{\boldsymbol{j}} \left\|g^{(q)}(\boldsymbol{\xi_i}, .)\right\|_{L^1(\Omega_{\boldsymbol{j}})} \int_\Omega \left\|\boldsymbol{y} - \boldsymbol{\xi_j}\right\|^2 \psi_{\boldsymbol{j}}(\boldsymbol{y})\, d\boldsymbol{y} \\ \leq \left\|g^{(q)}(\boldsymbol{\xi_i}, .)\right\|_{L^1(\Omega)} \sum_{\boldsymbol{j}} \int_{\Omega_{\boldsymbol{j}}} \left\|\boldsymbol{y} - \boldsymbol{\xi_j}\right\|^2 d\boldsymbol{y} &\leq \left\|g^{(q)}(\boldsymbol{\xi_i}, .)\right\|_{L^1(\Omega)} |\Omega|\, h^2 = O(h^2). \end{aligned} \tag{5.41}$$

For the first term one has to distinguish between the diagonal ($\boldsymbol{i} = \boldsymbol{j}$*) and non-diagonal (*$\boldsymbol{i} \neq \boldsymbol{j}$*) case. The kernel* $g^{(q)}(\boldsymbol{\xi_i}, .)$ *is analytic in the case* $\boldsymbol{i} \neq \boldsymbol{j}$ *and thus allows Taylor expansion, i.e.*

$$g^{(q)}(\boldsymbol{\xi_i}, \boldsymbol{y}) = g^{(q)}(\boldsymbol{\xi_i}, \boldsymbol{\xi_j}) + O(\left\|\boldsymbol{y} - \boldsymbol{\xi_j}\right\|). \tag{5.42}$$

The constant term in the expansion does not contribute to the first term in (5.40) *since* $\boldsymbol{y} - \boldsymbol{\xi_j}$ *is* *odd w.r.t.* $\boldsymbol{\xi}_j$ *in* Ω_j*. Thus, one gets (*$C_2 > 0$ *independent of* h*)*

$$\begin{aligned}\sum_{j\neq i}\left|\int_\Omega \left\langle \nabla m^{(q)}(\boldsymbol{\xi}_j), \boldsymbol{y}-\boldsymbol{\xi}_j\right\rangle g^{(q)}(\boldsymbol{\xi}_i,\boldsymbol{y})\,\psi_j(\boldsymbol{y})\,d\boldsymbol{y}\right| &\le \left\|\nabla m^{(q)}(\boldsymbol{\xi}_j)\right\| \sum_{j\neq i}\int_{\Omega_j} O(\left\|\boldsymbol{y}-\boldsymbol{\xi}_j\right\|^2)\\ &\le C_2\,|\Omega|\,h^2 = O(h^2).\end{aligned} \tag{5.43}$$

In the case $\boldsymbol{i} = \boldsymbol{j}$ *there holds for the first term in* (5.40)

$$\begin{aligned}\left|\int_\Omega \left\langle \nabla m^{(q)}(\boldsymbol{\xi}_i), \boldsymbol{y}-\boldsymbol{\xi}_i\right\rangle g^{(q)}(\boldsymbol{\xi}_i,\boldsymbol{y})\,\psi_i(\boldsymbol{y})\,d\boldsymbol{y}\right| &\le \left\|\nabla m^{(q)}(\boldsymbol{\xi}_i)\right\| \int_\Omega \left\|\boldsymbol{y}-\boldsymbol{\xi}_i\right\| \left|g^{(q)}(\boldsymbol{\xi}_i,\boldsymbol{y})\right| \psi_i(\boldsymbol{y})\,d\boldsymbol{y}\\ &\le C_2\,(I_1 f)(\boldsymbol{\xi}_i),\end{aligned} \tag{5.44}$$

where $(I_1 f)(\boldsymbol{\xi}_i) := \int_{\mathbb{R}^3} \frac{|f(\boldsymbol{y})|}{\|\boldsymbol{\xi}_i-\boldsymbol{y}\|^2}\,d\boldsymbol{y}$ *with* $f(\boldsymbol{y}) = (\xi_i^{(q)} - y^{(q)})\psi_i(\boldsymbol{y}) \in L^p(\Omega)$, $p \ge 1$ *with compact support* Ω_i*. Since* $1/\left\|\boldsymbol{x}\right\|^2 \in L^p(\Omega)$, $1 \le p < 3/2$*, one proceeds with*

$$(I_1 f)(\boldsymbol{\xi}_i) \le \int_{\Omega_i} \frac{h_q}{\left\|\boldsymbol{\xi}_i - \boldsymbol{y}\right\|^2}\,d\boldsymbol{y} \le h \int_{B_h(\boldsymbol{0})} \frac{1}{\left\|\boldsymbol{x}\right\|^2}\,d\boldsymbol{x} = 4\pi h^2, \tag{5.45}$$

which completes the proof. □

Fig. 5.4 shows the quadratic convergence of the error of the collocation scheme (5.36) compared to (5.3) evaluated at the origin[2] for the radial symmetric function $m(\boldsymbol{x}) = \exp(-\left\|\boldsymbol{x}\right\|^2)$.

The evaluation of the potential ϕ on T by abbreviating notation for the matrices $\boldsymbol{G}^{(q)} \in \mathbb{R}^{N\times N}$

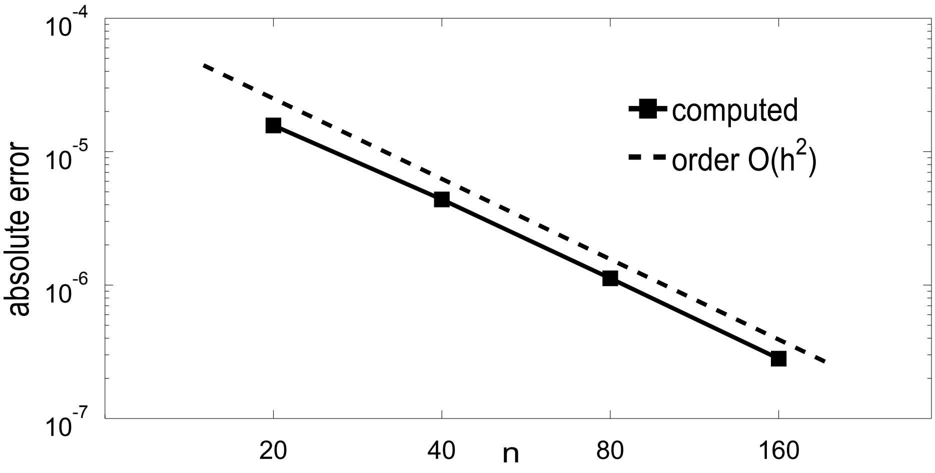

Figure 5.4: Absolute error of collocation scheme (5.36). *Reprinted from [26].*

---

[2] Maple 14 was used for evaluating the expression (5.3) at the points $\boldsymbol{\xi}_i = 1/2(h,h,h)$ with $h = 1/n$.

with entries

$$G_{ij}^{(q)} = \int_{\Omega} g^{(q)}(\boldsymbol{\xi}_i, \boldsymbol{y})\, \psi_j(\boldsymbol{y})\, \mathrm{d}\boldsymbol{y} \tag{5.46}$$

and the grid-sampled magnetization components $\boldsymbol{m}^{(q)} \in \mathbb{R}^N$ is given as

$$\phi \approx -\frac{1}{4\pi} \sum_{q=1}^{3} \boldsymbol{G}^{(q)}\, \boldsymbol{m}^{(q)}, \tag{5.47}$$

yielding computational effort $O(N^2)$ if no special structure on $\boldsymbol{G}^{(q)}$ can be imposed.
The term *(potential) operator* will be used for

$$P : \mathbb{R}^N \times \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}^N,\ (\boldsymbol{m}^{(1)}, \boldsymbol{m}^{(2)}, \boldsymbol{m}^{(3)}) \mapsto -\frac{1}{4\pi} \sum_{q=1}^{3} \boldsymbol{G}^{(q)}\, \boldsymbol{m}^{(q)}. \tag{5.48}$$

In the following *Gauss-Transform* (cf. [25]) and *sinc-quadrature* [75] will be used to construct a Kronecker product structure for (5.48) with an asymptotically optimal approximation error yielding a 'tensor' version $\mathcal{P}$ of (5.48) that can operate on structured tensors.

5.3.3 Sinc Quadrature

Here, some basic facts from the theory of sinc function based approximation are stated, see [75],[57].
The sinc function $\operatorname{sinc}(x) := \frac{\sin(\pi x)}{\pi x}$ is an analytic function which is 1 at $x = 0$ and zero at $x \in \mathbb{Z} \setminus \{0\}$. Sufficiently fast decaying continuous functions $f \in C(\mathbb{R})$ can be interpolated at the grid points $x = k\vartheta \in \vartheta\mathbb{Z}$, $\vartheta > 0$ (step size) by functions $S_{k,\vartheta}(x) := \operatorname{sinc}(x/\vartheta - k)$, i.e.

$$f_\vartheta(x) = \sum_{k\in\mathbb{Z}} f(k\vartheta) S_{k,\vartheta}(x). \tag{5.49}$$

Since $\int_{\mathbb{R}} \operatorname{sinc}(t)\, \mathrm{d}t = 1$, the interpolatory quadrature for $\int_{\mathbb{R}} f(t)\, \mathrm{d}t$ leads to

$$\int_{\mathbb{R}} f(t)\, \mathrm{d}t \approx \vartheta \sum_{k\in\mathbb{Z}} f(k\vartheta), \tag{5.50}$$

which can be seen as infinite trapezoidal rule. Truncation to $k = -R \ldots R$ of the infinite sum in (5.50) leads to the *sinc quadrature rule* with $2R+1$ terms with the truncation error $\vartheta \sum_{|k|>R} f(k\vartheta)$ that obviously depends on the decay-rate of f on the real axis.
For functions $f \in H^1(D_\delta)$, $\delta < \pi/2$ (*Hardy space*), i.e. which are holomorphic in the strip

$D_\delta := \{z \in \mathbb{C} : |\Im z| \le \delta\}$ with

$$N(f, D_\delta) := \int_{\partial D_\delta} |f(z)|\,|dz| = \int_{\mathbb{R}} (|f(t+i\delta)| + |f(t-i\delta)|)\,\mathrm{d}\,t < \infty, \tag{5.51}$$

and in addition to $f \in H^1(D_\delta)$ have double exponential decay on the real axis, the following exponential error estimate for the sinc quadrature holds (cf. [57], Proposition 2.1), which is stated here for the sake of completeness.

Theorem 6 ([57])**.** *Let $f \in H^1(D_\delta)$ with some $\delta < \pi/2$. If f satisfies the condition*

$$|f(t)| \le C\,\exp(-b e^{a|t|}) \quad \forall t \in \mathbb{R} \text{ with } a, b, C > 0, \tag{5.52}$$

then the quadrature error for the special choice $\vartheta = \log(\frac{2\pi a R}{b})/(aR)$ satisfies

$$\left| \int_{\mathbb{R}} f(t)\,dt - \vartheta \sum_{|k|\le R} f(k\vartheta) \right| \le C\,N(f, D_\delta)\,\exp\Big(\frac{-2\pi\delta a R}{\log(2\pi a R/b)}\Big). \tag{5.53}$$

Remark 3. *In the case $f(\rho)$ as in (5.1) the constants in (5.53) depend on ρ. For some fixed ρ, an accuracy of $\epsilon > 0$ can be achieved with $R = \mathcal{O}(|\log\epsilon| \cdot \log|\log\epsilon|)$.*

Remark 4. *Instead of using sinc quadrature like in [25], one could use the best approximation of $\|\boldsymbol{x} - \boldsymbol{y}\|^{-3}$ in a specified interval. Here, better error estimates are obtained [76]. Note that this approach is used in the recent work [15], see section 8.*

5.3.4 Separable Approximation

The Gaussian transform

$$\frac{1}{\rho^{3/2}} = \frac{2}{\sqrt{\pi}} \int_{\mathbb{R}} \tau^2\, e^{-\tau^2 \rho}\,\mathrm{d}\tau, \tag{5.54}$$

is used to obtain for $\rho = \left\|\boldsymbol{\xi}_i - \boldsymbol{y}\right\|^2$ and $q = 1 \ldots 3$ the new representation for (5.46)

$$\boldsymbol{G}^{(q)}_{\boldsymbol{ij}} = \frac{2}{\sqrt{\pi}} \int_{\mathbb{R}} \tau^2 \prod_{p=1}^{3} h^{(p)}_{i_p j_p}(\tau)\,\mathrm{d}\tau \equiv \frac{2}{\sqrt{\pi}} \int_{\mathbb{R}} f(\tau)\,\mathrm{d}\tau, \tag{5.55}$$

with

$$h^{(p)}_{i_p j_p}(\tau) = \begin{cases} \int_{I^{(p)}_{j_p}} e^{-\tau^2(\xi_{i_p} - y)^2}\,\mathrm{d}y & p \ne q, \\ \int_{I^{(p)}_{j_p}} (\xi_{i_p} - y)\, e^{-\tau^2(\xi_{i_p} - y)^2}\,\mathrm{d}y & p = q. \end{cases} \tag{5.56}$$

Lemma 7. *After applying the substitution $\tau = \sinh(t)$ the transformed integral (5.55) with integrand $\widetilde{f}(t) := \cosh(t)\, f(\sinh(t))$ allows an exponentially convergent sinc quadrature, cf. Theorem 6, where the constants in (5.53) depend on the parameters in (5.56).*

Proof. *Assume w.l.o.g. $q = 1$ and (5.56) to be transformed to integrals over intervals $[a_p, b_p]$, i.e. $a_p := \xi_{i_p} - j_p h_p$ and $b_p := \xi_{i_p} - (j_p - 1)h_p$, where h_p is the length of the interval $I^{(p)}_{j_p}$, which w.l.o.g. is assumed to be constant on the p-th axis. Set $C := \prod_{j=1}^{3}[a_j, b_j] \subset \prod_{j=1}^{3}[h_j/2, \beta_j - \alpha_j]$ (assume midpoints as collocation points); then the transformed integrand in (5.55) reads*

$$\widetilde{f}(t) = \int_C x^{(1)} \cosh(t) \sinh^2(t) \exp\big(-\sinh^2(t) \sum_{j=1}^{3} {x^{(j)}}^2\big)\, d(x^{(1)}, x^{(2)}, x^{(3)}). \tag{5.57}$$

One obtains analytically up to constants

$$\begin{aligned}\widetilde{f}(t) \propto \frac{\cosh(t)}{\sinh^2(t)}\Big(e^{-a_1^2 \sinh^2(t)} - e^{-b_1^2 \sinh^2(t)}\Big)\Big(\mathit{erf}(b_2 \sinh(t)) - \mathit{erf}(a_2 \sinh(t))\Big)\times \\ \Big(\mathit{erf}(b_3 \sinh(t)) - \mathit{erf}(a_3 \sinh(t))\Big),\end{aligned} \tag{5.58}$$

where the so-called error function is defined as

$$\mathit{erf}(x) := \frac{2}{\sqrt{\pi}} \int_0^x e^{-\tau^2}\, d\tau. \tag{5.59}$$

The functions $\mathit{erf}(\sinh(z))/\sinh(z)$, $\exp(-\sinh^2(z))$ and $\cosh(z)$ are all entire functions, hence, $\widetilde{f}$ is holomorphic over $\mathbb{C}$.
Moreover, there holds ($a > 0$)

$$\exp\big(-a^2 \sinh^2(t)\big) = \mathcal{O}\big(\exp(-\tfrac{a^2}{4} e^{2|t|})\big) \text{ as } |t| \to \infty \tag{5.60}$$

and using asymptotic expansion for the error function [3] gives ($a, b > 0$, $a \neq b$)

$$\mathit{erf}(b \sinh(t)) - \mathit{erf}(a \sinh(t)) = \mathcal{O}\Big(\frac{\exp\big(-a^2 \sinh^2(t)\big) - \exp\big(-b^2 \sinh^2(t)\big)}{\sinh(t)}\Big) \text{ as } |t| \to \infty, \tag{5.61}$$

which shows the required double exponential decay for $\widetilde{f}$.
It remains to show that $N(\widetilde{f}, D_\delta) < \infty$. For $z = t \pm i\delta$, $H := \sum_{j=1}^{3} h_j^2/4$ and $C_0 = \prod_{j=1}^{3}[h_j/2, \beta_j -$

---

[3] $\mathrm{erf}(x) \approx 1 - \frac{\exp(-x^2)}{\sqrt{\pi}x}\big(1 - \frac{1}{2x^2} + \frac{3}{(2x^2)^2} - \frac{15}{(2x^2)^3} \pm \ldots\big)$, $x \gg 1$

$\alpha_j]$ *one gets*

$$\int_{\mathbb{R}}\left|\int_C x^{(1)}\,\cosh(t\pm i\delta)\,\sinh^2(t\pm i\delta)\,\exp\big(-\sinh^2(t\pm i\delta)\sum_{j=1}^{3}x^{(j)2}\big)\,d(x^{(1)},x^{(2)},x^{(3)})\right|\,dt\ \le$$
$$|C_0|\int_{\mathbb{R}}|\cosh(t\pm i\delta)|\,\left|\sinh^2(t\pm i\delta)\right|\,\left|\exp\big(-\sinh^2(t\pm i\delta)H\big)\right|\,dt<\infty, \tag{5.62}$$

since the remaining integrand is a smooth function in t with (double) exponential decay as $|t|\to\infty$.
This completes the proof. □

Remark 5. *Since in an estimate for $N(f,D_\delta)$ a term like*

$$|\exp\big(-a^2\sinh^2(t\pm i\delta)\big)|=\exp\Big(-\tfrac{a^2}{2}\big(\cos(2\delta)\cosh(2t)-1\big)\Big),$$

can have positive exponent (e.g. t close to zero), the norm cannot be estimated uniformly in h_p, prohibiting an exponentially convergent sinc quadrature for $h_p\to 0$.
Nevertheless, for the grid assumed to be fixed, Lemma 7 gives a Kronecker product approximation of the operator (5.48) with separation rank R, see Cor. 8, where $R=\mathcal{O}(|\log\epsilon|\cdot\log|\log\epsilon|)$ for a prescribed accuracy ϵ.

Corollary 8. *The operator (5.48) admits a Kronecker product approximation of the form (4.12) with rank R, where $R=\mathcal{O}(|\log\epsilon|\cdot\log|\log\epsilon|)$ for a prescribed accuracy ϵ.*

Proof. *The substitution $\tau=\sinh(t)$ preserves the symmetry in the integrand (5.54), leading to a $R+1$-term sinc quadrature after applying Lemma 7. Omitting the first term, which is zero, leads to the R-term representation for the potential operator $\mathcal{P}:\bigtimes_{q=1}^{3}\bigotimes_{p=1}^{3}\mathbb{R}^{n_p}\to\bigotimes_{p=1}^{3}\mathbb{R}^{n_p}$ (cf. (5.48) and (5.55)) with* [4]

$$\mathcal{P}^{(q)}=-\frac{1}{2\pi^{\frac{3}{2}}}\sum_{k=1}^{R}\alpha_k\,\boldsymbol{D}_k^{(q_3)}\otimes\boldsymbol{D}_k^{(q_2)}\otimes\boldsymbol{D}_k^{(q_1)}, \tag{5.63}$$

with

$$(\boldsymbol{D}_k^{(q_p)})_{i_pj_p}=h^{(p)}_{i_pj_p}\big(\sinh(t_k)\big)\quad\text{and}\quad\alpha_k=\cosh(t_k)\,\sinh^2(t_k), \tag{5.64}$$

where $t_k=k\vartheta$ for $\vartheta=c_0\,\log(R)/R$ and appropriate $c_0>0$ cf. Theorem 6. □

The evaluation of one component of $\mathcal{P}$ (cf. (5.63)) for a rank-1 tensor, i.e. $\mathcal{X}=v^{(3)}\otimes v^{(2)}\otimes v^{(1)}\equiv$

---

[4] One notes that with $I=i_1+(i_2-1)n_1+(i_3-1)n_2^2$ and $J=j_1+(j_2-1)n_1+(j_3-1)n_2^2$ the entries of a matrix $\boldsymbol{A}\in\mathbb{R}^{\prod_{p=1}^{3}n_p\times\prod_{p=1}^{3}n_p}$ given by $a_{IJ}=a^{(1)}_{i_1j_1}a^{(2)}_{i_2j_2}a^{(3)}_{i_3j_3}$ correspond to the (I,J)-entry of $\boldsymbol{A}^{(3)}\otimes\boldsymbol{A}^{(2)}\otimes\boldsymbol{A}^{(1)}$.

$[\![v^{(3)}, v^{(2)}, v^{(3)}]\!] \in \mathcal{C}_{\boldsymbol{n},1}$, is given as

$$\mathcal{P}^{(q)}\mathcal{X} = -\frac{1}{2\pi^{\frac{3}{2}}} \sum_{k=1}^{R} \alpha_k \, (\boldsymbol{D}_k^{(q_3)} \, v^{(3)}) \otimes (\boldsymbol{D}_k^{(q_2)} \, v^{(2)}) \otimes (\boldsymbol{D}_k^{(q_1)} \, v^{(1)}), \tag{5.65}$$

and amounts in a computational cost of $\mathcal{O}(R \sum_{p=1}^{3} n_p^2)$.

It is not far to seek a reduction of this complexity by reducing the cost for the matrix-vector product (cf. Sec. 5.4) or reduction of the separation rank R (cf. Sec. 5.3.5).

By using the relations[5] [49] (assume appropriate dimensions for the involved matrices)

$$\begin{aligned} (\boldsymbol{A}_1 \otimes \boldsymbol{A}_2)(\boldsymbol{B}_1 \odot \boldsymbol{B}_2) &= \boldsymbol{A}_1\boldsymbol{B}_1 \odot \boldsymbol{A}_2\boldsymbol{B}_2 && (5.66) \\ (\boldsymbol{A}_1 \otimes \boldsymbol{A}_2)(\boldsymbol{C}_1 \otimes \boldsymbol{C}_2) &= \boldsymbol{A}_1\boldsymbol{C}_1 \otimes \boldsymbol{A}_2\boldsymbol{C}_2, && (5.67) \end{aligned}$$

and

$$\operatorname{vec}\Big([\![\boldsymbol{\lambda};\, \boldsymbol{V}^{(1)}, \boldsymbol{V}^{(2)}, \boldsymbol{V}^{(3)}]\!]\Big) = (\boldsymbol{V}^{(3)} \odot \boldsymbol{V}^{(2)} \odot \boldsymbol{V}^{(1)})\boldsymbol{\lambda}, \tag{5.68}$$

respectively

$$\operatorname{vec}\Big([\![\mathcal{C};\, \boldsymbol{V}^{(1)}, \boldsymbol{V}^{(2)}, \boldsymbol{V}^{(3)}]\!]\Big) = (\boldsymbol{V}^{(3)} \otimes \boldsymbol{V}^{(2)} \otimes \boldsymbol{V}^{(1)})\operatorname{vec}(\mathcal{C}), \tag{5.69}$$

one gets for the evaluation of (5.63) for $\mathcal{X} \in \mathcal{C}_{\boldsymbol{n},r}$ (also compare with Alg. 10 and [25])

$$\mathcal{P}^{(q)}\mathcal{X} = -\frac{1}{2\pi^{\frac{3}{2}}} \sum_{k=1}^{R} \alpha_k \, [\![\boldsymbol{\lambda};\, \boldsymbol{D}_k^{(q_1)}\boldsymbol{V}^{(1)}, \boldsymbol{D}_k^{(q_2)}\boldsymbol{V}^{(2)}, \boldsymbol{D}_k^{(q_1)}\boldsymbol{V}^{(3)}]\!], \tag{5.70}$$

respectively for $\mathcal{X} \in \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$ the formula

$$\mathcal{P}^{(q)}\mathcal{X} = -\frac{1}{2\pi^{\frac{3}{2}}} \sum_{k=1}^{R} \alpha_k \, [\![\mathcal{C};\, \boldsymbol{D}_k^{(q_1)}\boldsymbol{V}^{(1)}, \boldsymbol{D}_k^{(q_2)}\boldsymbol{V}^{(2)}, \boldsymbol{D}_k^{(q_1)}\boldsymbol{V}^{(3)}]\!]. \tag{5.71}$$

For unstructured tensors $\mathcal{X} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$ formula (5.69) gives

$$\mathcal{P}^{(q)}\mathcal{X} = -\frac{1}{2\pi^{\frac{3}{2}}} \sum_{k=1}^{R} \alpha_k \, [\![\mathcal{X};\, \boldsymbol{D}_k^{(q_1)}, \boldsymbol{D}_k^{(q_2)}, \boldsymbol{D}_k^{(q_1)}]\!]. \tag{5.72}$$

---

[5] The symbol $\odot$ stands here for the *Khatri-Rao product*, cf. Sec. 4.3

5.3.5 Some Practical Issues

Here the question is briefly addressed how to choose the rank R.
If one applies sinc quadrature to the Gaussian transformed kernel (5.54), one can adaptively determine the rank by controlling the relative error of the quadrature in an interval $\rho \in [\rho_{\min}, \rho_{\max}]$ corresponding to the mesh size parameter h by the relation $\rho = \left\|\boldsymbol{\xi}_i - \boldsymbol{y}\right\|^2 \geq 3h^2/4$ [cf. proof of Lemma 7 Eq. (5.57)]. One can scale the computational domain to unity, hence, in Fig. 5.5 $\rho_{\max} \leq 3$ was considered, which corresponding to $\Omega_{\text{scaled}} = [0,1]^3$. One observes a uniform relative error bound for h greater some $h_{\min}$. Also compare with section 8. Moreover, one can see from Fig. 5.5 the exponential decay of the error with respect to the rank (the logarithmic error is a decreasing affine function of the rank).

The separation rank could also be reduced by applying re-compression of the CP representation (cf. Sec. 4.2) corresponding to (5.63). This can be done by compressing (5.63) as Tucker tensor by Alg. 6 and subsequent approximation of the resulting core to CP.[6] This yields a canonical tensor with smaller rank. The resulting CP representation again corresponds to a Kronecker product representation.

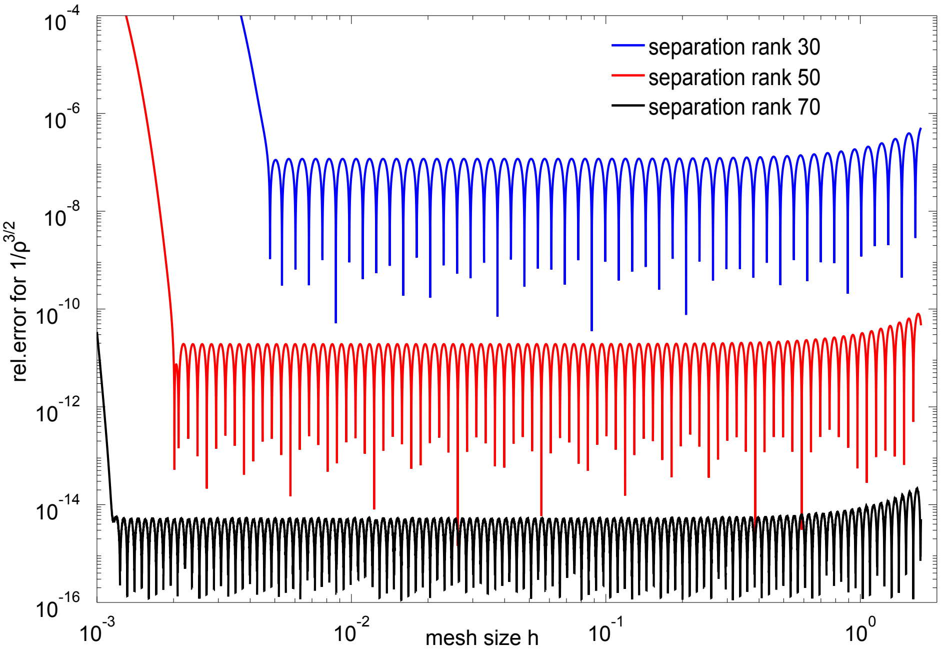

Figure 5.5: Relative error for sinc quadrature of (5.54) with substitution $\tau = \sinh(t)$. $c_0 = 2.15$. *Reprinted from [26].*

---

[6] E.g. by optimization based algorithms [51] or direct CP approximation to a smaller rank.

5.4 FFT Acceleration

Here the question is addressed how to use FFT in order to reduce computational costs for evaluating the operator $\mathcal{P}$, cf. Corr. 8 Eqn (5.63). The evaluation of this operator for rank-1 tensors [cf. (5.65)] scales quadratically in the number of collocation points in one dimension. Even though this is super-optimal, also referred to as *sub-linear* [cf. [25] or (slightly misleading) as *super-linear* (cf. [24])], one can still reduce this complexity (on uniform grids) by observing a convolution in (5.3) and using the properties of DFT for structured tensors of Lemma 3 and 4. Assume that the grid spacing, i.e. h_p, is constant for each p and the collocation points to be the midpoints of the intervals $I^{(p)}_{j_p} \equiv I^{(p)}$.
By applying the substitution used in the proof of Lemma 7 one gets for the functions in (5.56)

$$h^{(p)}_{i_p-j_p}(\tau) = h^{(p)}_{i_p j_p}(\tau) = \begin{cases} \int_{a_{i_p-j_p}}^{b_{i_p-j_p}} e^{-\tau^2 x^2}\,\mathrm{d}x & p \neq q, \\ \int_{a_{i_p-j_p}}^{b_{i_p-j_p}} x\, e^{-\tau^2 x^2}\,\mathrm{d}x & p = q, \end{cases} \tag{5.73}$$

where $a_{i_p-j_p} = (i_p - j_p)h_p - \frac{h_p}{2}$ and $b_{i_p-j_p} = (i_p - j_p)h_p + \frac{h_p}{2}$.
For $i_p, j_p = 1 \ldots n_p$ these are $2n_p-1$ different integrals. Therefore, set $i_p - j_p = J_p = 1 \ldots 2n_p - 1$ and identify the *convolution kernel* $\boldsymbol{G}^{(q)}_{\boldsymbol{i}-\boldsymbol{j}} \in \bigotimes_{p=1}^{3} \mathbb{R}^{2n_p-1}$ with entries

$$G^{(q)}_{J_1 J_2 J_3} = \frac{2}{\sqrt{\pi}} \int_{\mathbb{R}} \tau^2 \prod_{p=1}^{3} h^{(p)}_{J_p}(\tau)\,\mathrm{d}\tau. \tag{5.74}$$

Lemma 7 also holds for (5.74) [after the substitution $\tau = \sinh(t)$] and hence one can apply sinc quadrature leading to [cf. (5.48) and (5.55)] the *canonical representation*

$$\mathcal{P}^{(q)} = -\frac{1}{2\pi^{\frac{3}{2}}} [\![\, \lambda;\ \boldsymbol{D}^{(q)}_1, \boldsymbol{D}^{(q)}_2, \boldsymbol{D}^{(q)}_3 \,]\!] \in C_{2\boldsymbol{n}-1,R}, \tag{5.75}$$

with

$$(\boldsymbol{D}^{(q)}_p)_{J_p k} = h^{(p)}_{J_p}(\sinh(t_k)) \quad \text{and} \quad \lambda_k = \cosh(t_k)\,\sinh^2(t_k), \tag{5.76}$$

where $t_k = k\vartheta$ for $\vartheta = c_0 \log(R)/R$ and appropriate $c_0 > 0$ cf. Theorem 6.
By denoting the element-wise product (*Hadamard product*) for tensors with $\bullet$ and using DFT, the evaluation of one component of $\mathcal{P}$ [cf. (5.75)] for a tensor $\mathcal{X} \in \bigotimes_{p=1}^{3} \mathbb{R}^{n_p}$ with (zero-padded) Fourier transform $\widehat{\mathcal{X}} \in \bigotimes_{p=1}^{3} \mathbb{C}^{2n_p-1}$ reads

$$\mathcal{P}^{(q)} * \mathcal{X} = \mathcal{F}^{-1}\Big(-\frac{1}{2\pi^{\frac{3}{2}}} [\![\, \lambda;\ \widehat{\boldsymbol{D}}^{(q)}_1, \widehat{\boldsymbol{D}}^{(q)}_2, \widehat{\boldsymbol{D}}^{(q)}_3 \,]\!] \bullet \widehat{\mathcal{X}} \Big). \tag{5.77}$$

The complexity of FFT-based convolution in (5.77) with an unstructured tensor $\mathcal{X}$ is dominated by the costs for the FFT of $\mathcal{X}$, and amounts in a complexity of $O(R\sum_{p=1}^{3}\log n_p \prod_{q=1}^{3} n_q)$, which is comparable with usual FFT-based computation of the scalar potential (except the constant R, the order of sinc quadrature), cf. [18], [77]. However, the storage for the kernel is lower, see remark below.

For structured tensors $\mathcal{X}$ the complexity of FFT-based convolution with (5.75) scales like $n_p \log n_p$ in the mode sizes due to the Lemmas 3 and 4 of Sec.4.5, i.e. if $\mathcal{X}$ is a Tucker tensor one has complexity $O(R\sum_{p=1}^{3} r_p n_p \log n_p)$ and $O(Rr\sum_{p=1}^{3} n_p \log n_p)$ for canonical tensors, respectively. Of course, the Hadamard product in (5.77) gives additional costs, see section 4.

One can think of re-compression of (5.75), e.g. efficient compression of a CP tensor to a Tucker tensor by Alg.6 or to the *Tensor Train format* (TT) 4.4, if the magnetization itself is represented in Tucker representation. The (complex) Hadamard product in Fourier space can be performed with the techniques described in section 4, depending on the structure of the magnetization component tensors. Alternatively, one could use a black-box approximation of the result of the Hadamard product by means of adaptive cross approximation. For more details on that, the reader is referred to the corresponding remarks in section 4 and the given references. Recently, a related idea was used in the context of the Poisson equation arising in the *Stokes problem* and experimentally tested for the two dimensional case [78].

In the FFT variant described in this section, choosing a large number of sinc-quadrature terms ($\propto$ rank R) becomes a feasible option, which results in an accurate representation of the operator (5.48). Since the computation and the re-compression of the discrete kernels (5.75) are done in a setup-phase in a micromagnetic simulation, the higher amount of work due to higher ranks does not significantly influence the overall computing time.

Alg. 11 describes the FFT-based procedure for computing the scalar potential for Tucker magnetization. Fig. 5.6 shows results on complexity and accuracy using Alg. 11 for the case of uniform magnetization (means constant ranks $\equiv 1$), cf. [25]. One can observe the quasi-linear complexity ($n \log n$), while the relative error in the energy decreases with order about 1.5. The discretization for the energy and the stray field calculation from the potential were carried out second order. Fig. 5.7 shows logarithmic rank-grows for the stray field (averaged over modes) induced by a non-trivial (flower-like) magnetization state [18], while using an accuracy of 1e-12 in Alg. 11.

Remark 6. *A version of Alg. 11 for CP magnetization without representing the corresponding CP tensors as Tuckers (which is of course also possible) is straightforward, i.e. in the setup the CP kernel is not compressed in Tucker format and the Hadamard product is performed within the CP format. The result is a CP tensor.*

Remark 7. *A version of Alg. 11 for dense magnetization has the same computational complex-*

Algorithm 11 Scalar potential DFFT; `scpotDFFT`($\mathcal{M}^{(1)}, \mathcal{M}^{(2)}, \mathcal{M}^{(3)}, \boldsymbol{h}, R, c_0,$ `tol` > 0)

```
Require: $\mathcal{M}^{(p)} \in \mathcal{T}_{\mathbf{n},\mathbf{r}^{(p)}}$, $h_p > 0\,(p = 1 \ldots 3)$, $R \in \mathbb{N}$, $c_0 > 0$, tol $> 0$
Ensure: $\Phi \in \mathcal{T}_{\mathbf{n},\mathbf{r}'}$
  Setup
    • Compute the CP kernels (5.75)
    • Compress the kernels to Tucker tensors with tolerance tol
    • Compute DFFT of Tucker kernels by using Lemma 4
  Actual computation
  for $q = 1 \ldots 3$ do
    • Compute DFFT of $q$-th Tucker magnetization component by using Lemma 4
    • Compute Tucker Hadamard products of magnetization component and kernels in Fourier space using e.g. the tolerance tol
    • Compute IDFFT for result of previous step
  end for
  Perform approximate summation of results in previous step using e.g. the tolerance tol
```

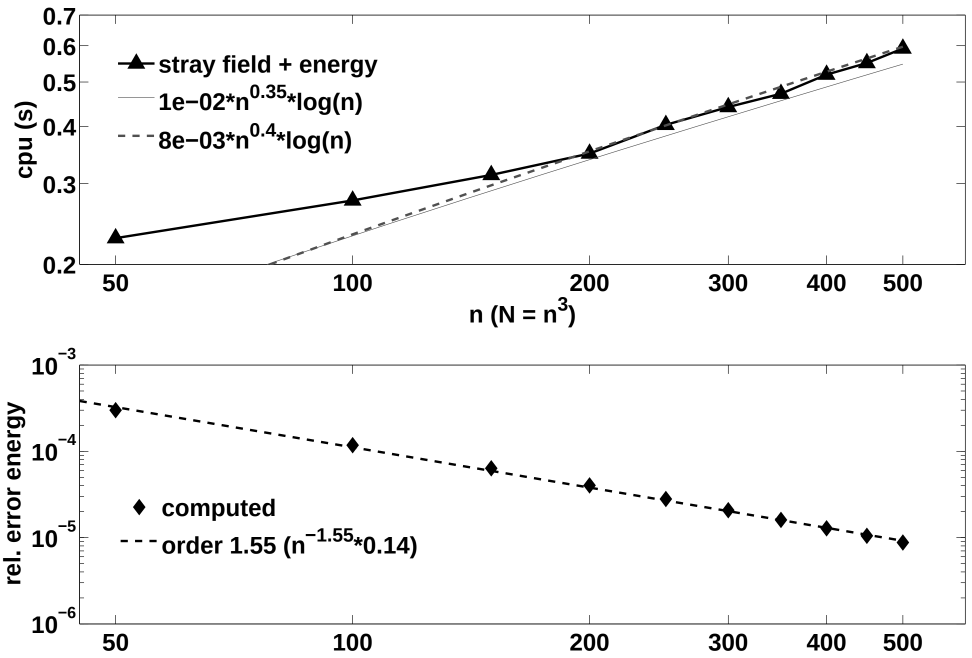

Figure 5.6: Complexity and relative error in the energy for Alg. 11 tested for uniform magnetization. *Reprinted from [26].*

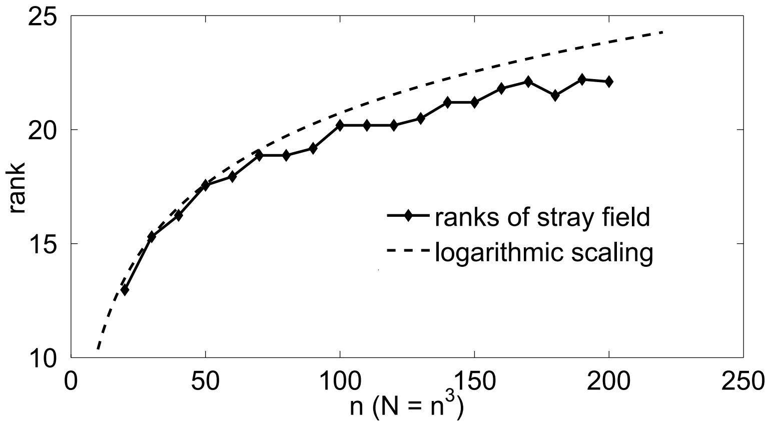

Figure 5.7: Ranks of the computed stray field using Alg. 11 with accuracy 1e-12 and a flower-like magnetization state. *Reprinted from [26].*

ity as other FFT based stray field algorithms. However, an advantage of Alg. 11 is the lower storage requirements for the kernels $\mathcal{P}^{(q)}$ *from* (5.75)*. It is only* $\mathcal{O}(R\sum_{q=1}^{3} n_q)$ *since the DFT of the kernels* (5.75) *can be carried out by Lemma 3 (Sec. 4.5) and result in CP tensors. This is much lower than for the demagnetizing-tensor or scalar potential method from [18]. For the elementwise product in Fourier space, the entries of the kernel have to be re-calculated from their factorized representations at a cost of* $\mathcal{O}(R)$ *operations.*

Chapter 6

Approximation of Magnetization in the Tucker format

An optimization procedure for the total energy that works with structured tensors for the magnetization components would benefit from low compression ranks, i.e. the complexity is lower, and of course also storage is less. On the other hand, the realization of such an algorithm is not straight forward at all. All the energy and gradient terms have to be calculated with tensor structured magnetization components, see chapter 5 for the stray field (the most tricky part). Also, all the operations in an existing minimization algorithm have to be performed within the tensor structure. Due to rank growing operations (like the Hadamard product), this is often only possible when approximate tensor arithmetics (truncation) is used, cf. Sec. 7.3. In this context most attention should be paid to the effective computation of element-wise operations (cf. chapter 4), which certainly plays an important role in any algorithmic realization of the point-wise constraint on the magnitude of the magnetization. For more information see chapter 7 where also alternative approaches for 'low-rank' minimization of the energy are introduced.

This section is dedicated to the question whether there exist 'low rank solutions' to micromagnetic minimization problems and especially the investigation of the behavior of the ranks of the magnetization components while hysteresis.

First, tensor approximation of a function related tensor which corresponds to data in a body with curved surface is briefly discussed in section 6.1 for the example of a sphere. Apart from that, Tucker compression for relaxed states[1], rank-growth with respect to the compression and optimization tolerance but also with respect to the applied field during demagnetizing curve calculation of a cubic (hard and soft) magnetic particle is analyzed in the subsequent sections.

---

[1] There holds $\mathcal{C}_{\boldsymbol{n},r} \subseteq \mathcal{T}_{\boldsymbol{n},r}$ since a canonical tensor can be identified with a Tucker tensor which has a superdiagonal core tensor. Hence, the 'Tucker-rank' ($r \equiv r_q$) is smaller or equal the 'CP-rank'.

Table 6.1: Tucker approximation of indicator function of a 'staircase approximation' of a sphere by Alg. 6 with tolerance 1e-06.

mode size n ($N = n^3$)	30	50	80	100	140	200
rank r ($r_q \equiv r$)	13	22	37	54	72	100

6.1 Approximation of the Indicator Function of a Sphere

Assume a Cartesian grid. By assigning the value 1 for tensor entries which correspond to grid points inside a sphere and zero else, one gets a tensor which is a discrete approximation of the indicator function χ of the sphere, see Fig. 6.1 left. Numerical tests show that a tensor approximation of a multivariate function with low tensor rank, e.g. something like $x^2 + y^2 + z^2$, and defined in a sphere is only as good as the tensor approximation of the indicator function χ. On the other hand, the tensor approximation of χ in the Tucker format by Alg. 6 yields a rather bad result, see Tab. 6.1. The approximation ranks are about $n/2$, which corresponds to a compression by a factor of about $7 - 8$ (about the ratio of the volume of the cube and the inscribed sphere). This means there is no effective compression possible.

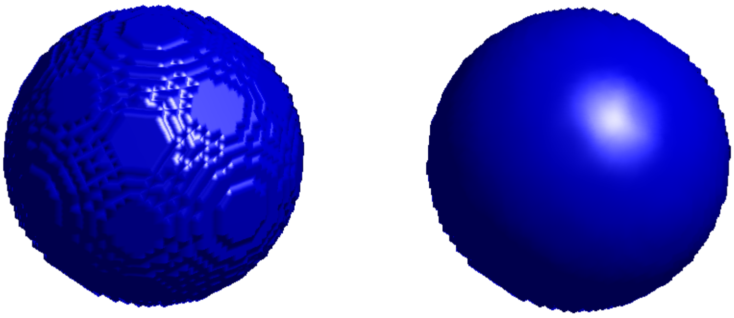

Figure 6.1: Left: 'Staircase approximation' of a sphere. Right: Approximation on the left but smoothed by Gaussian filter.

Fig. 6.1 also shows the staircase approximation convoluted with a Gaussian filter like in section 4.6. Remarkably enough, the smoothed sphere allows a slightly better Tucker approximation, e.g. for $n = 100$ the approximation ranks reduce from 54 to 40, for $n = 200$ from 100 to 68. This result depends on the parameters in the used filter, i.e. for the Gaussian filter as given in (4.42) the parameter m has to be large enough (i.e. the convolution result has to be smooth enough) in order to get a better approximation rank. On the other hand, larger m yields higher complexity in the (local) computation of the convolution. Fast convolution (using Fast Fourier transform and convolution theorem) could be considered instead.

Overall, one can not expect 'ϵ-accurate' low-rank approximations for function-related tensors

Table 6.2: Remanent states for uniform initial magnetization. Ranks $\bar{r}$ (average of ranks, i.e. $1/9\sum_{p=1}^{3}\sum_{q=1}^{3} r_p^q$) determined with tolerance 1e-14. $Q = 1$ and $n = 2.5\lambda$.

grid size n ($N = n^3$)	30	40	50	60	70	80	100
average rank $\bar{r}$	15	20	25	29.3	31	34	35.7

which 'live' in a body with curved surface. This result is disillusioning, and so, methods described in this thesis, which assume discrete representations of magnetization components as tensors refer to hexahedral magnets. Exceptions are the generic descriptions of minimization methods of chapter 3 and of course the finite element stray field method in chapter 8.

For the sake of completeness it should be mentioned that, in the special case of a sphere, spherical coordinates would lead to a tensor product domain. In the words of W. Hackbusch [47, p.464]:

'Tensor applications require a Cartesian grid $I := I_1 \times \ldots \times I_d$ of unknowns. This does not mean that the underlying domain $\Omega \subset \mathbb{R}^d$ must be of product form. It is sufficient that Ω is the image of a domain $\Omega_1 \times \ldots \times \Omega_d$. For instance, Ω may be a circle, which is the image of the polar coordinates varying in $\Omega_1 \times \Omega_2$.'

6.2 Approximation for zero External Field

In this section initial states in the unit cube are relaxed using the energy minimization algorithm 2 form section 3.3.1 with no external field and anisotropy axis parallel to the $z-$axis. Their compression ranks are investigated for given tolerances. First, the initial state is chosen to be uniform in z-direction, i.e. $m_x = m_y = 0$, $m_z = 1$. Hard magnetic material is assumed: for this test $Q = 1$ (cf. section 2.2), while the length of the cube λ (and so the exchange constant) is varied. L is set to 1, which corresponds to the unit cube. The number of grid points in one direction, i.e. n, is varied according to $n = 2.5\lambda$. Since λ has units of exchange length l_{ex}, this means that the exchange length is 2.5 times larger than the grid-spacing, cf. section 2.2. Tab. 6.2 shows the results for compression ranks with relative accuracy in the Frobenius norm below ϵ = 1e-14. Stopping criterion for the optimization procedure was a maximum norm of the projected gradient less than 1e-9 (very tight). The relaxed magnetization configurations were all symmetric flower states [12] with average magnetization $\langle m_x \rangle = \langle m_y \rangle = 0$, $\langle m_z \rangle = 0.9981$. From Tab. 6.2 one observes linear rank-growth for small λ and gradually logarithmic increase for larger cubes. Anyway, it is remarkable that there exists a low-rank approximation with an accuracy of almost double machine precision (1e-14).

In [18] similar tests were performed for vortex-like magnetization; also rank-growth for increasing λ was observed. Moreover, Tucker compression turned out to be more efficient than

Table 6.3: Average ranks for Tucker approximation with tolerance ϵ for remanent states of randomly disturbed uniform initial magnetization for different stopping tolerance (opt-tol) of the maximum norm of the projected gradient. Grid size $n = 50$, $\lambda = 35$ and $Q = 1/2$.

tolerance ϵ	1e-9	1e-8	1e-7	1e-6	1e-5	1e-4	
opt-tol 1e−06	50	50	50	50	48.7	27.7	(average rank $\bar{r}$)
opt-tol 1e−08	49	41.3	27	17	6.6	4	(average rank $\bar{r}$)
opt-tol 1e−10	47.7	21	11	7	6	4	(average rank $\bar{r}$)

using CP format. A canonical ALS approximation of the relaxed magnetization components in Tab. 6.2 with rank $\lceil \bar{r} \rceil$ yields relative errors in the range of 1e-5 – 1e-7.

The minimization of the energy can regularize initial magnetization, such that the relaxed state is smooth enough to allow a low rank approximation (even though the initial state does not allow it). In order to demonstrate this, uniform magnetization, i.e. $m_x = m_y = 0, m_z = 1$, is disturbed randomly, i.e. random numbers from the interval [−1e-2,1e-2] are added to each component of the magnetization at each node. Afterwards, this configuration is normalized and taken as initial state for minimization by Alg 2. Note, that this disturbed magnetization has full (Tucker) ranks for approximation tolerances much below 1e-2, e.g. for < 1e-5. The optimization again leads to symmetric flower states, but this time for $Q = 1/2$ with average magnetization $\langle m_x \rangle = \langle m_y \rangle = 0$, $\langle m_z \rangle = 9.927\text{e-}01$. Tab. 6.3 shows the (average) rank of the z-component of the relaxed state for different stopping tolerance of the maximum norm of the projected gradient. The more relaxed the resulting state is, i.e. tighter stopping criterion in the optimization, the better Tucker compression for given tolerance is possible. On the other hand, tighter stopping criteria in an optimization procedure that uses Tucker compressed magnetization components for its internal functions (e.g. gradient calculation, line search) demands for more accurate compression. For instance, this consideration concerns the stopping criteria in demagnetizing curve calculations, i.e. the optimization tolerance which has to be reached before the applied field is reduced by some Δh. If one feels certain about the choice of the stopping tolerance (together with the decrement value Δh), only compression tolerances below the stopping tolerance are valid. Otherwise white noise disturbs the calculation. The next section is dedicated to this and similar considerations related to calculation of demagnetizing curves (hysteresis).

6.3 Approximation during Demagnetization

In this section the compression of the magnetization components is determined during demagnetization through an external field (applied field) that starts with some positive magnitude and is then gradually decreased, i.e. the states during demagnetization are analyzed regarding their

(Tucker) compression properties. Alg. 2 is used for optimization, where this algorithm operates on dense/full magnetization component tensors. The relaxed states are compressed by Alg. 6 and used as new initial states in full tensor format for the minimization of the energy with reduced external field. Hence, the tensor compression in this section only has the effect of 'rounding' the already relaxed states to the prescribed accuracy in HOOI Alg. 6, whereas the numerical optimitaion is performed in double machine precision (full tensor format). Of course, this procedure has no computational advantages over the original algorithm , but it may help one to understand the effect of compression (used as truncation operation) in a minimization scheme for the total energy. The Tucker compression accuracy is therefore tighter than the used stopping criteria in the optimization scheme. Clearly, the opposite case would make no sense at all. The tests in this section reflect the behavior of a version of Alg. 2 that produces, up to the compression tolerance, the same results like the original algorithm.

The first test takes a $L = 70$nm cube with uniaxial anisotropy axis in z-direction, $J_s = 1.61$T, $K_1 = 4.3\text{e+}6\ J/m^3$ and $A = 7.3\text{e-}12 J/m$, which corresponds to $Nd_2Fe_{14}B$ [9]. The material parameter give $Q = 2.08$ and a wall width parameter of $\sqrt{A/K_1} \sim 1.3$nm. Since the exchange length is $l_{\text{ex}} = 2.6$nm, larger than the wall width (hard magnetic case), a mesh size parameter $n = 50$ is considered to yield a sufficient fine discretization, i.e. grid spacing ~ 1.4nm about half of l_{ex} and about the wall width. The external field axis is parallel to the vector $(1, 0, 10)^T$, so it is nearly parallel to the anisotropy axis. Subsequent equilibrium states are now computed by Alg. 2 for decreasing external field $h = \|\boldsymbol{h}_{ext}\| \in [-3.5, 1]$, where the value is reduced by $\Delta h = 5\text{e-}3$ if the maximum norm of the projected gradient is below 1e-6. Tucker compression is performed using Alg. 6 with tolerance 1e-7. Fig. 6.3 shows average ranks and errors as a function of compression tolerance (cf. Alg. 6) for the z-component of different states in the demagnetizing curve, which itself is shown in Fig. 6.2.

In Fig. 6.3 one clearly recognizes rank-growth as h is approaching the coercive field. Away form this value only very slow increase with respect to decreasing h is observed. For the relation between average compression ranks $\widetilde{r}$ and the tolerance `tol` in the compression there holds approximately $\widetilde{r} \sim \mathcal{O}(\log \texttt{tol}^{-1})$ if h is not near the coercive field.

Also note that the maximum observed ranks were $\boldsymbol{r} = (50, 25, 50)$ for all magnetization components.

It is not far to seek an application of Tucker compressed magnetization in an optimization scheme for the purpose of calculating hysteresis, at least for external field values away from the coercive field. This critical field value could be detected by (exploding) rank-growth, see Fig. 6.3 and 6.4, but also compare with Fig. 6.5, where the ranks do not grow while approaching the coercive field (only right before, i.e. $h \sim h_c$) due to a too loose stopping tolerance in the optimization.

Fig. 6.4 shows average ranks as function of the external field for Tucker compression with tol-

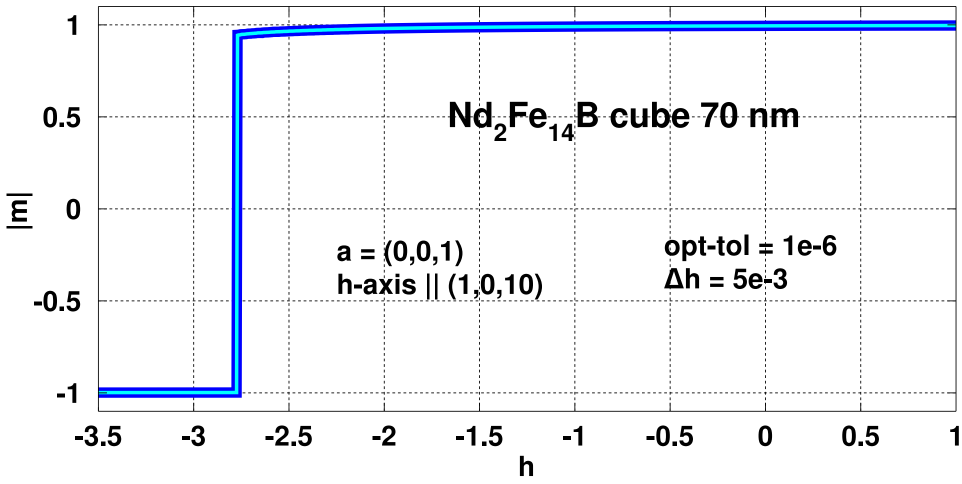

Figure 6.2: Demagnetizing curve of 70nm $Nd_2Fe_{14}B$ cube, i.e. projection of magnetization onto field axis $(1, 0, 10)^T$ ($|m|$) as function of external field h. The brighter curve is calculated with Tucker compression with tolerance 1e-7, whereas the dark blue curve uses full magnetization. The calculated coercive field is in both cases $h_c = -2.767$. Uniaxial anisotropy axis is $(0, 0, 1)^T$, $\Delta h = 0.005$, opt-tol $=$ 1e-6.

erance 1e-7 in the computation of the demagnetizing curve in Fig. 6.2. Fig. 6.4 shows larger ranks for the x and y component compared to the z component, which is also true for the most part in Fig. 6.3. Right before and during switching of the magnetization (i.e. near the coercive field) the ranks 'explode' in an oscillating manner.

The choice of a small enough stopping tolerance for given Δh is important to calculate a good approximation of the coercive field. Fig. 6.5 shows, in the same case like in Fig. 6.2-6.4 ($Nd_2Fe_{14}B$), the ranks and the demagnetizing curve for a higher stopping tolerance (opt-tol) of 1e-5. The calculated coercive field is $h_c = -2.955$. Although the Tucker approximation tolerance can now be increased to e.g. 1e-6, which leads to lower ranks, the calculated value of the coercive field exceeds that from the previous calculation (opt-tol $=$ 1e-6) by 6.8 %. Also, the ranks do not grow while approaching the coercive field due to a too loose stopping tolerance in the optimization. Only right before the switching the ranks grow to saturation.

On the other hand, performing the same test but with a tighter stopping tolerance (opt-tol) of 1e-7 and Tucker approximation tolerance 1e-8 yields the coercive field $h_c = -2.752$ which differs from that corresponding to opt-tol $=$ 1e-6 by only 0.5 %. Fig. 6.6 shows this test, where one clearly recognizes high ranks throughout the whole simulation.

A smaller cube with $L = 35$nm with stopping tolerance (opt-tol) 1e-6 and Tucker approximation tolerance 1e-7 is shown in Fig. 6.7; still the mesh size is $n = 50$. Compared to the 70nm cube in Fig. 6.4 the ranks grow faster.

Interestingly, the opposite seems to be true for a large sample, i.e. for $L = 100$nm, $n = 80$ and the material parameters of $Nd_2Fe_{14}B$. Fig. 6.8 shows, for stopping tolerance (opt-tol) 1e-6 and

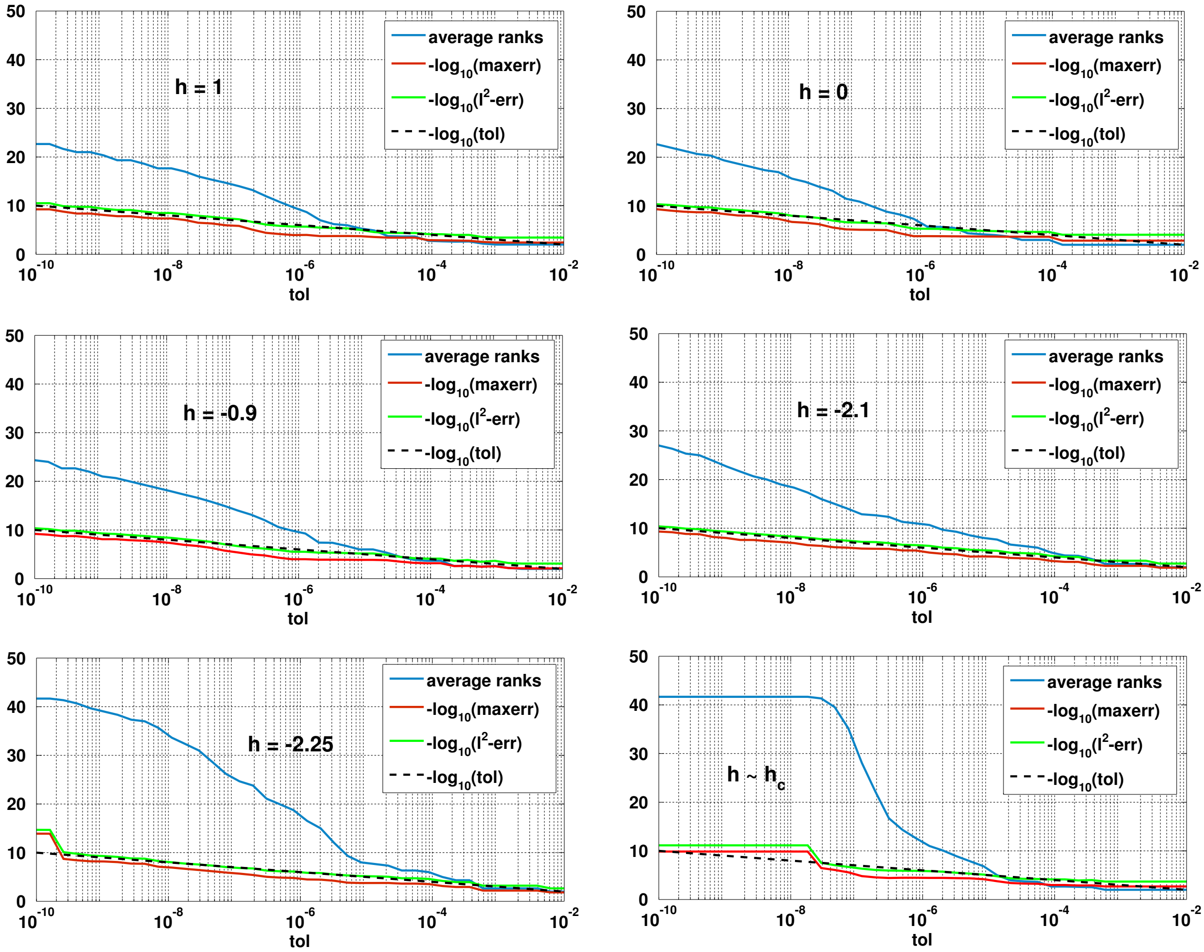

Figure 6.3: Average ranks and corresponding error (maximum- and l^2-norm) as a function of compression tolerance (tol) for the z-component of different states in the demagnetizing curve (cf. Fig. 6.2) for a 70nm $Nd_2Fe_{14}B$ cube. Axis of applied field is parallel to $(1, 0, 10)^T$, uniaxial anisotropy axis is $(0, 0, 1)^T$, $\Delta h = 0.005$, opt-tol $=$ 1e-6.

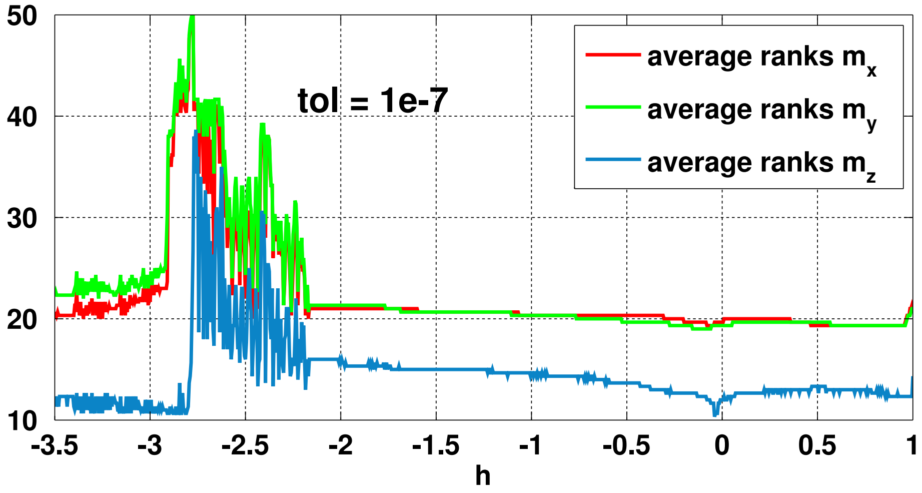

Figure 6.4: Average ranks of the magnetization components in the demagnetizing curve (cf. Fig. 6.2 bright curve) as a function of external field. Compression tolerance used in HOOI is 1e-7.

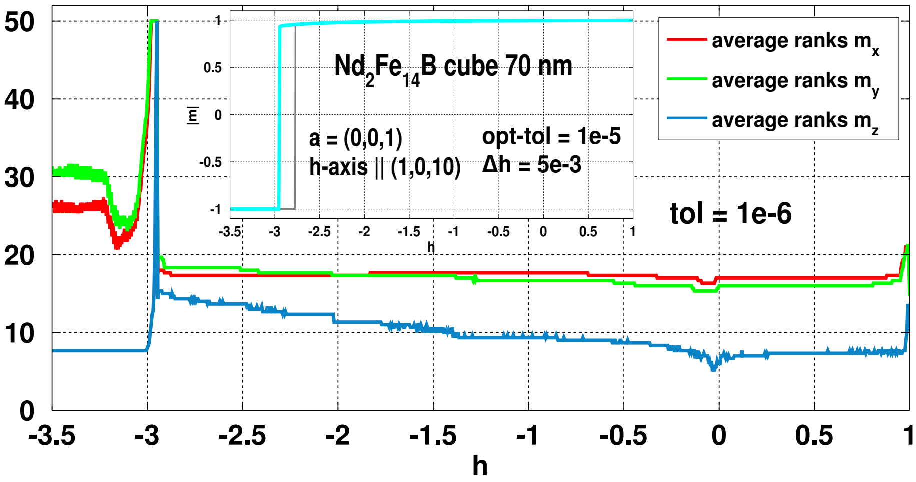

Figure 6.5: Average ranks of the magnetization components in the demagnetizing curve (inset) of a 70nm $Nd_2Fe_{14}B$ cube as a function of external field. Compression tolerance used in HOOI is 1e-6. The stopping tolerance in the optimization is 1e-5.

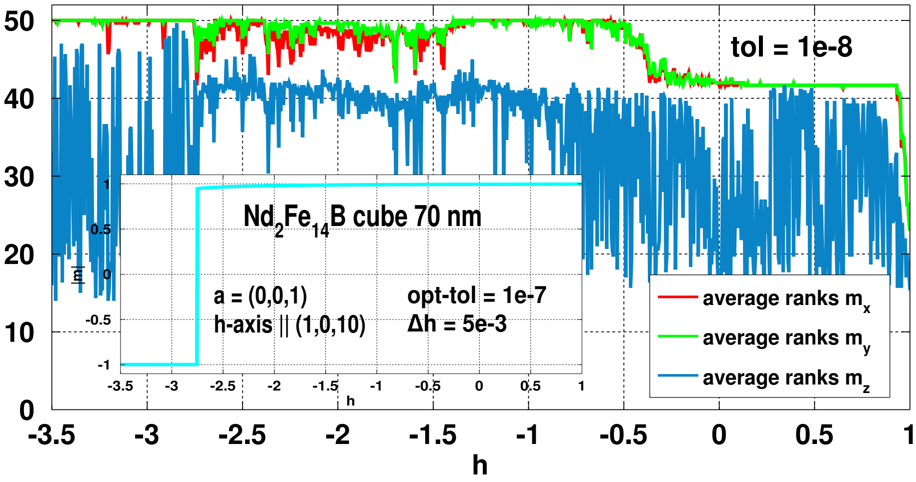

Figure 6.6: Average ranks of the magnetization components in the demagnetizing curve (inset) of a 70nm $Nd_2Fe_{14}B$ cube as a function of external field. Compression tolerance used in HOOI is 1e-8. The stopping tolerance in the optimization is 1e-7.

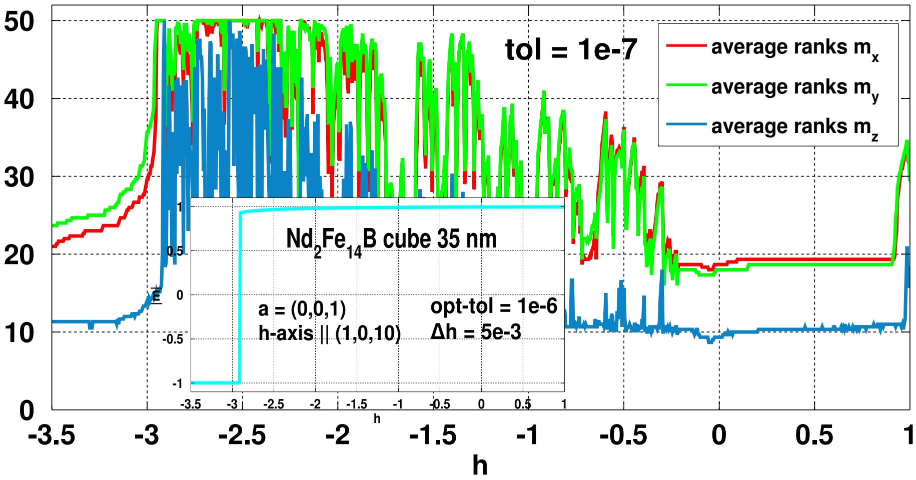

Figure 6.7: Average ranks of the magnetization components in the demagnetizing curve (inset) of a 35nm $Nd_2Fe_{14}B$ cube as a function of external field. Compression tolerance used in HOOI is 1e-7. The stopping tolerance in the optimization is 1e-6.

Tucker approximation tolerance 1e-7, increasing ranks up to $h \sim -0.8$ and smaller ranks afterwards until h reaches the coercive field (calculated value $h_c = -2.727$). There are similarities to Fig. 6.5 where the stopping tolerance was too loose.

Fig. 6.9 shows the same as Fig. 6.4 but with different anisotropy $Q = 0.05$, which corresponds to a soft magnetic material. One observes rapid rank growth for all components as h approaches the coercive field.

For different applied field axis, e.g. 45 degree to the anisotropy axis, tests similar to those

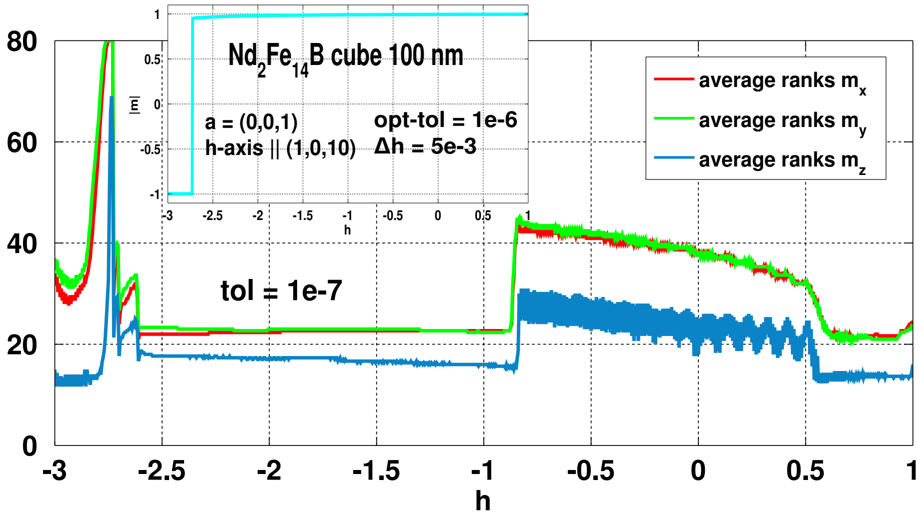

Figure 6.8: Average ranks of the magnetization components in the demagnetizing curve (inset) of a 100nm $Nd_2Fe_{14}B$ cube as a function of external field. Compression tolerance used in HOOI is 1e-7. The stopping tolerance in the optimization is 1e-6.

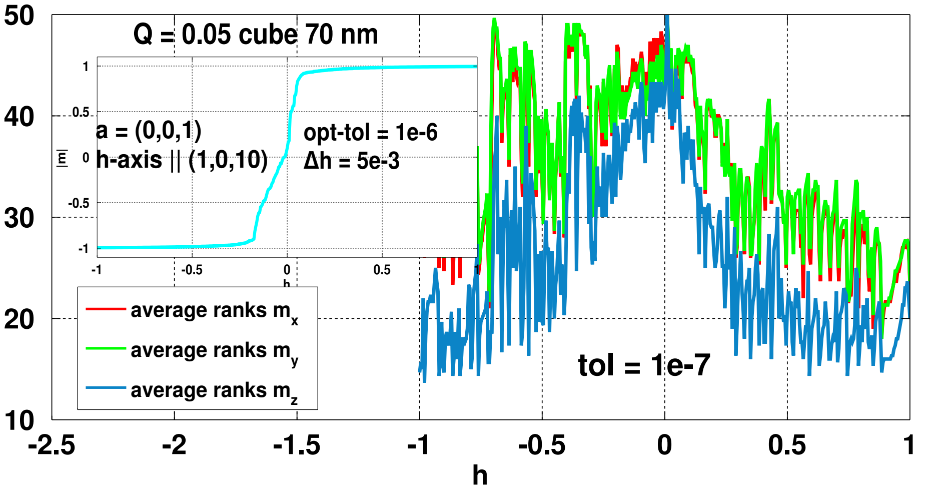

Figure 6.9: Average ranks of the magnetization components in the demagnetizing curve (inset) of a soft magnetic cube as a function of external field. Compression tolerance used in HOOI is 1e-7. The stopping tolerance in the optimization is 1e-6.

performed in this section show higher and more rapidly increasing ranks.

6.4 Conclusions

The considerations of sections 6.1 indicate difficulties for 'ϵ-accurate' low-rank approximations for function-related tensors which 'live' in a body with curved surface. Micromagnetic methods for curved geometries, which assume discrete representations of magnetization components as tensors, are therefore rather inconceivable. Also the results of the subsequent sections strongly indicate that 'ϵ-accurate' low-rank approximation is absolutely necessary in order to be able to implement such rounding/truncation in an optimization routine. These methods strongly depend on calculations that are at least as accurate as a stopping tolerance[2]. On the other hand, the choice of a small enough stopping tolerance for given Δh is important to calculate a good approximation of the coercive field, compare Fig. 6.4, 6.5 and 6.6.

The tests in section 6.2 indicate an asymptotically logarithmic rank-growth with respect to the side-length of a (rather) hard magnetic cube (with no external field). It is also shown that the minimization of the energy has a regularizing effect on randomly disturbed initial magnetization.

In section 6.3 the compression ranks corresponding to a prescribed tolerance during demagnetization are adaptively determined. The dependence of the ranks with respect to the tolerance `tol` for the compression is found to be approximately $\widetilde{r} \sim \mathcal{O}(\log 1/\mathtt{tol})$ if h is not near the coercive field. Right before and during switching of the magnetization (i.e. near the coercive

---

[2] Such a tolerance is usually in the range of 1e-8 - 1e-5 for a maximum norm used

field) the ranks 'explode' in an oscillating manner, while for the region away from the critical field ranks do not grow drastically. Anyway, the field range, where ranks can be small and do not increase too fast, is large for hard magnetic materials. This gives one the possibility to apply compressed magnetization in an optimization scheme for the purpose of calculating hysteresis, at least in the restricted cases where this is possible in principle[3]. Soft magnetic material instead shows (as it is well known) a very narrow hysteresis loop and, as shown in the previous section, rapid rank-growth towards the field value were the magnetization switches.

In the context of optimization methods for 'low-rank tensor magnetization', this chapter was dedicated to *minimization of the energy with truncated operations*; in Ch. 7 also alternatives are discussed.

---

[3] E.g. geometry restriction or storage problem for element-wise operations in Tucker format

Chapter 7

Energy Minimization with Structured Tensors

Recall that a *stable magnetization configuration* $\boldsymbol{m}^*$ is an isolated local solution to (also compare with Sec. 3.1 and 3.2)

$$\min_{\boldsymbol{m}} \psi_t(\boldsymbol{m}) \quad \text{subject to} \quad \|\boldsymbol{m}\| = 1 \quad \text{point-wise.} \tag{7.1}$$

In this chapter the focus is on solving (7.1) within either the CP or the Tucker format for *low representation ranks*. This is interesting not only for the purpose of accelerating existing methods like those already introduced in chapter 3, but also to make energy minimization possible where the degrees of freedom exceed the capacities of state-of-the-art methods. In fact, in the case of rather small or medium sized systems, methods for structured tensor magnetization can be expected to be less efficient[1] than most of the methods in chapter 3. Reasons for that are the complications in connection with the additional side constraint $\mathcal{M} \in$ *structured tensor field* and, especially, the fulfillment of the unit norm constraint. This might force one to use the numerically inferior *penalty formulation*, cf. Sec. 3.4.1. Nevertheless, energy minimization with structured tensors can yield cheaper iterations and, above all, less storage requirements.

The discrete version of (7.1) for unstructured magnetization is a minimization problem consisting of a quadratic functional with a (point-wise) non-linear and non-convex side constraint. Inserting a structured tensor magnetization (fixed format) allows one to treat the problem (7.1) with respect to the parameters of the tensor format. As a consequence the discretized energy loses the property of being a (simple) quadratic function. The higher degree of non-linearity makes the numerical optimization more difficult. Moreover, the unit norm constraints can not be treated directly, e.g. by iterations that make use of re-normalization, but have to be incorporated into the problem formulation by a penalty term. Also, additional local minima can occur

---

[1] In terms of number of iterations, convergence rate or function evaluations.

which are not local solutions to (the discretized version of) (7.1). In the case of CP tensors for the parametrization of the magnetization, the non-closedness of $C_{\boldsymbol{n},r}$, $r \geq 2, |\boldsymbol{n}| \geq 3$ might lead to non-stable iterations. Regularization can be considered to overcome this problem.
A second notion of minimizing (7.1) in a data-sparse fashion is the use of truncated iterations from optimization methods from Sec. 3. This idea is briefly discussed in Sec. 7.3, however, some numerical analysis was already done in Ch. 6 in Sec. 6.2 and 6.3.
Both notions of minimizing the energy with structured tensors (within the parameter set or 'exterior' with truncation) are briefly described in Sec. 7.3 and 7.4. An intermediate procedure is introduced in Sec. 7.5. *Successive rank-k updates* within a penalty setup do not suffer too much from rapid rank growth like truncation methods and the degree of non-linearity in the optimization is lower than for optimization with fixed prescribed ranks.
In the following section 7.1 the discrete energy on a tensor grid is obtained. Sec. 7.2 discusses discrete optimization on tensor spaces and subsets.

7.1 Discretization of the Energy on a Tensor Grid

In order to solve the micromagnetic energy minimization problem (7.1) the energy (2.17) has to be discretized. The magnetization on a tensor grid is generally given as a tensor field $\mathcal{M} = (\mathcal{M}^{(1)}, \mathcal{M}^{(2)}, \mathcal{M}^{(3)})$ where $\mathcal{M}^{(p)} \in \bigotimes_{q=1}^{3} \mathbb{R}^{n_q}$, $p = 1 \ldots 3$, also compare with Sec. 5.1.4. For the sake of simpler notation a uniformly discretized unit cube with subcubes of volume $1/n^3$ is assumed. In the more general case of $\boldsymbol{n} = (n_1, n_2, n_3)$ the following definitions and derivations can be easily adapted.
The energy (2.17) is discretized using the midpoint rule, i.e.

$$\begin{aligned}\psi_t \approx \Psi_t(\mathcal{M}) = &-\frac{\widetilde{A}}{n^3}\sum_{p=1}^{3}\sum_{q=1}^{3}\langle \mathcal{M}^{(p)}, \mathcal{M}^{(p)} \times_q \boldsymbol{K}_n\rangle - \frac{1}{2n^3}\sum_{p=1}^{3}\langle \mathcal{M}^{(p)}, \mathcal{H}_d^{(p)}(\mathcal{M})\rangle \\ &+ Q(1 - \frac{1}{n^3}\sum_{p=1}^{3}\sum_{q=1}^{3} a^{(p)}a^{(q)}\langle \mathcal{M}^{(p)}, \mathcal{M}^{(q)}\rangle) - \frac{1}{n^3}\sum_{p=1}^{3}\langle \mathcal{M}^{(p)}, \mathcal{H}_{ext}^{(p)}\rangle,\end{aligned} \tag{7.2}$$

where $\boldsymbol{K}_n \in \mathbb{R}^{n \times n}$ is the central three-point discretization of the second derivative with *Neumann boundary conditions* and $\boldsymbol{a} \in \mathbb{R}^{3\times 1}$ the unit vector of the easy axis. The expression (7.2) is a second order approximation to (2.17). All operations in (7.2) can be performed efficiently if the components of $\mathcal{M}$ are structured tensors.
According to Sec. 5.1.5 the stray field components are approximated by

$$\mathcal{H}_d^{(p)}(\mathcal{M}) = \frac{1}{2\pi^{3/2}}\sum_{q=1}^{3}\sum_{l=1}^{R} \mathcal{N}_p^l(\mathcal{M}^{(q)}), \tag{7.3}$$

with

$$\mathcal{N}_p^l(\mathcal{M}^{(q)}) = \mathcal{M}^{(q)} \times_1 \widetilde{\boldsymbol{D}}_1^l \times_2 \widetilde{\boldsymbol{D}}_2^l \times_3 \widetilde{\boldsymbol{D}}_3^l \times_p \boldsymbol{J}_n, \tag{7.4}$$

and

$$\widetilde{\boldsymbol{D}}_j^l := (\omega_l \sinh^2 \tau_l)^{1/3} \boldsymbol{D}_j^l. \quad \text{(cf. Sec. 5.1.5)} \tag{7.5}$$

Remember that in the case of structured tensors the evaluation of $\mathcal{N}_p^l$ simplifies, e.g. for $\mathcal{M}^{(q)} = [\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!] \in \mathcal{C}_{n,r}$ and $p = 1$

$$\mathcal{N}_1^l(\mathcal{M}^{(q)}) = [\![\boldsymbol{J}_n \widetilde{\boldsymbol{D}}_1^l \boldsymbol{U}^{(1)}, \widetilde{\boldsymbol{D}}_2^l \boldsymbol{U}^{(2)}, \widetilde{\boldsymbol{D}}_3^l \boldsymbol{U}^{(3)}]\!] \in \mathcal{C}_{n,r}. \tag{7.6}$$

Remark 8. *The discrete energy (7.2) is a quadratic functional in the sense that there holds*

$$\Psi_t(\boldsymbol{m}) - Q = \frac{1}{2}\boldsymbol{m}^T \boldsymbol{H} \boldsymbol{m} - \boldsymbol{h}^T \boldsymbol{m}, \tag{7.7}$$

where $\boldsymbol{m} = (vec(\mathcal{M}^{(1)})^T, vec(\mathcal{M}^{(2)})^T, vec(\mathcal{M}^{(3)}))^T$, $\boldsymbol{h} = 1/n^3(vec(\mathcal{H}^{(1)})^T, vec(\mathcal{H}^{(2)})^T, vec(\mathcal{H}^{(3)}))^T$ *and the symmetric (block) Kronecker product structured Hessian*

$$\boldsymbol{H} = -\frac{1}{n^3}(\widetilde{A}\boldsymbol{K} - \frac{1}{2}\boldsymbol{N} + 2Q\boldsymbol{A}), \tag{7.8}$$

with the symmetric $3n^3 \times 3n^3$ *block matrices*

$$\begin{aligned}
\boldsymbol{K} &= diag(\Delta_{n^3}, \Delta_{n^3}, \Delta_{n^3}),\ \Delta_{n^3} = (\boldsymbol{K}_n + \boldsymbol{K}_n^T) \otimes I_n \otimes I_n + I_n \otimes (\boldsymbol{K}_n + \boldsymbol{K}_n^T) \otimes I_n + I_n \otimes I_n \otimes (\boldsymbol{K}_n + \boldsymbol{K}_n^T),\\
\boldsymbol{N} &= [\mathcal{P}^{(q)} \times_p \boldsymbol{J}_n + (\mathcal{P}^{(p)} \times_q \boldsymbol{J}_n)^T]_{p,q=1\dots3},\ cf.\ (5.63),\\
\boldsymbol{A} &= \boldsymbol{a}\boldsymbol{a}^T \otimes I_{n^3},
\end{aligned} \tag{7.9}$$

where $I_m \in \mathbb{R}^{m \times m}$ *is the identity matrix.*
As a consequence, the gradient of the discretized energy with respect to $\boldsymbol{m}$ *is* $\nabla_{\boldsymbol{m}} \Psi_t(\boldsymbol{m}) = \boldsymbol{H}\boldsymbol{m} - \boldsymbol{h}$ *and the Hessian is* $\nabla^2_{\boldsymbol{mm}} \Psi_t(\boldsymbol{m}) = \boldsymbol{H}$. *Hence, the action of the Hessian on a grid vector* $\boldsymbol{v}$ *is* $\boldsymbol{H}\boldsymbol{v} = \nabla_{\boldsymbol{m}} \Psi_t(\boldsymbol{v}) + \boldsymbol{h}$, *which allows the usage of iterative solvers (e.g. CG or GMRES) for linear systems containing the Hessian without explicit storage of* $\boldsymbol{H}$. *The Kronecker structure of* $\boldsymbol{H}$ *can only be exploited if* $\boldsymbol{m}$ *has a certain separable structure itself, e.g. one of the tensor formats of Sec. 4. Computational costs for evaluating the energy and its gradient are both dominated by the costs of the stray field part. However, preconditioning of a structured matrix like* $\boldsymbol{H}$ *is a serious task which is by far not trivial if the structure should be preserved. This topic will not be discussed in this work and therefore left for future research, especially in connection with a*

generalization of the Newton method described in Sec. 3.4.2 to structured tensor magnetization with help of truncated iterative procedures for solving linear systems in tensor format. □

Often an important ingredient for constraint optimization with tensor structured input is a (l^2-) penalty (or regularization) term (cf. Sec. 3.4.1), which, in the case of the point-wise unit vector constraint in micromagnetics takes the discrete form

$$P_{\boldsymbol{n}}(\mathcal{M}) = \frac{1}{n^3}\|\mathbf{1}_{\boldsymbol{n}} - \sum_{p=1}^{3} \mathcal{M}^{(p)^2}\|^2 = \frac{1}{n^3}\sum_{p=1}^{3} \|\mathcal{M}^{(p)^2}\|^2 + \frac{1}{n^3}\sum_{p=1}^{3}\sum_{\substack{q=1\\ q\neq p}}^{3} \langle \mathcal{M}^{(p)^2}, \mathcal{M}^{(q)^2}\rangle - \frac{2}{n^3}\sum_{p=1}^{3}\|\mathcal{M}^{(p)}\|^2 + 1, \tag{7.10}$$

where $\mathbf{1}_n$ is the tensor with all entries equal to one and the notation $\mathcal{M}^{(q)^2}$ stands for element-wise squaring. For the more general case of $\boldsymbol{n} = (n_1, n_2, n_3)$ the definition of $P_{\boldsymbol{n}}$ can be easily adapted.

7.2 Discrete Energy Minimization Problem on Tensor Spaces and Subsets

The optimization problem (3.11) from Sec. 3.2 can now be formulated for $\mathcal{M} = (\mathcal{M}^{(1)}, \mathcal{M}^{(2)}, \mathcal{M}^{(3)}) \in \left(\bigotimes_{q=1}^{3} \mathbb{R}^{n_q}\right)^3$, i.e.

$$\min_{\mathcal{M}\in\left(\bigotimes_{q=1}^{3}\mathbb{R}^{n_q}\right)^3} \Psi_t(\mathcal{M}) \quad \text{s.t.} \quad C(\mathcal{M}) := \mathcal{M}^{(1)^2} + \mathcal{M}^{(2)^2} + \mathcal{M}^{(3)^2} = \mathbf{1}_{\boldsymbol{n}}. \tag{7.11}$$

This problem formulation is equivalent to (3.11) and can be translated by vectorization, respectively tensorization. If the set of tensors in (7.11) is restricted to a certain tensor format $\mathcal{F} \subset \bigotimes_{q=1}^{3} \mathbb{R}^{n_q}$ the new problem reads

$$\min_{\mathcal{M}\in\mathcal{F}^3} \Psi_t(\mathcal{M}) \quad \text{s.t.} \quad C(\mathcal{M}) = \mathbf{1}_{\boldsymbol{n}}. \tag{7.12}$$

In general the fulfillment of the side constraint in (7.12) is not possible explicitly within the format $\mathcal{F}$, i.e. the ranks are increased. Hence, a penalty formulation might be considered

$$\min_{\mathcal{M}\in\mathcal{F}^3} \Psi_t(\mathcal{M}) + \mu P(\mathcal{M}), \tag{7.13}$$

for $\mu > 0$ increasing or large enough.

An important condition for the existence of a minimizer of a minimization problem restricted to a subset is the *closedness of this subset*. More precise there holds the following statement in finite dimension:

Lemma 9. *Let V be a normed vector space with dim$(V) < \infty$ and $\emptyset \neq U \subseteq V$ a closed subset. Further let $J : V \to \mathbb{R}$ be a continuous functional, and v^* a local minimizer of J, i.e. there exists a bounded neighborhood $T \subseteq V$ containing v^* with $J(v^*) \leq J(v)$ for all $v \in T$. Then* $\min_{u \in T \cap U} J(u)$ *exists if $T \cap U \neq \emptyset$.*

Proof. *The proof is similar to that of Lemma 2. Define for arbitrary $\widetilde{u} \in T \cap U$ the set $D := T \cap U \cap \{v \in V : J(v) \leq J(\widetilde{u})\} \subseteq T \cap U$, which is nonempty ($\widetilde{u} \in D$), bounded ($T$ bounded) and closed (J continuous) and hence compact (dim$(V) < \infty$). Due to the continuity, J attains its minimum in the compact set $D \subseteq T \cap U$. By definition of D a minimizer of J over D is also a minimizer of J over $T \cap U$.* □

Remark 9. *Due to the continuity of J one can assume w.l.o.g. that the neighborhood in the definition of a local minimizer is closed: For a sequence t_n in T one gets $J(v^*) \leq J(t_n)$ for all n and hence $J(v^*) \leq \lim_{n\to\infty} J(t_n) = J(\lim_{n\to\infty} t_n)$.*
Moreover, a similar consideration leads to the fact that the set $\{v \in V : J(v) \leq J(\widetilde{u})\}$ is closed.

Of course, one cannot decide beforehand whether the condition $T \cap U \neq \emptyset$ in Lemma 9 holds, since T will not be available in general. However, in the case of the minimization of a penalty formulation (7.13), where $V = \left(\bigotimes_{q=1}^{3} \mathbb{R}^{n_q}\right)^3$ and $\mathcal{F}^{(q)}$ is either the closure of $\mathcal{C}_{\boldsymbol{n},r_q}$, i.e. $\overline{\mathcal{C}}_{\boldsymbol{n},r_q}$ or a closed subset of the canonical tensors, e.g. $\mathcal{C}^c_{\boldsymbol{n},r_q}$ (see below), a sufficiently large rank ensures the existence of a local minimizer in the set $\times_{q=1}^{3} \mathcal{F}^{(q)} \cap T$. For this, consider that by definition of $\bigotimes_{q=1}^{3} \mathbb{R}^{n_q}$ as the vector space generated by all linear combinations of rank-1 tensors[2], for each element $v \in T \subseteq \left(\bigotimes_{q=1}^{3} \mathbb{R}^{n_q}\right)^3$ there exist $r^{(q)} \in \mathbb{N}$, $q = 1, 2, 3$ such that each $v^{(q)}$ has rank $r^{(q)}$. Thus, one only has to choose some $\mathcal{F}^{(q)} \supseteq \mathcal{C}_{\boldsymbol{n},r_q}$, $q = 1, 2, 3$.
Note, however, that the optimistic aim is to find an approximation to the micromagnetic energy minimization problem with *low* rank.
For the sake of completeness, it is mentioned here that a closed subset of canonical tensors are those with rank-1 terms bounded by a constant $c > 0$, i.e.

$$\mathcal{C}^c_{\boldsymbol{n},r} := \{\mathcal{X} = \sum_{j=1}^{r} \boldsymbol{u}_j^{(1)} \circ \boldsymbol{u}_j^{(2)} \circ \boldsymbol{u}_j^{(3)} \in \mathcal{C}_{\boldsymbol{n},r} : \sum_{j=1}^{r} \left\| \boldsymbol{u}_j^{(1)} \circ \boldsymbol{u}_j^{(2)} \circ \boldsymbol{u}_j^{(3)} \right\|^2 = \sum_{j=1}^{r} \prod_{q=1}^{3} \left\| \boldsymbol{u}_j^{(q)} \right\|^2 \leq c\}. \quad (7.14)$$

In fact, the best approximation problem $\min_{\mathcal{X} \in \mathcal{C}_{\boldsymbol{n},r}} \|\mathcal{X} - \mathcal{Y}\|$ is unsolvable if and only if infimum sequences are unstable, i.e. their rank-1 terms get unbounded, see §9.4 and §9.5.3 in reference [47].

---

[2] See e.g. §3.2.6.1 in reference [47].

In the context of optimization problems on $C^c_{n,r}$, the constraint of bounded terms can be formulated as a regularization/penalty term, i.e.

$$p_{C^c_{n,r}}(\mathcal{X}) = \frac{1}{2\,\|\mathcal{X}\|^2} \sum_{j=1}^{r} \prod_{q=1}^{3} \left\|\boldsymbol{u}_r^{(q)}\right\|^2 , \tag{7.15}$$

where a constant and moderately sized penalty parameter might be sufficient [79].

Another question is the non-uniqueness of the CP format due to scaling indeterminacy, i.e. $[\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!] = [\![\frac{1}{1+\epsilon}\boldsymbol{U}^{(1)}, (1+\epsilon)\boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]$. In the context of optimization problems this means that there exists a continuous manifold of equivalent solutions (as opposed to isolated local solutions). This might cause numerical difficulties, which, however, might be regularized by a *Tikhonov term* which is added to the objective function [51]

$$f_R([\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]) = f([\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]) + \frac{\mu_R}{2} \sum_{q=1}^{3} \left\|\boldsymbol{U}^{(q)}\right\|_F^2 , \tag{7.16}$$

where f is an objective function [e.g. (7.13)] and $\mu_R > 0$ a moderately sized regularization parameter. The effect of the Tikhonov term is discussed in [51] in the context of optimization based CP approximation of tensors, which can also be applied to more general problems like (7.16): If an optimal solution is fixed except for scaling the j-th rank-1 term with constants $\alpha_{q,j}$ with $\prod_{q=1}^{3} \alpha_{q,j} = \prod_{q=1}^{3} \|\boldsymbol{u}_j^{(q)}\| =: \gamma_j$, i.e.

$$\boldsymbol{u}_j^{(1)} \circ \boldsymbol{u}_j^{(2)} \circ \boldsymbol{u}_j^{(3)} = \alpha_{1,j} \frac{\boldsymbol{u}_j^{(1)}}{\|\boldsymbol{u}_j^{(1)}\|} \circ \alpha_{2,j} \frac{\boldsymbol{u}_j^{(2)}}{\|\boldsymbol{u}_j^{(2)}\|} \circ \alpha_{3,j} \frac{\boldsymbol{u}_j^{(3)}}{\|\boldsymbol{u}_j^{(3)}\|}, \tag{7.17}$$

the only part which is varying is the regularization term. This leads to r independent problems of the form

$$\min \alpha_{1,j}^2 + \alpha_{2,j}^2 + \alpha_{3,j}^2 \quad \text{s.t.} \quad \alpha_{1,j}\alpha_{2,j}\alpha_{3,j} = \gamma_j \quad j = 1 \ldots r. \tag{7.18}$$

It can be shown that the best solution is given when all $\boldsymbol{u}_j^{(q)}$ have equal norms, i.e.

$$\|\boldsymbol{u}_j^{(1)}\| = \|\boldsymbol{u}_j^{(2)}\| = \|\boldsymbol{u}_j^{(3)}\| = {\gamma_j}^{1/3}, \quad \text{for all } j = 1 \ldots r, \tag{7.19}$$

which also implies equal Frobenius norms of the factor matrices.

However, μ_R has to be chosen carefully so that it is small enough not to negatively influence the optimization. Also, the Tucker format suffers from scaling indeterminacy, which might also be treated with a Tikhonov regularization.

Furthermore, the canonical format and the Tucker format both have permutation indeterminacy, which, however, leads to isolated equivalent minimizers in the context of solutions of optimization problems. Thus, this is not critical.

7.3 Minimization with Truncated Iteration

In principle all the numerical methods of chapter 3 could be adapted to the case where operations are performed in a data sparse format. Operations on structured tensor magnetization which increase the number of parameters[3] have to be 'rounded/truncated' to a format with fewer parameters, at least, at some point determined by storage capacity. The necessary truncation procedure has to be stable and cheap. Hence, Tucker tensors should be considered for truncated iterations, due to the lack of such stable rounding procedures in the CP case (non-closedness). Truncation can be performed for fixed format and ranks or adaptively, e.g. by Alg. 6. Re-normalization steps, which are used in some of the algorithms of Sec. 3 can be formulated as an optimization problem, which then can be performed for structured tensors. Nevertheless, the ranks during the optimization procedure (both for the whole energy minimization or only the re-normalization) can increase very fast and do not allow truncation without significant loss of accuracy. For instance, the computation of the core of the Hadamard product of Tucker tensors needs $\prod_{j=1}^{3} r_j r'_j$ stored numbers and the same amount of operations, cf. Sec 4.3. For squaring a (real double) tensor with rank $r_q \equiv r = 23$ this already exceeds $1GB$. This operation is used for evaluating the unit norm constraint. In order to accomplish such operations one has to further reduce the degrees of freedom in the core representation of the original Tuckers *before* performing e.g. a Hadamard product. This is generally not possible without losing accuracy. Together with the results from Sec. 6, this gives a serious limitation for the optimization approach via truncated iterations. However, the following section describes a possible way to treat re-normalization in the case where operations are performed within a structured format. Note that the two algorithms which were introduced in Sec. 3.3 both rely on re-normalization. In the case of Alg. 2 the normalization is the last step in (3.34).

7.3.1 The Constraint on the Magnitude as Optimization Problem

The re-normalization of structured tensor magnetization is generally not possible in a direct fashion. Beside the treatment of the micromagnetic side-constraint via a penalty term [cf. (7.10) and Sec. 3.4.1], one could alternatively make use of the following optimization problem formulation of the normalization procedure:

---

[3] E.g. the ranks of CP tensors get multiplied for the Hadamard product.

For $\boldsymbol{a} = (a_x, a_y, a_z)^T \neq \boldsymbol{0}$ find $\boldsymbol{m} = (m_x, m_y, m_z)^T$ such that

$$\min_{\boldsymbol{m}} \|\boldsymbol{m} - \boldsymbol{a}\|^2 \quad \text{subject to} \quad \|\boldsymbol{m}\|^2 = 1. \tag{7.20}$$

Setting the gradient of the Lagrange function for the projection problem $\mathcal{L}_P(\boldsymbol{m}; \lambda) = \|\boldsymbol{m} - \boldsymbol{a}\|^2 - \lambda(\|\boldsymbol{m}\|^2 - 1)$ with respect to $\boldsymbol{m}$ and λ equal to zero yields the necessary optimality condition $\lambda^* = 1 \pm \|\boldsymbol{a}\|$ and $\boldsymbol{m}^* = \mp\boldsymbol{a}/\, \|\boldsymbol{a}\|$. The Hessian $\nabla^2_{\boldsymbol{mm}}\mathcal{L}_P(\boldsymbol{m}^*; \lambda^*) = \mp\|\boldsymbol{a}\|\mathrm{id}$ is positive definite for the case $\boldsymbol{m}^* = +\boldsymbol{a}/\, \|\boldsymbol{a}\|$, thus, this is the minimizer of (7.20).
Solving (7.20) in a penalty formulation, i.e.

$$\boldsymbol{m}^*_{\mu_k} = \arg\min_{\boldsymbol{m}} \|\boldsymbol{m} - \boldsymbol{a}\|^2 + \mu_k(\|\boldsymbol{m}\|^2 - 1)^2 \tag{7.21}$$

for increasing μ_k represents an alternative to the normalization procedure, which now can be performed in tensor format (with truncated iterations), i.e. for tensors (7.21) reads

$$\mathcal{M}^*_{\mu_k} = \arg\min_{\mathcal{M}} \frac{1}{n^3} \sum_{p=1}^{3} \left\|\mathcal{M}^{(p)} - \mathcal{A}^{(p)}\right\|^2 + \mu_k\, P_{\boldsymbol{n}}(\mathcal{M}), \tag{7.22}$$

cf. (7.10).

7.4 Minimization within the Representation

In this section the discrete optimization problem (7.12) is solved with the additional side-constraint of tensor structured magnetization components with fixed ranks. Some of the subsections, especially the last one, rely on the canonical tensor format.

7.4.1 Reformulation for Fixed Format

Let $\mathcal{F}$ be a tensor format and $\rho_{\mathcal{F}} : \mathbb{P} \to \bigotimes_{q=1}^{3} \mathbb{R}^{n_q}$ be the corresponding map from a *parameter set* $\mathbb{P}$ into the tensor space. For instance, in the case of the canonical format $\rho_{CP}(\boldsymbol{p}) \equiv \rho_{CP}(\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}) := [\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]$, where the weights are assumed to be absorbed by the factor matrices.
Problem (7.12) is equivalent to

$$\min_{(\boldsymbol{p}_1,\boldsymbol{p}_2,\boldsymbol{p}_3)\in\mathbb{P}_1\times\mathbb{P}_2\times\mathbb{P}_3} \Psi_t\Big((\rho_{\mathcal{F}}(\boldsymbol{p}_1), \rho_{\mathcal{F}}(\boldsymbol{p}_2), \rho_{\mathcal{F}}(\boldsymbol{p}_3))\Big) \quad \text{s.t.} \quad C\Big((\rho_{\mathcal{F}}(\boldsymbol{p}_1), \rho_{\mathcal{F}}(\boldsymbol{p}_2), \rho_{\mathcal{F}}(\boldsymbol{p}_3))\Big) = \boldsymbol{1}_{\boldsymbol{n}}. \tag{7.23}$$

The same reformulation can be done for (7.13).
One is interested in the case where the number of parameters is less than the dimension of the tensor space $\bigotimes_{q=1}^{3} \mathbb{R}^{n_q}$. Clearly, the objective function of (7.23) is more complicated now than

in (7.12). One gets additional local solutions, which are in the kernel of the Jacobian of the format. For instance, the simple objective $\|.\|^2$ minimized for rank-1 tensors has infinite many equivalent minimizers, i.e. all rank-1 tensors where at least one factor is the zero vector. See also the discussion about scaling indeterminacy in Sec. 7.2.

One can now apply methods for nonlinear optimization to the penalty formulation of (7.23), see Sec. 7.4.3. Efficient computation of the gradients is discussed in the next section.

7.4.2 Derivatives involving Tensor Formats

Here it is briefly discussed how to determine gradients of functionals like those coming from the tensor grid discretization of the energy (7.2) with respect to tensor formats. For the sake of convenience, the techniques are explained for the case of canonical tensors, whereas generalizations to other formats like Tucker tensors or tensor trains can be derived along the same lines.

For instance, imagine the functional

$$f : \mathcal{C}_{n,r} \to \mathbb{R},\ \mathcal{A} := [\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!] \mapsto \langle \mathcal{A}, \mathcal{B} \rangle, \tag{7.24}$$

where for the sake of simplicity the weights $\boldsymbol{\lambda}$ of the format description in (4.9) are assumed to be absorbed by the factor matrices. The tensor $\mathcal{B}$ can be assumed to be represented in canonical format too. Hence, the CP format is determined by the parameters of the factor matrices, i.e. $\mathcal{A} = \rho_{CP}(\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)})$, where ρ_{CP} is the map

$$\rho_{CP}(\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}) := [\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]. \tag{7.25}$$

The CP representation is not unique (ρ_{CP} is not injective), e.g. $\rho_{CP}(\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}) = \rho_{CP}(\frac{1}{2}\boldsymbol{U}^{(1)}, 2\boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)})$, which implies that the Jacobian of ρ_{CP} does not have full rank. Especially for solutions of optimization problems, this means that there exists a continuous manifold of equivalent solutions (as opposed to isolated local solutions). This might cause numerical difficulties, which, however, can be regularized, see Sec. 7.2 for more details.

Nevertheless, since the canonical format (as well as the other tensor formats introduced in chapter 4) are multi-linear in the parameters, it is not too difficult to determine the derivative

$$\frac{\partial f(\rho_{CP}(\boldsymbol{p}))}{\partial \boldsymbol{p}} = \frac{\partial f}{\partial \mathcal{A}} \frac{\partial \rho_{CP}}{\partial \boldsymbol{p}}, \tag{7.26}$$

where $\boldsymbol{p} := (\mathrm{vec}(\boldsymbol{U}^{(1)})^T, \mathrm{vec}(\boldsymbol{U}^{(2)})^T, \mathrm{vec}(\boldsymbol{U}^{(3)})^T)^T$.

A practical strategy for calculating the gradient of functionals like f in (7.24) with respect to the factor matrices is calculating the Fréchet derivative. E.g. in order to determine the gradient of f w.r.t. $\boldsymbol{U}^{(1)}$ one writes

$$f([\![\boldsymbol{U}^{(1)} + \delta, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]) - f([\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]) = \langle [\![\delta, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!], \mathcal{B} \rangle. \tag{7.27}$$

The remaining strategy is to represent the arguments in the inner product as matrices. The formulas for matricization in [49] are helpful for that purpose. One gets

$$\langle [\![\delta, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!], \mathcal{B} \rangle = \langle \delta (\boldsymbol{U}^{(3)} \odot \boldsymbol{U}^{(2)})^T, \mathcal{B}_{(1)} \rangle_F = \langle \delta, \mathcal{B}_{(1)} (\boldsymbol{U}^{(3)} \odot \boldsymbol{U}^{(2)}) \rangle_F. \tag{7.28}$$

The gradient of f w.r.t. $\boldsymbol{U}^{(1)}$ is therefore

$$\partial_{\boldsymbol{U}^{(1)}} f(\mathcal{A}) = \mathcal{B}_{(1)} (\boldsymbol{U}^{(3)} \odot \boldsymbol{U}^{(2)}), \tag{7.29}$$

which, in the case that $\mathcal{B} = [\![\boldsymbol{V}^{(1)}, \boldsymbol{V}^{(2)}, \boldsymbol{V}^{(3)}]\!]$, simplifies to

$$\partial_{\boldsymbol{U}^{(1)}} f(\mathcal{A}) = \boldsymbol{V}^{(1)} (\boldsymbol{V}^{(3)} \odot \boldsymbol{V}^{(2)})^T (\boldsymbol{U}^{(3)} \odot \boldsymbol{U}^{(2)}) = \boldsymbol{V}^{(1)} ({\boldsymbol{V}^{(3)}}^T \boldsymbol{U}^{(3)} \bullet {\boldsymbol{V}^{(2)}}^T \boldsymbol{U}^{(2)}), \tag{7.30}$$

where $\bullet$ stands for element-wise multiplication of matrices. The costs for computing (7.30) are $\mathcal{O}(3 n r_A r_B + r_A r_B)$, where $n_q \equiv n$ and r_A, r_B are the ranks of $\mathcal{A}$ and $\mathcal{B}$, respectively.
Functionals that arise from the penalty formulation of the discrete energy (see Sec. 7.1) are of the form

$$f(\mathcal{A}) = \langle \mathcal{A}^l, \mathcal{B}^m \rangle, \; l \in \mathbb{N}, m \in \mathbb{N} \cup \{0\}. \tag{7.31}$$

$$g(\mathcal{A}) = \langle \mathcal{A} \times_1 \boldsymbol{J}_1 \times_2 \boldsymbol{J}_2 \times_3 \boldsymbol{J}_3, \mathcal{A} \rangle. \tag{7.32}$$

The notation $\mathcal{A}^l$ stands for l-times element-wise power (tensor Hadamard product). $\mathcal{B}^0 =: \mathbf{1}_{\boldsymbol{n}} = \mathbf{1}_{n_1} \circ \mathbf{1}_{n_2} \circ \mathbf{1}_{n_3}$ is therefore understood as the tensor with all entries equal to one, which is a rank-1 tensor. Note that, there hold the useful identities

$$\langle \mathcal{A}^l, \mathcal{B}^m \rangle = \langle \mathcal{A}^{l-1}, \mathcal{B}^m \bullet \mathcal{A} \rangle \tag{7.33}$$

$$\langle \mathcal{A} \times_q \boldsymbol{J}, \mathcal{B} \rangle = \langle \mathcal{A}, \mathcal{B} \times_q \boldsymbol{J}^T \rangle, \tag{7.34}$$

which allows one to express the functionals of Sec. 7.1 in the forms of f and g.
The derivative of f w.r.t. a factor matrix is given as

$$\partial_{\boldsymbol{U}^{(p)}} f(\mathcal{A}) = l\, \mathcal{D}_{(p)} (\odot_{q \neq p} \boldsymbol{D}^{(q)}) = l\, \boldsymbol{D}^{(p)} (\bullet_{q \neq p} {\boldsymbol{D}^{(q)}}^T \boldsymbol{U}^{(q)}) \in \mathbb{R}^{n_p \times r_A}, \tag{7.35}$$

where $\mathcal{D} = \mathcal{A}^{l-1} \bullet \mathcal{B}^{m} \equiv [\![\boldsymbol{D}^{(1)}, \boldsymbol{D}^{(2)}, \boldsymbol{D}^{(3)}]\!] \in C_{\boldsymbol{n}, r_A^{l-1} r_B^{m}}$. The symbols $\odot_q$ and $\bullet_q$ stand for successive Khatri-Rao [descending order of indices, cf. (7.29)] and Hadamard products of matrices, respectively. The costs for calculating the derivative (7.35) are $\mathcal{O}(3nr_A^l r_B^m + r_A^l r_B^m)$.
The derivative of g w.r.t. a factor matrix is given as

$$\partial_{\boldsymbol{U}^{(p)}} g(\mathcal{A}) = \boldsymbol{J}_p \boldsymbol{U}^{(p)} (\bullet_{q \neq p} \, \boldsymbol{U}^{(q)^T} \, \boldsymbol{J}_q^T \, \boldsymbol{U}^{(q)}) + \boldsymbol{J}_p^T \boldsymbol{U}^{(p)} (\bullet_{q \neq p} \, \boldsymbol{U}^{(q)^T} \, \boldsymbol{J}_q \, \boldsymbol{U}^{(q)}) \in \mathbb{R}^{n_p \times r_A}. \tag{7.36}$$

Calculating (7.36) for $\boldsymbol{J}_q \in \mathbb{R}^{n_q \times n_q}$ dense costs $\mathcal{O}(6n^2 r_A + 6nr_A^2 + 2r_A^2)$ operations, whereas for sparse $\boldsymbol{J}_q$ the scaling is (at best) reduced to $\mathcal{O}(6nr_A + 6nr_A^2 + 2r_A^2)$.

Remark 10. *Note that in the case were $\mathcal{A}$ is a rank-1 tensor the operation counts for the derivatives of f reduce to $\mathcal{O}(3nr_B^m + r_B^m)$ (for all n) and for g to $\mathcal{O}(6n^2 + 6n)$ and $\mathcal{O}(12n)$ in the dense and sparse case, respectively.*
If, on the other hand, one aims to calculate the derivative with respect to a rank-1 update, i.e. $\partial_{\mathcal{E}} f(\widetilde{\mathcal{A}} + \mathcal{E})$, with $\mathcal{E} = [\![\boldsymbol{e}^{(1)}, \boldsymbol{e}^{(2)}, \boldsymbol{e}^{(3)}]\!] \in C_{\boldsymbol{n},1}$, one can use the above formulas (7.35) and (7.36) with $\mathcal{A} := \widetilde{\mathcal{A}} + \mathcal{E} \in C_{\boldsymbol{n}, r_A}$ by only evaluating the last column. More precise, one gets

$$\partial_{\boldsymbol{e}^{(p)}} f(\mathcal{A}) = l \, \boldsymbol{D}^{(p)} (\bullet_{q \neq p} \boldsymbol{D}^{(q)^T} \boldsymbol{e}^{(q)}) \in \mathbb{R}^{n_p \times 1}, \tag{7.37}$$

and

$$\partial_{\boldsymbol{e}^{(p)}} g(\mathcal{A}) = \boldsymbol{J}_p \Big(\boldsymbol{U}^{(p)} (\bullet_{q \neq p} \, \boldsymbol{U}^{(q)^T} \, \boldsymbol{J}_q^T \, \boldsymbol{e}^{(q)}) \Big) + \boldsymbol{J}_p^T \Big(\boldsymbol{U}^{(p)} (\bullet_{q \neq p} \, \boldsymbol{U}^{(q)^T} \, \boldsymbol{J}_q \, \boldsymbol{e}^{(q)}) \Big) \in \mathbb{R}^{n_p \times 1}. \tag{7.38}$$

Hence, the costs are $\mathcal{O}(3nr_A^{l-1} r_B^m + r_A^{n-1} r_B^m)$ in the case of f and $\mathcal{O}(6n^2 + 6nr_A + 2r_A)$ and $\mathcal{O}(6n + 6nr_A + 2r_A)$ in the case of g with dense and sparse matrices $\boldsymbol{J}_q$, respectively.
Similar, in the general case $\mathcal{E} \in C_{\boldsymbol{n}, r_E}$ with $r_E \geq 1$ the derivatives with respect to $\mathcal{E}$ can be calculated by using the formulas (7.35) and (7.36) with $\mathcal{A} := \widetilde{\mathcal{A}} + \mathcal{E}$ by only evaluating the last r_E columns. The computational costs are those for the previous case (rank-1 update) times r_E.

Remark 11. *For the discretized stray field energy in (7.2) one can also use the functional f with $l = m = 1$ and $vec(\mathcal{B}) = (\boldsymbol{N}\boldsymbol{m})^{(p)}$ cf. Sec. 7.1:*
First note that the Kronecker matrix $\boldsymbol{N}$ is symmetric. Furthermore the stray field energy functional is (up to a factor) $-\frac{1}{2} \boldsymbol{m}^T \boldsymbol{N} \boldsymbol{m}$ where $\boldsymbol{m}$ comes from vectorizations of tensor formats for the magnetization components. Tensor formats are linear in their parameters (e.g. factor matrices in the case of CP tensors, or also the core tensor for Tuckers) and hence, the energy is a quadratic functional with respect to the parameters of the format. Thus, the first variation with respect to the k-th parameter in the p-th component can be calculated from $-\boldsymbol{\delta}^T \boldsymbol{N}\boldsymbol{m} = -\boldsymbol{\delta}^{(p)^T} (\boldsymbol{N}\boldsymbol{m})^{(p)}$, where $\boldsymbol{\delta}^{(p)}$ is the only non-zero component which arises from the p-th component of $\boldsymbol{m}$ by substituting the k-th parameter by a variation. E.g. for a canonical x

component $\mathcal{M}^{(1)} = [\![\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!]$ *the first variation with respect to the second factor is*

$$\begin{aligned}-\langle ten(\boldsymbol{\delta}^{(2)}), ten((\boldsymbol{N}\boldsymbol{m})^{(1)})\rangle = -\langle [\![\boldsymbol{U}^{(1)}, \boldsymbol{\delta}^{(2,2)}, \boldsymbol{U}^{(3)}]\!], ten((\boldsymbol{N}\boldsymbol{m})^{(1)})\rangle = \\ -\langle [\![\boldsymbol{U}^{(1)}, \boldsymbol{\delta}^{(2,2)}, \boldsymbol{U}^{(3)}]\!], [\![\boldsymbol{H}_x^{(1)}, \boldsymbol{H}_x^{(2)}, \boldsymbol{H}_x^{(3)}]\!]\rangle = -\langle \boldsymbol{\delta}^{(2,2)}, \boldsymbol{H}_x^{(2)}(\boldsymbol{H}_x^{(3)^T}\boldsymbol{U}^{(3)} \bullet \boldsymbol{H}_x^{(1)^T}\boldsymbol{U}^{(1)})\rangle_F,\end{aligned} \tag{7.39}$$

where $\mathcal{B} = [\![\boldsymbol{H}_x^{(1)}, \boldsymbol{H}_x^{(2)}, \boldsymbol{H}_x^{(3)}]\!]$ *is the canonical tensor computed by*

$$(\boldsymbol{N}\boldsymbol{m})^{(1)} = \sum_{q=1}^{3} \Big(\mathcal{P}^{(q)} \times_1 \boldsymbol{J}_n + (\mathcal{P}^{(1)} \times_q \boldsymbol{J}_n)^T\Big)\boldsymbol{m}^{(q)}, \tag{7.40}$$

cf. Sec. 7.1.

Remark 12. *Functionals involving Tucker tensors can be treated along the same lines as in this section. Matricization formulas for those cases can also be found in reference [49].*

7.4.3 Minimization with fixed CP rank

A penalty method is applied to the reformulated problem (7.23) for fixed prescribed CP ranks r_p, $p = 1, 2, 3$. The penalty function for the subproblems (fixed μ) is given as

$$Q_\mu(\mathcal{M}) := \Psi_t(\mathcal{M}) + \mu P_{\boldsymbol{n}}(\mathcal{M}), \tag{7.41}$$

cf. Sec. 7.1. The weights of the CP representations are assumed to be absorbed by the factor matrices, like in Sec. 7.4.2. In general, it is possible to perform the minimization with respect to a certain subset of parameters of the CP representation of $\mathcal{M}$. In the description of Alg. 12 the set of such 'free parameters' is denoted with $\boldsymbol{p}(\mathcal{M})$. For instance, if one decides to minimize a subproblem with respect to the last r'_p columns of $\mathcal{M}^{(p)}$, $p = 1, 2, 3$, this means solving the problem

$$\min_{\boldsymbol{p}(\mathcal{M})} Q_\mu(\mathcal{M}) = \min_{\mathcal{D}} Q_\mu(\mathcal{M}' + \mathcal{D}), \tag{7.42}$$

where the factors of $\mathcal{M}'$ consist of the first $r_p - r'_p$ columns of the factors of $\mathcal{M}^{(p)}$, $p = 1, 2, 3$ and $\mathcal{D} \in \times_{q=1}^{3} C_{\boldsymbol{n}, r'_q}$. In this case one has $\boldsymbol{p}(\mathcal{M}) = \rho_{CP}(\mathcal{D})$. In the numerical tests below the subproblems are solved with a quasi-Newton method; $\mu_0 = 10$ and $M_n = 1$ or 5. Tab. 7.1 shows results for a nearly uniform flower in the unit cube as initial magnetization and parameters $Q = 0.05$ (anisotropy) and $\lambda = 8.45$ (units of exchange length), cf. Sec. 2.2. Grid spacing is $h = 1/20$ and no external field is applied. The parameters are near the so-called *single domain limit*, also compare with the μMag standard problem #3 [12]. The initial magnetization is generated as CP approximation from the formulas (5.33) with $a = c = 5$ and $b = 2$. The projected gradients

Algorithm 12 `qPM` for CP-magnetization; `qPM_CP(`$\mathcal{M}^0$`,`$\boldsymbol{p}(\mathcal{M}^0)$`,tol,` μ_0`,`$\tau_n \to 0$`)`

```
Require: $\mathcal{M}^0 \in \times_{q=1}^{3} \mathcal{C}_{\boldsymbol{n},r_q}$, $\boldsymbol{p}(\mathcal{M}) \subseteq \rho_{CP}(\mathcal{M}^0)$, tol$> 0$, $\mu_0 > 0, \tau_n \to 0$ (nonnegative)
Ensure: $\mathcal{M} \in \times_{q=1}^{3} \mathcal{C}_{\boldsymbol{n},r_q}$
$\boldsymbol{p}(\mathcal{M}) \leftarrow \boldsymbol{p}(\mathcal{M}^0)$, $\mu \leftarrow \mu_0$, $n \leftarrow 0$
while $\left\|\nabla_{\boldsymbol{p}(\mathcal{M})} Q_\mu(\mathcal{M})\right\| >$ tol do
    $\boldsymbol{p} \leftarrow \boldsymbol{p}(\mathcal{M})$
    Find approximate solution $\boldsymbol{p}(\mathcal{X})$ to the sub-problem $\min_{\boldsymbol{p}} Q_\mu(.)$ starting at $\boldsymbol{p}$ and ensure $\left\|\nabla_{\boldsymbol{p}} Q_\mu(\mathcal{X})\right\| \le \tau_n$
    $\mu \leftarrow M_n \mu \; (M_n \ge 1)$
    $\mathcal{M} \leftarrow$ overwrite free parameters in $\mathcal{M}$ with $\boldsymbol{p}(\mathcal{X})$
    $n \leftarrow n + 1$
end while
```

Table 7.1: Minimization w.r.t. fixed CP format with Alg. 12 with $\mu_0 = 10$ and $M_n = 1$ or 5. For each line in the table 15 loops in Alg. 12 were performed. Results for a nearly uniform flower (5.33) $[a = c = 5, b = 2]$ in the unit cube as initial magnetization with different CP ranks. Parameters are $Q = 0.05$ (z-direction) and $\lambda = 8.45[l_{ex}]$. Grid spacing is $h = 1/200$ and no external field is applied. In the first column the ranks are shown, columns 2-5 show the energies, column 6 shows the average z-component of the magnetization, column 7 shows the constraint violation measured by the function P_n from (7.10), column 8 shows the 2- norm of the projected gradient and the last column shows the number of function evaluations (function and gradient evaluations are counted as one evaluation).

$r \equiv r_q$	e_{tot}	e_d	e_{ex}	e_{an}	$\langle m_z \rangle$	$P_{\boldsymbol{n}}(\mathcal{M})$	$\|\nabla_\Pi \Psi_t(\mathcal{M})\|$	#evaluations
1	0.1558	0.1518	0.0031	0.0009	0.9906	$1.12\,e{-}05$	$1.47\,e{-}04$	139
2	0.1525	0.1454	0.0055	0.0017	0.9828	$1.75\,e{-}06$	$1.34\,e{-}04$	176
3	0.1573	0.1541	0.0026	0.0006	0.9938	$6.81\,e{-}05$	$2.16\,e{-}04$	118
4	0.1513	0.1404	0.0082	0.0027	0.9721	$6.17\,e{-}06$	$9.63\,e{-}05$	152
6	0.1598	0.1587	0.0008	0.0003	0.9975	$4.84\,e{-}07$	$1.57\,e{-}04$	155

are calculated with approximate Tucker arithmetic with Alg. 6, meaning the CP magnetization is approximated in Tucker format (accuracy 1e-14) and all rank increasing operations for computing the projected gradient are truncated with an accuracy of 1e-09.

Convergence behavior for this example can be observed by the 'opening of the flower'; the value of the average z-component at the equilibrium is $\langle m_z \rangle = 0.9711$. One can clearly observe that higher ranks do *not* automatically lead to better convergence. On the contrary, the minimization with $r = 3$ or 6 almost fails completely, whereas $r = 2$ leads to good convergence. The objective functions corresponding to the minimization problems with different ranks have different degree of non-linearity. This and also the non-closedness of the format might be reasons why the behavior is not comparable or predictable. The attraction to 'artificial' local minima can not be excluded either. Furthermore, the prior determination of the ranks is not possible. In the next section a more sophisticated scheme is developed which is a combination of Alg. 12 and steepest descent.

7.5 Successive Rank-k Update

The approaches of section 7.3 and 7.4 are very different to each other. While minimization with truncated iterations generalizes 'ordinary' minimization schemes to structured tensors, the scheme in Alg. 13 is especially constructed for the CP format. Both approaches have their weak points. Truncated iterations need a stable and closed format, which allows application of rank - adaptive rounding. The Tucker format is an example of such a format. Point-wise operations like the Hadamard product, which occurs in the evaluation of the micromagnetic side constraint, are a serious stumbling block for any of the introduced formats due to the rapid increase of rank. The storage requirements for the dense core of the Tucker format limit the usage of this format in connection with rank increasing operations. In the approach with truncated operations the amount of rank increasing operations is very large. On the other hand, the second notion [fixing the format] complicates the optimization problem drastically because the objective gets more non-linear. The easy test summarized in Tab. 7.1 confirms this.

The following approach can be seen as a beneficial combination of the two approaches which were introduced in the previous sections of this chapter.

Imagine that for given magnetization a *descent direction* is calculated. A condition for such a direction in the case of an at least two times differentiable function $h : \mathbb{R}^N \to \mathbb{R}$ can be derived from Taylor expansion:

$$h(\boldsymbol{x} + s\boldsymbol{d}) - h(\boldsymbol{x}) = s\,\nabla h(\boldsymbol{x})^T \boldsymbol{d} + O(s^2), \tag{7.43}$$

and therefore $\boldsymbol{d}$ is a direction of descent at the position $\boldsymbol{x}$ iff $\nabla h(\boldsymbol{x})^T \boldsymbol{d} < 0$. Hence, the negative gradient at non stationary points is always a descent direction, even the steepest. For micromagnetic energy minimization this is the negative projected gradient $-\nabla_\Pi \Psi_t(\mathcal{M})$. As suggested in Sec. 7.4.3 the quantity $\nabla_\Pi \Psi_t(\mathcal{M})$ for canonical tensor magnetization $\mathcal{M}$ can be calculated by approximating $\mathcal{M}$ as Tucker tensor and performing truncated operations with e.g. Alg. 6. For moderate CP-ranks this can be performed very fast and efficient.

Of course, any other descent direction can be used, it does not have to be the steepest descent, namely the negative projected gradient. However, the following considerations are restricted to the steepest descent case.

In order to use this descent direction as an CP update with certain ranks an ALS- or optimization based CP approximation is performed, see remark 13 for more details. The resulting descent direction $\mathcal{D}$ is used in a backtracking loop to determine a step length s that ensures a decrease of the energy. Afterwards, Alg. 12 is used with the updated magnetization as initial guess and $\boldsymbol{p}(\mathcal{M}) = \rho_{CP}(s\mathcal{D})$. This method is summarized in Alg. 13. It consists of two steps:

1. First, a tensor structured rank-k descent direction is determined.
2. then, the new update is calculated and serves as initial guess for minimization with respect to the new free coordinates.

The ranks of the approximations increase linearly with the number of outer loops. Nevertheless, a restarted version could be considered. Namely, after a specific amount of iterations the current iterate is approximated with lower ranks and taken as initial configuration for a restart.

Moreover, in the description of Alg. 13 the initial penalty parameter μ_0 is the same for each outer iteration. Of course, one can increase it from one to the other outer iteration in order to accelerate convergence.

Tab. 7.2 is the equivalent of Tab. 7.1, where rank-1 updates were used in Alg. 13. Significant improvements can be recognized. Tab. 7.3 shows the results for the same example like in Tab. 7.2 but with much smaller grid spacing $1/200$. Note that the computation time for function evaluations scales with $O(1/h^2)$[4], which allows computations on large grids.

Tab. 7.4 and 7.5 show analogue tests for a vortex like initial configuration and rank-1 and rank-2 updates, respectively.

In order to include an external field into the tests, Tab. 7.6 shows the results for uniform initial magnetization and an applied field in the direction $(0.57, 0.57, 0.57)$. No anisotropy is assumed. One recognizes quite fast convergence.

Remark 13. *The best approximation* $\mathcal{D}^{(q)}$ *of* $-\left(\nabla_\Pi \Psi_t(\mathcal{M})\right)^{(q)}$ *by a rank-*r_q *CP tensor is only well-defined for the rank-*1 *case, cf. Sec.* 4.2*. However, the approximation only has to be an*

---

[4] For $h = 1/200$ the Matlab implementation that was used needs about one second for computing the energies, the penalty term and all the gradients; and about 2.5 seconds for $h = 1/300$.

Algorithm 13 Successive rank-k update method; $\texttt{qPM\_CP\_rk}(\mathcal{M}^0, \boldsymbol{r}, \texttt{tol}_{1,2},\ \mu_0, \tau_n \to 0)$

```
Require: $\mathcal{M}^0 \in \times_{q=1}^{3} \mathcal{C}_{\boldsymbol{n},r_q}$, $\boldsymbol{r} = (r_1, r_2, r_3)$, $\texttt{tol}_{1,2} > 0$, $\mu_0 > 0$, $\tau_n \to 0$ (nonnegative)
Ensure: $\mathcal{M} \in \times_{q=1}^{3} \mathcal{C}_{\boldsymbol{n},r_q}$
$\mathcal{M} \leftarrow \texttt{qPM\_CP}(\mathcal{M}^0, \rho_{CP}(\mathcal{M}^0), \texttt{tol}_1,\ \mu_0, \tau_n \to 0)$
while $\|\nabla_\Pi \Psi_t(\mathcal{M})\| > \texttt{tol}_2$ do
    $E \leftarrow \Psi_t(\mathcal{M})$
    $\mathcal{D}^{(q)} \leftarrow$ Rank-$r_q$ CP approximation of $-\left(\nabla_\Pi \Psi_t(\mathcal{M})\right)^{(q)}$, cf. Remark 13.
    $\widetilde{\mathcal{M}^{(q)}} \leftarrow \mathcal{M}^{(q)} + \mathcal{D}^{(q)},\ q = 1,2,3$
    $s \leftarrow 1$
    while $\Psi_t(\widetilde{\mathcal{M}}) > E$ do
        $s \leftarrow s/2$
        $\widetilde{\mathcal{M}}^{(q)} = \mathcal{M}^{(q)} + s\mathcal{D}^{(q)},\ q = 1,2,3$
    end while
    $\boldsymbol{p} \leftarrow \rho_{CP}(s\mathcal{D})$
    $\mathcal{M} \leftarrow \texttt{qPM\_CP}(\widetilde{\mathcal{M}}, \boldsymbol{p}, \texttt{tol}_1,\ \mu_0, \tau_n \to 0)$
end while
```

arbitrary descent direction and not the best approximation: In the situation here, the condition for descent translates to

$$\sum_q \langle \mathcal{D}^{(q)}, (\nabla_\Pi \Psi_t(\mathcal{M}))^{(q)} \rangle < 0, \tag{7.44}$$

where $\|\nabla_\Pi \Psi_t(\mathcal{M})\| = \sqrt{\sum_q \|(\nabla_\Pi \Psi_t(\mathcal{M}))^{(q)}\|^2} \neq 0$ *is assumed. Hence, if the ordinary rank-*r_q *approximations* $\mathcal{D}^{(q)}$ *do not yield a descent direction* $\mathcal{D}$*, one can force it by claiming* (7.44)*. By introducing a slack variable* $S > 0$ *such that*

$$C_S(\mathcal{D}) := S + \sum_q \langle \mathcal{D}^{(q)}, (\nabla_\Pi \Psi_t(\mathcal{M}))^{(q)} \rangle = 0, \tag{7.45}$$

one can treat the side constraint (7.44) *in a penalty framework with subproblems*

$$\min_{\mathcal{D}} \sum_q \|\mathcal{D}^{(q)} + (\nabla_\Pi \Psi_t(\mathcal{M}))^{(q)}\|^2 + \tfrac{\mu}{2} C_S(\mathcal{D})^2, \tag{7.46}$$

for increasing μ*. This is equivalent to*

$$\min_{\mathcal{D}} \sum_q \|\mathcal{D}^{(q)}\|^2 + (2 + \mu S) \sum_q \langle \mathcal{D}^{(q)}, (\nabla_\Pi \Psi_t(\mathcal{M}))^{(q)} \rangle + \tfrac{\mu}{2} \Big(\sum_q \langle \mathcal{D}^{(q)}, (\nabla_\Pi \Psi_t(\mathcal{M}))^{(q)} \rangle \|^2 \Big)^2. \tag{7.47}$$

Table 7.2: Minimization with rank-1 update with Alg. 13. Results for a nearly uniform flower (5.33) $[a = c = 5, b = 2]$ in the unit cube as initial magnetization with different CP ranks. Parameters are $Q = 0.05$ (z-direction) and $\lambda = 8.45[l_{ex}]$. Grid spacing is $h = 1/20$ and no external field is applied. The first column gives the iteration number in the outer loop and the corresponding ranks in the brackets. In each outer iteration 8 iterations in `qPM_CP` were performed, while no other stopping criterion was used. The subproblems in Alg. 12 are solved with a quasi-Newton method; $\mu_0 = 10$ and $M_n = 1$ or 5. In total 350 function evaluations (function and gradient evaluations are counted as one evaluation) were performed. Columns 2-5 show the energies, column 6 shows the average z-component of the magnetization, column 7 shows the constraint violation measured by the function P_n from (7.10), column 8 shows the 2- norm of the projected gradient.

iteration ($r \equiv r_q$)	e_{tot}	e_d	e_{ex}	e_{an}	$\langle m_z \rangle$	$P_{\boldsymbol{n}}(\mathcal{M})$	$\|\nabla_\Pi \Psi_t(\mathcal{M})\|$
1 (3)	0.1624	0.1618	0.0001	0.0006	0.9944	1.42 e−04	1.72 e−04
2 (4)	0.1576	0.1553	0.0018	0.0005	0.9947	6.65 e−06	1.62 e−04
3 (5)	0.1520	0.1418	0.0077	0.0025	0.9747	4.23 e−06	9.86 e−05
4 (6)	0.1519	0.1417	0.0077	0.0025	0.9746	3.85 e−06	9.13 e−05
5 (7)	0.1519	0.1417	0.0077	0.0025	0.9745	1.68 e−06	8.85 e−05
6 (8)	0.1514	0.1399	0.0087	0.0028	0.9711	6.38 e−06	8.01 e−05

The optimal solution fulfills

$$\sum_q \langle \mathcal{D}^{(q)}, (\nabla_\Pi \Psi_t(\mathcal{M}))^{(q)} \rangle = -\frac{1 + S^{\frac{\mu}{2}}}{1 + \|\nabla_\Pi \Psi_t(\mathcal{M})\|^{2\,\frac{\mu}{2}}} \|\nabla_\Pi \Psi_t(\mathcal{M})\|^2 < 0. \tag{7.48}$$

Furthermore, note that if $\nabla_\Pi \Psi_t(\mathcal{M}))^{(q)}$ *is a Tucker approximation (as suggested in Sec. 7.4.3), the inner product with the canonical tensors* $\mathcal{D}^{(q)}$ *can be carried out efficiently:*
Let $\mathcal{X} = [\![C; \boldsymbol{V}^{(1)}, \boldsymbol{V}^{(2)}, \boldsymbol{V}^{(3)}]\!] \in \mathcal{T}_{\boldsymbol{n},\boldsymbol{r}}$ *and* $\mathcal{Y} = [\![\lambda; \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}]\!] \in \mathcal{C}_{\boldsymbol{n},R}$*, then there holds (cf. Ch. 4)*

$$\langle \mathcal{X}, \mathcal{Y} \rangle = vec(C)^T (\boldsymbol{V}^{(3)^T} \boldsymbol{U}^{(3)} \odot \boldsymbol{V}^{(2)^T} \boldsymbol{U}^{(2)} \odot \boldsymbol{V}^{(1)^T} \boldsymbol{U}^{(1)})\, \lambda, \tag{7.49}$$

which needs $\mathcal{O}(R \sum_q r_q n_q + R \prod_q r_q + \prod_q r_q)$ *operations.* □

Table 7.3: Minimization with rank-1 update with Alg. 13. The same configurations as in Tab. 7.2 but with grid spacing $h = 1/200$.

iteration ($r \equiv r_q$)	e_{tot}	e_d	e_{ex}	e_{an}	$\langle m_z \rangle$	$P_{\boldsymbol{n}}(\mathcal{M})$	$\lVert\nabla_\Pi \Psi_t(\mathcal{M})\rVert$
1 (3)	0.1613	0.1604	0.0003	0.0006	0.9939	$1.97\,e{-}04$	$6.25\,e{-}02$
2 (4)	0.1602	0.1572	0.0026	0.0004	0.9960	$1.07\,e{-}05$	$1.36\,e{-}03$
3 (5)	0.1551	0.1508	0.0033	0.0009	0.9807	$2.22\,e{-}05$	$2.68\,e{-}03$
4 (6)	0.1530	0.1463	0.0052	0.0015	0.9843	$7.72\,e{-}06$	$1.19\,e{-}04$
5 (7)	0.1521	0.1442	0.0060	0.0019	0.9807	$5.82\,e{-}06$	$9.49\,e{-}05$
6 (8)	0.1519	0.1421	0.0075	0.0023	0.9746	$4.25\,e{-}06$	$8.36\,e{-}06$

Table 7.4: Minimization with rank-1 update with Alg. 13. Results for the vortex configuration (5.32) [$r_c = 0.25$] in the unit cube as initial magnetization with different CP ranks. Parameters are $Q = 0.05$ (z-direction) and $\lambda = 8.45[l_{ex}]$. Grid spacing is $h = 1/20$ and no external field is applied. The first column gives the iteration number in the outer loop and the corresponding ranks in the brackets. In each outer iteration 8 iterations in `qPM_CP` were performed, while no other stopping criterion was used. In total 519 function evaluations (function and gradient evaluations are counted as one evaluation) were performed. Columns 2-5 show the energies, column 6 shows the average y-component of the magnetization, column 7 shows the constraint violation measured by the function P_n from (7.10), column 8 shows the 2- norm of the projected gradient.

iteration ($r \equiv r_q$)	e_{tot}	e_d	e_{ex}	e_{an}	$\langle m_y \rangle$	$P_{\boldsymbol{n}}(\mathcal{M})$	$\lVert\nabla_\Pi \Psi_t(\mathcal{M})\rVert$
1 (2)	0.1610	0.0251	0.1097	0.0262	0.0978	$4.32\,e{-}04$	$2.26\,e{-}04$
2 (3)	0.1606	0.0312	0.1031	0.0262	0.2035	$8.97\,e{-}06$	$2.67\,e{-}04$
3 (4)	0.1585	0.0316	0.1025	0.0244	0.2034	$4.94\,e{-}06$	$2.13\,e{-}04$
4 (5)	0.1570	0.0341	0.0978	0.0250	0.2256	$3.86\,e{-}06$	$1.91\,e{-}04$
5 (6)	0.1569	0.0343	0.0974	0.0251	0.2306	$2.25\,e{-}06$	$1.87\,e{-}04$
6 (7)	0.1553	0.0322	0.0980	0.0251	0.2271	$3.76\,e{-}06$	$1.80\,e{-}04$

Table 7.5: Minimization with rank-2 update with Alg. 13. Results for the vortex configuration (5.32) [$r_c = 0.25$] in the unit cube as initial magnetization with different CP ranks. Parameters are $Q = 0.05$ (z-direction) and $\lambda = 8.45[l_{ex}]$. Grid spacing is $h = 1/20$ and no external field is applied. The first column gives the iteration number in the outer loop and the corresponding ranks in the brackets. In each outer iteration 8 iterations in `qPM_CP` were performed, while no other stopping criterion was used. In total 309 function evaluations (function and gradient evaluations are counted as one evaluation) were performed. Columns 2-5 show the energies, column 6 shows the average y-component of the magnetization, column 7 shows the constraint violation measured by the function P_n from (7.10), column 8 shows the 2- norm of the projected gradient.

iteration ($r \equiv r_q$)	e_{tot}	e_d	e_{ex}	e_{an}	$\langle m_y \rangle$	$P_{\boldsymbol{n}}(\mathcal{M})$	$\|\nabla_\Pi \Psi_t(\mathcal{M})\|$
1 (2)	0.1610	0.0251	0.1097	0.0262	0.0978	4.32 e−04	2.26 e−04
2 (4)	0.1573	0.0389	0.0908	0.0275	0.2897	2.10 e−06	2.29 e−04
3 (6)	0.1543	0.0352	0.0916	0.0275	0.2853	4.73 e−06	2.09 e−04
4 (8)	0.1539	0.0364	0.0897	0.0278	0.2968	2.61 e−06	1.77 e−04

Table 7.6: Minimization with rank-1 update with Alg. 13 with external field in direction $(1, 1, 1)$ with strength $\|\boldsymbol{h}_{ext}\|_2 = 1$. Results for uniform magnetization in z-direction of the unit cube as initial magnetization with different CP ranks. Parameters are $Q = 0$ and $\lambda = 11.30[l_{ex}]$. Grid spacing is $h = 1/40$ and no anisotropy field is assumed. The first column gives the iteration number in the outer loop and the corresponding ranks in the brackets. In each outer iteration 15 iterations in `qPM_CP` were performed. The final stopping criterion was a projected gradient norm of less than 1e-5. In total 301 function evaluations (function and gradient evaluations are counted as one evaluation) were performed. Columns 2-5 show the energies, column 6-8 shows the average x, y, z-component of the magnetization, column 9 shows the constraint violation measured by the function P_n from (7.10), column 10 shows the 2- norm of the projected gradient.

iter	e_{tot}	e_d	e_{ex}	e_{ext}	$\langle m_x \rangle$	$\langle m_y \rangle$	$\langle m_z \rangle$	$P_{\boldsymbol{n}}(\mathcal{M})$	$\|\nabla_\Pi \Psi_t(\mathcal{M})\|$
1 (2)	-0.7628	0.1691	0.0009	-0.9328	0.3666	0.3788	0.8703	7.22 e−03	2.16 e−01
2 (3)	-0.8352	0.1638	0.0003	-0.9993	0.5774	0.5761	0.5774	8.27 e−08	2.49 e−04
3 (4)	-0.8352	0.1638	0.0003	-0.9993	0.5774	0.5761	0.5774	8.34 e−08	9.36 e−06

7.6 Summary and Conclusions

The micromagnetic energy minimization problem was investigated with the additional constraint of structured tensor magnetization components on a Cartesian grid. For that purpose, two distinct approaches were described:

1. Truncated iterations [Exterior approach],
2. Minimization within the representation [Interior approach].

Both methods have their weak points. Truncated iterations need a stable and closed format, which allows application of rank - adaptive rounding. However, the storage requirements in connection with rank increasing operations limit the usage of such formats. Especially, element-wise operations on structured tensors are a serious stumbling block. In the exterior approach the amount of rank increasing operations is very large. On the other hand, the second concept [fixed format] complicates the optimization problem drastically, because the objective function gets more non-linear. It turns out that a combination of both approaches yields a promising scheme, cf. Alg. 13:

3. Successive rank-k update scheme [Combined approach].

This method consists of an 'exterior step', namely calculating the rank-k update, and an 'interior step', minimization with respect to the new free coordinates. The ranks of the approximations increase linearly with the number of outer loops. Nevertheless, a restarted version could be considered: After a specific amount of iterations the current iterate is approximated with lower ranks and taken as initial configuration for a restart.

There is still the question left whether the choice of descent direction can significantly improve the method. Instead of the projected gradient, one could use the gradient of the penalty function or variations of it. This directions might be cheaper to determine, since it is a CP tensor output of the previous iteration.

In any case, larger grids can be used due to the sublinear scaling in the volume size.[5] Hence, the discrete representation of the energy is more accurate. Also, large particles demand for larger grids due to the required resolution in the order of the exchange length or domain wall width.

---

[5] The evaluation of the objective function and the gradient scales with $O(1/h^2)$, where h is the grid spacing. This is sublinear, since the number of total grid points/computational cells is $O(1/h^3)$.

Chapter 8

Non-Uniform FFT for the Finite Element Computation of the Scalar Potential

Large portions of this chapter were previously submitted for publication as [15] and have been reproduced here with permission of the co-authors. Content which was not generated by the author of this thesis is explicitly denoted.

8.1 Introduction and Motivation

From the previous chapters the question is still left whether it is possible to successfully apply tensor grid techniques to unstructured meshes, e.g. tetrahedral meshes from general geometries Ω within a finite element (FE) approach.

If the data are not sampled on a structured grid beforehand, the first task would be to get them on a tensor grid by some *gridding procedure*. The cost for this step has to be at least linear in the data-size, since each location (e.g. node in the case of data from a FE-mesh) has to be considered at least once. A micromagnetic algorithm (e.g. for computing the stray field), which works with data from unstructured grids and tries to use ideas form tensor methods for function related (structured) tensors, relies on such gridding in each step of the iterative procedure. Hence, it will scale at best linear in time.

Nevertheless, the idea of combining finite element computation with methods for structured grids can be worth being considered. For instance in stray field calculations, one could benefit from fast Fourier transform acceleration. The combination of *gridding* and *fast Fourier transform* (FFT) dates back to the beginning of the so-called *non-uniform fast Fourier transform* (NFFT) [80, 81]:

For the sake of brief discussion, assume arbitrarily located data points $\boldsymbol{y}_j \in (-\frac{1}{2}, \frac{1}{2})^3,\ j = 1 \dots M$ and define for $\boldsymbol{n} = (n_1, n_2, n_3)$: $I_{\boldsymbol{n}} := \{\boldsymbol{k} \in \mathbb{Z}^3 \mid -\boldsymbol{n}/2 \leq_c \boldsymbol{k} \leq_c \boldsymbol{n}/2 - 1\}$.

The sums

$$\begin{aligned} \widehat{f}_{\boldsymbol{k}} &= \sum_{j=1}^{M} f_j \, e^{-2\pi \mathrm{i} \langle \boldsymbol{k}, \boldsymbol{y}_j \rangle}, \; f_j \in \mathbb{C} \quad \text{(NDFT transposed)}, \\ g_j &= \sum_{\boldsymbol{k} \in I_{\boldsymbol{n}}} \widehat{f}_{\boldsymbol{k}} \, e^{2\pi \mathrm{i} \langle \boldsymbol{k}, \boldsymbol{y}_j \rangle}, \; \widehat{f}_{\boldsymbol{k}} \in \mathbb{C} \quad \text{(NDFT)}, \end{aligned} \tag{8.1}$$

are analogues of *the discrete Fourier transform* (DFT), so-called *non-uniform discrete Fourier transforms* (NDFT). Observe, however, that a big difference to ordinary *discrete Fourier transform* makes the fact that these sums are not *inverse or unitary transformations* to each other in general. An exception is the case where the data $\boldsymbol{y}_j$ are equispaced on a tensor grid with $M = \boldsymbol{n}$, which corresponds to discrete Fourier transform itself. Likewise, the above operations (8.1) generally have nothing to do with *acquisition of data* in the Fourier domain [82]. Fast methods have been invented, which break down the quadratic computational complexity to $O(\prod_{q=1}^{3} n_q \log \prod_{q=1}^{3} n_q + M)$ in the case of *non-uniform fast Fourier transform* (NFFT). The basic idea is to smear the data over a tensor grid by a gridding process and then apply FFT. The data-smearing is undone afterwards. In the case of convolution with e.g. a *Gaussian heat kernel* as gridding procedure, this step is a simple scalar division (*deconvolution*) in Fourier space. For a tutorial the reader is referred to [83].

Remarkably enough, the convolution theorem for discrete convolution of equispaced data with radial kernels generalizes to non-equispaced data [84]. Namely, consider a discrete convolution of non-equispaced data, i.e.

$$\phi(\boldsymbol{x}_i) = \sum_{j=1}^{M} \alpha_j \mathcal{K}(\boldsymbol{x}_i - \boldsymbol{y}_j), \quad \alpha_j \in \mathbb{R}, \, \boldsymbol{x}_i, \boldsymbol{y}_j \in (-\tfrac{1}{4}, \tfrac{1}{4})^3, \; j = 1 \dots M, \; i = 1 \dots N, \tag{8.2}$$

where $\mathcal{K}$ is some radial kernel function like, for example, the *Newtonian kernel* $\mathcal{N}(\boldsymbol{x}) := 1/\|\boldsymbol{x}\|$. Approximation of a smoothed version of the kernel by a Fourier series leads to a computational scheme similar to that derived from the *convolution theorem* for equispaced data (convolution is computed by the inverse Fourier transform of the product of the Fourier transforms of the data α_j and the kernel function), where (inverse) Fourier transform is replaced by the (adjoint) discrete Fourier transform, also compare with Eqn. (8.21). The application of this idea for the quadrature approximation of the integral representation of the micromagnetic scalar potential (compare with Eqn. (2.23)) in two dimensions was reported in the reference [28].

In the following sections a first order polynomial (P1) finite element method is developed that solves the set of partial differential equations (2.22) for the scalar potential and, which can be considered as a significant improvement of the method from [28]. This is mainly due to the fact that all numerical integration is done in a precomputation phase. Thus, all occurring integrals can be computed numerically up to machine precision without increasing computational costs, which makes the method faster and more accurate than the algorithm described in [28]. More precise, a splitting approach from [32] is used to derive two tasks:

- The solution of a *Dirichlet problem*, and
- The evaluation of the *single layer potential*.

The Dirichlet problem is solved by a usual Galerkin finite element ansatz. This can be done in linear time, provided preconditioning of the resulting linear system or a direct solving strategy (LU decomposition in the precomputation phase) is considered. The crucial part is the evaluation of the single layer potential which is a quadratically scaling task, if done in a naive way. Approximation of a smoothed version of the Newtonian kernel $N(x) = 1/|x|$ by a Fourier series leads to a computational scheme which is similar to the convolution theorem used e.g. in integral methods [77], also compare with section 8.4. Based on FFT for non-equispaced data (non-uniform FFT, NFFT) [80, 81, 83] and linearly scaling near-field correction, the single layer potential can be computed efficiently. Subsequently, this solution is combined with the finite element solution of the Dirichlet problem, yielding, in total, a complexity of $\mathcal{O}(M + N + (\prod_{q=1}^{3} n_q)(\log \prod_{q=1}^{3} n_q))$ for N nodes, M surface triangles and n_q being the number of discretization points of a tensor grid in the $q-$th direction.

In section 8.7 a closer analysis of the error of the scheme gives rise to the scaling of $\mathcal{O}(M + N + N \log N)$ in the case of roughly uniform meshes.

In the last section of this chapter, the idea is briefly discussed of exploiting the structured tensor representation of the smoothed kernel and to apply tensor compression on certain parts of the convolution formula (8.21). In principle, in addition to the reduction of storage, this gives one the opportunity to reduce the costs for the FFT-part, compare with section 8.9.

8.2 Ansatz of García-Cervera and Roma

In the subsequent text $H^1(\Omega)$ denotes the usual *Sobolev space*, i.e.

$$H^1(\Omega) := \{u \in L^2(\Omega) \mid \text{weak derivatives } \partial_q u \in L^2(\Omega),\ q = x, y, z\}, \tag{8.3}$$

and

$$H^1_{loc}(\overline{\Omega}^c) := \{u \in H^1(C) \mid C \subset \overline{\Omega}^c \text{ compact}\}. \tag{8.4}$$

Consider the splitting of the potential into $\phi = \phi_1 + \phi_2$. Comparing with Eqn. (2.22) results for $\phi_1 = (\phi_1^{int}, \phi_1^{ext}) \in H^1(\Omega) \times H^1_{loc}(\overline{\Omega}^c)$ in

$$\begin{aligned} -\Delta\phi_1^{int} &= -\nabla \cdot \boldsymbol{m} \qquad \text{in } \Omega, \\ \phi_1^{int} &= 0 \qquad\quad\;\; \text{on } \partial\Omega, \end{aligned} \tag{8.5}$$

where one sets $\phi_1^{ext} = 0$ in $\overline{\Omega}^c$. Hence, there holds $\left[\frac{\partial\phi_1}{\partial \boldsymbol{n}}\right] = -\frac{\partial\phi_1^{int}}{\partial \boldsymbol{n}}$.
The second part $\phi_2 = (\phi_2^{int}, \phi_2^{ext}) \in H^1(\Omega) \times H^1_{loc}(\overline{\Omega}^c)$ consequently fulfills

$$\begin{aligned} -\Delta\phi_2^{int} &= 0 && \text{in } \Omega, \\ -\Delta\phi_2^{ext} &= 0 && \text{in } \overline{\Omega}^c, \\ [\phi_2] &= 0 && \text{on } \partial\Omega, \\ \left[\frac{\partial\phi_2}{\partial \boldsymbol{n}}\right] &= -\boldsymbol{m} \cdot \boldsymbol{n} + \frac{\partial\phi_1^{int}}{\partial \boldsymbol{n}} && \text{on } \partial\Omega, \\ \phi_2^{ext}(\boldsymbol{x}) &= O(\tfrac{1}{\|\boldsymbol{x}\|}) \quad \text{as } \|\boldsymbol{x}\| \to \infty, \end{aligned} \tag{8.6}$$

with solution given by the *single layer potential*

$$\phi_2(\boldsymbol{x}) = \int_{\partial\Omega} g(\boldsymbol{y})\, \mathcal{N}(\boldsymbol{x} - \boldsymbol{y})\, \mathrm{d}\sigma(\boldsymbol{y}), \tag{8.7}$$

with the *Newtonian potential* $\mathcal{N}(\boldsymbol{x}) = \frac{1}{4\pi\|\boldsymbol{x}\|}$ and $g(\boldsymbol{y}) = \boldsymbol{m} \cdot \boldsymbol{n} - \frac{\partial\phi_1^{int}}{\partial \boldsymbol{n}}$.
The advantage of this ansatz is twofold. First, Eqn. (8.5) is a *Poisson equation* with *Dirichlet data* and, therefore, its *Galerkin system* after FE discretization is symmetric, positive definite and sparse, and only has to be solved for *free nodes*, i.e. *non-boundary nodes*, see Sec.8.3.
Secondly, as pointed out in [32], the single layer potential in Eqn. (8.7) is continuous towards the boundary and less singular than the *double layer potential* which arises in the *ansatz of Fredkin-Koehler* [30] and hence can be handled numerically more easily, also see Sec. 8.5.4.

The potential (8.7) might be evaluated at boundary nodes, providing the Dirichlet data for the Laplace equation in (8.6). Thus, an approximation of the solution ϕ_2^{int} to (8.6) could be determined by evaluation of the single layer potential at boundary nodes and subsequently solving a Dirichlet problem $-\Delta\phi_2^{int} = 0$. In this connection, direct evaluation of the single layer potential at boundary nodes in a naive way scales quadratically in the number of boundary nodes.
However, the intention of the forthcoming sections is to evaluate the single layer potential (8.7) on all nodes of a tetrahedral finite element (FE) mesh within a $P1$ finite element method. A *non-uniform Fourier* approach is used, which yields the complexity $O(M + N)$, i.e. linear in the number of boundary elements and nodes of the mesh, respectively.

Without any restrictions, the same idea of the fast evaluation scheme could also be applied for the above mentioned calculation of the Dirichlet data for (8.6), followed by solving the arising Dirichlet Galerkin system to obtain an approximation of ϕ_2 at the free nodes.
Furthermore, the author wants to stress that this approach could also be adapted for the ansatz of Fredkin and Koehler, which, however, will not be further discussed in this thesis.

8.3 FEM for the Dirichlet problem

For the sake of completeness, the finite element method for Dirichlet problems like (8.5) is briefly described here.
Let $H_0^1(\Omega)$ denote the *Sobolev space with zero boundary condition*.

The variational formulation of (8.5) reads:
Find the potential ϕ_1 *in the Sobolev space with zero-boundary conditions, i.e.* $\phi_1 \in H_0^1(\Omega)$*, such that*

$$\int_\Omega \nabla\phi_1 \cdot \nabla v = \int_\Omega \boldsymbol{m} \cdot \nabla v, \tag{8.8}$$

for all $v \in H_0^1(\Omega)$.

Eqn. (8.8) is now discretized on a tetrahedral mesh $\mathcal{T}$ with elements T_j, $j = 1 \dots M$ (here M denotes the number of tet-elements) and nodes $\boldsymbol{x}_i$, $i = 1 \dots N$, where affine basis functions $\varphi_\alpha^{(T_j)}$, $\alpha = 1 \dots 4$ in each tetrahedron are used. The usual assembly process by local stiffness matrices and load vectors leads to a linear system of size $N \times N$, i.e. $\boldsymbol{Sx} = \boldsymbol{b}$. From the local 4×4 stiffness matrix $\boldsymbol{S}_{\text{loc}}^{(T_j)}$ corresponding to the element T_j, i.e.

$$\boldsymbol{S}_{\text{loc}}^{(T_j)}(\alpha,\beta) = |T_j|\, \nabla\varphi_\alpha^{(T_j)}\, \nabla\varphi_\beta^{(T_j)}, \tag{8.9}$$

the stiffness matrix is computed by accumulation in a loop over all tetrahedral elements, i.e.

$$\boldsymbol{S}(\boldsymbol{k},\boldsymbol{k}) \mathrel{+}= \boldsymbol{S}_{\text{loc}}^{(T_j)}, \tag{8.10}$$

where $\boldsymbol{k}$ are the global indices of the nodes of element T_j.
The *stiffness matrix* $\boldsymbol{S}$ then has the entries

$$a_{km} = \sum_{j=1}^{M} \int_{T_j} \nabla\eta_m \cdot \nabla\eta_k, \tag{8.11}$$

where η_k, $k = 1 \dots N$ is the *nodal basis* (also often called *'the hat functions'*) of the space of

$\mathcal{T}$-piecewise affine, globally continuous functions (a N-dimensional subspace of the Sobolev space $H^1(\Omega)$). In a similar ('local to global') way, the load vector has the entries

$$b_k = \sum_{j=1}^{M} \int_{T_j} \boldsymbol{m} \cdot \nabla \eta_k, \tag{8.12}$$

where $\boldsymbol{m}$ itself is assumed to be a $\mathcal{T}$-piecewise affine nodal interpolation. It can be implemented by a matrix-vector product, where the matrix corresponds to the divergence operator, see [72] for more details.

Note that, due to the known values of the solution at the boundary nodes (in the case of ϕ_1 this values are already equal zero) every Dirichlet system can always be rewritten to a system with homogeneous boundary conditions. Since the nodal basis functions corresponding to free nodes (non-boundary nodes) form a basis of the space of $\mathcal{T}$-piecewise affine, globally continuous functions that are zero at the boundary (a finite dimensional subspace of the Sobolev space $H_0^1(\Omega)$), one, hence, only has to solve a *subsystem*, i.e.

$$\boldsymbol{S}(fn, fn)\boldsymbol{x}(fn) = \boldsymbol{b}(fn) - (\boldsymbol{S}\boldsymbol{x}_{bn})(fn) =: \widetilde{\boldsymbol{b}}(fn), \tag{8.13}$$

where *fn* and *bn* denote the N_f and N_b indices of *free nodes* and *boundary nodes*, respectively. The vector $\boldsymbol{x}_{bn}$ is understood as the vector of Dirichlet data (in the case here discussed equal to zero, thus $\widetilde{\boldsymbol{b}}(fn) = \boldsymbol{b}(fn)$) extended to length N by zero-padding for indices of free nodes.

For an easily readable Matlab implementation in the 2-dimensional case the reader is referred to the work [85].

The resulting system is reduced to the size $N_f \times N_f$ and is symmetric, positive definite and sparse. The solution gives the weights of the nodal basis functions at free nodes. In the numerical tests it is solved by using an ILU-preconditioned CG method, but *algebraic multigrid preconditioned CG* or (onetime) *LU decomposition* with backward substitution and exploiting the sparsity could be used, which makes the complexity for (8.5) linear in N_f.

8.4 Single-Layer Potential

While ϕ_1 is determined in linear time by an ordinary FEM for Dirichlet problems, the direct evaluation of the single-layer potential, i.e.

$$\phi_2(\boldsymbol{x}) = \int_{\partial\Omega} g(\boldsymbol{y})\, \mathcal{N}(\boldsymbol{x} - \boldsymbol{y})\, \mathrm{d}\sigma(\boldsymbol{y}), \tag{8.14}$$

at boundary nodes or all nodes of a FE mesh would cost $O(N_b^2)$ or $O(N_b N)$ respectively, where N_b is the number of nodes on the boundary and N the total number of nodes in the discretized domain Ω.

In the following an efficient evaluation technique of (8.7) based on Fourier approximation of the Newton kernel on an auxiliary tensor grid will be introduced.

Before going into detail, the main idea is briefly stated.

Note that in our $P1$ FE ansatz the first term of the function $g = \boldsymbol{m} \cdot \boldsymbol{n} - \frac{\partial \phi_1^{int}}{\partial \boldsymbol{n}}$ is piecewise affine, where the second one is constant for each surface triangle. For $\boldsymbol{m} \cdot \boldsymbol{n}$ the L^2-orthogonal projection onto the space of element-wise constant functions is applied by taking the integral mean over surface triangles, i.e. $1/|S_j| \int_{S_j} \boldsymbol{m} \cdot \boldsymbol{n}$. Thus Eqn. (8.14) in its discretized form reads

$$\phi_2(\boldsymbol{x}_i) \approx \sum_{j=1}^{M} g_j \int_{S_j} \mathcal{N}(\boldsymbol{x}_i - \boldsymbol{y})\, \mathrm{d}\sigma(\boldsymbol{y}), \quad i = 1 \ldots N, \tag{8.15}$$

where the S_j denote the M surface triangles.

Following the idea in [84], the kernel $N(x) := \mathcal{N}(\|\boldsymbol{x}\|) = 1/x,\ x := \|\boldsymbol{x}\|$ is split in a *singular* and *smooth part* respectively, i.e.

$$N(x) = \big(\underbrace{N(x) - N_s(x)}_{=:N_{\mathrm{NF}}}\big) + N_s(x), \tag{8.16}$$

where $N_s(.)$ is some approximation of $N(.)$ on an interval $[\epsilon, \beta]$, $\beta > \epsilon > 0$ (see Sec. 8.5.3), which is defined on the whole real axis and entirely smooth. $N_{NF}(.)$, on the other hand, is a *'near field' correction*. The corresponding multivariate functions is denoted by $\mathcal{N}_s(.) := N_s(\|.\|)$ and $\mathcal{N}_{NF}(.) := N_{NF}(\|.\|)$, respectively.

The approximation scheme (8.15) gets the form

$$\phi_2(\boldsymbol{x}_i) \approx \sum_{j=1}^{M} g_j \int_{S_j} \mathcal{N}_{\mathrm{NF}}(\boldsymbol{x}_i - \boldsymbol{y})\, \mathrm{d}\sigma(\boldsymbol{y}) + \sum_{j=1}^{M} g_j \int_{S_j} \mathcal{N}_s(\boldsymbol{x}_i - \boldsymbol{y})\, \mathrm{d}\sigma(\boldsymbol{y}) =: \phi_2^{\mathrm{NF}}(\boldsymbol{x}_i) + \phi_2^s(\boldsymbol{x}_i). \tag{8.17}$$

Note that the near field part ϕ_2^{NF} only has to be computed for elements that have less or equal distance than ϵ to the target point $\boldsymbol{x}_i$, i.e. $\mathcal{N}_{\mathrm{NF}}$ has small support. For the (weakly) singular cases, i.e. $\boldsymbol{x}_i \in S_j$, a simple integral transformation is used, see Sec. 8.5.4.

The fast computation of the part ϕ_2^s is achieved by approximation of the smooth kernel $\mathcal{N}_s$ by a Fourier series:

For the sake of simpler notation, assume a scaled domain, i.e. $\Omega \subset (-1/4, 1/4)^3$, such that the arguments of $\mathcal{N}$ lie in $\mathbb{T} := \{\boldsymbol{x} \in \mathbb{R}^3 \mid -1/2 \leq_c \boldsymbol{x} <_c 1/2\}$.

The smooth kernel $\mathcal{N}_s$ is approximated by its *Fourier series* on $\mathbb{T}$, where $\leq_c$ means component-wise $\leq$, i.e.

$$\mathcal{N}_s(\boldsymbol{x}) \approx \mathcal{F}\mathcal{N}_s := \sum_{\boldsymbol{l}\in I_{\boldsymbol{n}}} c_{\boldsymbol{l}}(\mathcal{N}_s)\, e^{2\pi \mathrm{i}\mathrm{x}\cdot \boldsymbol{l}}, \tag{8.18}$$

where $I_{\boldsymbol{n}} := \{\boldsymbol{l} \in \mathbb{Z}^3 \mid -\boldsymbol{n}/2 \leq_c \boldsymbol{l} \leq_c \boldsymbol{n}/2 - 1\}$ and the *Fourier coefficients*

$$c_{\boldsymbol{l}}(\mathcal{N}_s) = \int_{\mathbb{T}} \mathcal{N}_s(\boldsymbol{x})\, e^{-2\pi \mathrm{i}\mathrm{x}\cdot \boldsymbol{l}}\, \mathrm{d}\boldsymbol{x}. \tag{8.19}$$

Inserting (8.18) into ϕ_2^s in (8.17) and exchanging summation order yields

$$\phi_2^s(\boldsymbol{x}_i) = \sum_{\boldsymbol{l}\in I_{\boldsymbol{n}}} c_{\boldsymbol{l}}(\mathcal{N}_s) \underbrace{\Big(\sum_{j=1}^{M} g_j \int_{S_j} e^{-2\pi \mathrm{i}\mathrm{y}\cdot \boldsymbol{l}}\, \mathrm{d}\sigma(\boldsymbol{y})\Big)}_{=:b_{\boldsymbol{l}}} e^{2\pi \mathrm{i}\mathrm{x}_i\cdot \boldsymbol{l}} = \sum_{\boldsymbol{l}\in I_{\boldsymbol{n}}} \underbrace{d_{\boldsymbol{l}}}_{:=c_{\boldsymbol{l}}(\mathcal{N}_s)\, b_{\boldsymbol{l}}} e^{2\pi \mathrm{i}\mathrm{x}_i\cdot \boldsymbol{l}}. \tag{8.20}$$

The latter sum is a *non-uniform discrete Fourier transform* (NDFT), which can be computed efficiently using FFT in $O(|I_{\boldsymbol{n}}| \log |I_{\boldsymbol{n}}| + N)$ operations by so-called *non-uniform fast Fourier transform* (NFFT), [27].
The efficient computation of the tensor $\mathcal{B} = (\boldsymbol{b}_{\boldsymbol{l}})_{\boldsymbol{l}\in I_{\boldsymbol{n}}}$ will be discussed in the next section.

Overall, the approximation scheme for ϕ_2^s has a similar form as the well known *convolution theorem* for equispaced data, i.e.

$$\phi_2^s = \mathrm{NFFT}\Big(\big(c_{\boldsymbol{l}}(\mathcal{N}_s)\big)_{\boldsymbol{l}\in I_{\boldsymbol{n}}} \bullet \mathcal{B}\Big), \tag{8.21}$$

where $\bullet$ denotes element-wise multiplication and $\mathcal{B}$ is some generalization of an *adjoint non-uniform discrete Fourier transform* [83] to an 'integrated Fourier basis', i.e. $\int_{S_j} e^{-2\pi \mathrm{i}\mathrm{y}\cdot \boldsymbol{l}}\, \mathrm{d}\sigma(\boldsymbol{y})$.

Remember that the starting point was the splitting $\mathcal{N} = \mathcal{N}_{NF} + \mathcal{N}_s$, where, due to the Fourier series approximation of $\mathcal{N}_s$, i.e. $\mathcal{F}\mathcal{N}_s$, the splitting of $\mathcal{N}$ reads now

$$\mathcal{N} = (\mathcal{N} - \mathcal{N}_s) + \mathcal{F}\mathcal{N}_s + (\mathcal{N}_s - \mathcal{F}\mathcal{N}_s). \tag{8.22}$$

Only the approximation $\mathcal{N} \approx (\mathcal{N} - \mathcal{N}_s) + \mathcal{F}\mathcal{N}_s = \mathcal{N}_{NF} + \mathcal{F}\mathcal{N}_s$ is taken into account, which introduces the error $\mathcal{N}_s - \mathcal{F}\mathcal{N}_s$, which, however, can be controlled by the size of the tensor grid, i.e. $\boldsymbol{n} = (n_1, n_2, n_3)$, and the near field ϵ, cf. definition of $\mathcal{N}_{\mathrm{NF}}$ in (8.16). Analysis of the error in connection with the choice for approximating $\mathcal{N}$ by a smooth function $\mathcal{N}_s$ in the far field region, see Sec. 8.5.3, will be given in section 8.7.

8.5 Non-Uniform FFT for the Single Layer Potential

For the computation of the tensor $\mathcal{B}$ with entries $b_{\boldsymbol{l}} = \sum_{j=1}^{M} g_j \int_{S_j} e^{-2\pi \mathrm{i} \boldsymbol{y}\cdot \boldsymbol{l}}\, \mathrm{d}\sigma(\boldsymbol{y})$ one goes similar lines as for the efficient computation of the adjoint non-uniform discrete Fourier transform (NDFT) [83].

The essential step is a *gridding procedure* of the data $(g_j)_{j=1\ldots M}$ and FFT of the resulting tensor containing the 'smeared' source strengths. Hereby, gridding is done by convoluting the data with localized functions, whereas this is undone in Fourier space. The result is a generalization of the discrete Fourier transform to non-equispaced data.

First a well-localized univariate *window function* φ is introduced, e.g. a Gaussian function or Kaiser-Bessel function, see Sec. 8.5.2 for more details, with a uniformly convergent Fourier series of its $1-$periodic extension, i.e.

$$\widetilde{\varphi}(x) := \sum_{r\in\mathbb{Z}} \varphi(x+r). \tag{8.23}$$

For 3 dimensions one simply takes the *tensor product* of the univariate functions to obtain a multivariate window function, i.e.

$$\widetilde{\Phi}(\boldsymbol{x}) := \prod_{q=1}^{3} \widetilde{\varphi}(x^{(q)}). \tag{8.24}$$

For ease of computation, the truncated version of $\widetilde{\Phi}$ is introduced with some *truncation (or cut-off) parameter* $m \ll \min_{q=1\ldots 3} n_q$, $m \in \mathbb{N}$, $\boldsymbol{n} = (n_1, n_2, n_3)$, i.e.

$$\widetilde{\Psi}(\boldsymbol{x}) := \prod_{q=1}^{3} \widetilde{\varphi}(x^{(q)})\, \chi_{[-\frac{m}{n_q}, \frac{m}{n_q}]}(x^{(q)}), \tag{8.25}$$

where χ is the indicator function [1].

Then an auxiliary tensor $\mathcal{A} = (a_{\boldsymbol{r}})_{\boldsymbol{r}\in I_{\alpha \boldsymbol{n}}}$ is computed, where $\alpha > 1$ is an *over-sampling factor*, i.e.

$$a_{\boldsymbol{r}} := \sum_{j=1}^{M} g_j \int_{S_j} \widetilde{\Psi}(\boldsymbol{r} \bullet (\alpha\, \boldsymbol{n})^{-1} - \boldsymbol{y})\, \mathrm{d}\sigma(\boldsymbol{y}), \tag{8.26}$$

where $\bullet$ denotes element-wise multiplication and $(\alpha\, \boldsymbol{n})^{-1}$ is meant component-wise and corresponds to the mesh size of the auxiliary tensor grid, see Fig. 8.1.

Formula (8.26) can be seen as *gridding* of the source strengths g_j on an auxiliary tensor grid of size $|I_{\alpha \boldsymbol{n}}|$. The desired tensor $\mathcal{B} = (b_{\boldsymbol{l}})_{\boldsymbol{l}\in I_{\boldsymbol{n}}}$ in (8.20) can be computed by the Fourier transform

---

[1] $\chi_{[a,b]}(x) = 1$ for x in $[a,b]$ and 0 else.

of $\mathcal{A}$. More precisely, a function f is defined according to the definition of $\mathcal{A}$ by

$$f(\boldsymbol{x}) := \sum_{j=1}^{M} g_j \int_{S_j} \widetilde{\Phi}(\boldsymbol{x} - \boldsymbol{y})\,\mathrm{d}\sigma(\boldsymbol{y}). \tag{8.27}$$

By expressing the Fourier coefficients of f in two different ways, one will end up with a simple formula for computing the tensor $\mathcal{B}$.
First the Fourier coefficients of f (cf. (8.27)) are approximated by the trapezoidal rule for $\boldsymbol{l} \in I_{\boldsymbol{n}}$ and $\widetilde{\Phi}$ by the truncated version $\widetilde{\Psi}$ in (8.25), i.e.

$$c_{\boldsymbol{l}}(f) = \int_{\mathbb{T}} f(\boldsymbol{x})\, e^{-2\pi \mathrm{i} \boldsymbol{x}\cdot\boldsymbol{l}}\,\mathrm{d}\boldsymbol{x} \approx \frac{1}{|I_{\alpha\,\boldsymbol{n}}|} \sum_{\boldsymbol{r}\in I_{\alpha\,\boldsymbol{n}}} \underbrace{\Big(\sum_{j=1}^{M} g_j \int_{S_j} \widetilde{\Psi}(\boldsymbol{r} \bullet (\alpha\,\boldsymbol{n})^{-1} - \boldsymbol{y})\,\mathrm{d}\sigma(\boldsymbol{y})\Big)}_{=a_{\boldsymbol{r}}} e^{-2\pi\mathrm{i}(\boldsymbol{r}\bullet(\alpha\,\boldsymbol{n})^{-1})\cdot\boldsymbol{l}}, \tag{8.28}$$

which can be computed by a multivariate FFT of the tensor $\mathcal{A}$.

On the other hand, one also obtains an approximation of $c_{\boldsymbol{l}}(f)$ by inserting the truncated Fourier series of $\widetilde{\Phi}$, i.e.

$$\widetilde{\Phi}(\boldsymbol{x}) \approx \sum_{\boldsymbol{l}\in I_{\boldsymbol{n}}} c_{\boldsymbol{l}}(\widetilde{\Phi})\, e^{2\pi \mathrm{i} x\cdot\boldsymbol{l}}, \tag{8.29}$$

into the expression for the function f, i.e.

$$f(\boldsymbol{x}) \approx \sum_{\boldsymbol{l}\in I_{\boldsymbol{n}}} \underbrace{\Big(\sum_{j=1}^{M} g_j\, c_{\boldsymbol{l}}(\widetilde{\Phi}) \int_{S_j} e^{-2\pi \mathrm{i} y\cdot\boldsymbol{l}}\,\mathrm{d}\sigma(\boldsymbol{y})\Big)}_{=c_{\boldsymbol{l}}(f)} e^{2\pi \mathrm{i} x\cdot\boldsymbol{l}} \tag{8.30}$$

$$= \sum_{\boldsymbol{l}\in I_{\boldsymbol{n}}} \Big(c_{\boldsymbol{l}}(\widetilde{\Phi}) \underbrace{\sum_{j=1}^{M} g_j \int_{S_j} e^{-2\pi \mathrm{i} y\cdot\boldsymbol{l}}\,\mathrm{d}\sigma(\boldsymbol{y})\Big)}_{=b_{\boldsymbol{l}}} e^{2\pi \mathrm{i} x\cdot\boldsymbol{l}}. \tag{8.31}$$

Thus, the following relation holds ($\boldsymbol{l} \in I_{\boldsymbol{n}}$):

$$b_{\boldsymbol{l}} = c_{\boldsymbol{l}}(f)/c_{\boldsymbol{l}}(\widetilde{\Phi}). \tag{8.32}$$

Overall the computation of $\mathcal{B}$ consists of computing the coefficients $c_{\boldsymbol{l}}(f)$ in (8.28) by a multivariate FFT of the gridding tensor $\mathcal{A}$, followed by element-wise division by the precomputed coefficients $c_{\boldsymbol{l}}(\widetilde{\Phi})$.
Hence, these two steps together scale $\mathcal{O}(|I_{\alpha\boldsymbol{n}}|\,\log(|I_{\alpha\boldsymbol{n}}|) + |I_{\boldsymbol{n}}|)$. As will be shown in the next

section, the computation of the gridding tensor $\mathcal{A}$ can be done linearly in the number of surface elements, i.e. $O(M)$. Hence, in total, computing $\mathcal{B}$ scales $O(M + |I_{\alpha \boldsymbol{n}}| \log(|I_{\alpha \boldsymbol{n}}|))$.

Observe that, alternatively to the above procedure for computing the tensor $\mathcal{B}$, one also could have directly transformed the expression into a discrete sum by using quadrature, i.e

$$b_{\boldsymbol{l}} = \sum_{j=1}^{M} g_j \int_{S_j} e^{-2\pi \mathrm{i} \mathbf{y} \cdot \boldsymbol{l}} \, \mathrm{d}\sigma(\mathbf{y}) \approx \sum_{j=1}^{M} \sum_{s=1}^{Q_j} \omega_{j,s} g_j \, e^{-2\pi \mathrm{i} \mathbf{y}_{j,s} \cdot \boldsymbol{l}} \equiv \sum_{k=1}^{Q} \widetilde{\omega}_k \, e^{-2\pi \mathrm{i} \mathbf{y}_k \cdot \boldsymbol{l}}, \tag{8.33}$$

where k is a long index, e.g. $k = j + (s-1)M$. Eqn. (8.33) could then be computed by a standard adjoint NFFT [27] in $O(Q + |I_{\alpha \boldsymbol{n}}| \log(|I_{\alpha \boldsymbol{n}}|))$, $Q := \sum_{j=1}^{M} Q_j$ operations, also compare with [28]. Since the number of quadrature points Q might be very large, this approach is rather impractical. For that reason, the proposed method uses Eqn. (8.32) for the computation of the coefficients $b_{\boldsymbol{l}}$, where the integrals can be precomputed in a setup phase of a micromagnetic simulation, compare with Alg. 14.

However, at least it gives us a direct analogy to the standard adjoint NFFT. In particular, the choice of window functions can be justified, since, basically the same error estimates with respect to the *cut-off parameter* m and *over-sampling factor* α hold for the (standard) NFFT and the proposed method, see Sec. 8.5.2.

8.5.1 Computation of the Gridding Tensor

Remember the computation of the tensor $\mathcal{A}$ (compare with (8.26)), i.e.

$$a_{\boldsymbol{r}} = \sum_{j=1}^{M} g_j \int_{S_j} \widetilde{\Psi}(\boldsymbol{r} \bullet (\alpha \, \boldsymbol{n})^{-1} - \boldsymbol{y}) \, \mathrm{d}\sigma(\boldsymbol{y}). \tag{8.34}$$

The aim is to compute (8.34) through sparse summation by exploiting the locality of the function $\widetilde{\Psi}$.

Assume further that the domain is scaled into the hypercube $(-1/4, 1/4)^3$, hence there also holds $\Omega \subset \mathbb{T}$.

A triangle of the surface mesh is given as $S_j \equiv \{\boldsymbol{y}_{0,j}, \dots, \boldsymbol{y}_{2,j}, \ \boldsymbol{y}_{k,j} \neq \boldsymbol{y}_{l,j}, \ \text{for } k \neq l\}$ where

$$\boldsymbol{y} \in S_j \Leftrightarrow \exists \xi_1, \xi_2 \in \Delta_0 : \boldsymbol{y} = \boldsymbol{y}_{0,j} + \xi_1(\boldsymbol{y}_{1,j} - \boldsymbol{y}_{0,j}) + \xi_2(\boldsymbol{y}_{2,j} - \boldsymbol{y}_{0,j}), \tag{8.35}$$

where Δ_0 denotes the unit triangle in 2d.

In order to achieve linear complexity in M one defines a subset of $I_{\alpha \boldsymbol{n}}$ for each surface element

S_j that ensures that $\boldsymbol{r} \bullet (\alpha\,\boldsymbol{n})^{-1} - \boldsymbol{y}$ in (8.34) lies in the hypercube $\bigtimes_{q=1}^{3}[-mn_q^{-1}, mn_q^{-1}]$, i.e.

$$\begin{aligned} I_{\alpha\,\boldsymbol{n},m}(S_j) &:= \{\boldsymbol{l} \in I_{\alpha\,\boldsymbol{n}} \mid -m\boldsymbol{n}^{-1} \le_c (\alpha\,\boldsymbol{n})^{-1} \bullet \boldsymbol{l} - \boldsymbol{y} \le_c m\boldsymbol{n}^{-1},\ \boldsymbol{y} \in S_j\} && (8.36)\\ &= \{\boldsymbol{l} \in I_{\alpha\,\boldsymbol{n}} \mid \boldsymbol{y} \bullet \alpha\,\boldsymbol{n} - m\mathbf{1} \le_c \boldsymbol{l} \le_c \boldsymbol{y} \bullet \alpha\,\boldsymbol{n} + m\mathbf{1},\ \boldsymbol{y} \in S_j\}. && (8.37)\end{aligned}$$

The q−th component of (8.36) is denoted by $I^{(q)}_{\alpha\,\boldsymbol{n},m}(S_j)$, where the q−th components of the vector expressions in the definition is used.
For the sake of computation one may rewrite

$$I^{(q)}_{\alpha\boldsymbol{n},m}(S_j) = \{l_q \in I^{(q)}_{\alpha\,\boldsymbol{n}} \mid \alpha\, n_q \min_{\boldsymbol{y}\in S_j} y^{(q)} - m \le l_q \le \alpha\, n_q \max_{\boldsymbol{y}\in S_j} y^{(q)} + m\}. \tag{8.38}$$

From (8.35) it is easily seen that for the expressions $y^{(q),j}_{\min} := \min_{\boldsymbol{y}\in S_j} y^{(q)}$ and $y^{(q),j}_{\max} := \max_{\boldsymbol{y}\in S_j} y^{(q)}$ in (8.38) simply holds

$$\begin{aligned} y^{(q),j}_{\min} &= \min_{k=0,1,2} y^{(q),j}_k && (8.39)\\ y^{(q),j}_{\max} &= \max_{k=0,1,2} y^{(q),j}_k. && (8.40)\end{aligned}$$

Due to the assumption $\Omega \subset \mathbb{T}$, there holds $|I^{(q)}_{\alpha\,\boldsymbol{n},m}(S_j)| \le 2m + 1 + \alpha\, n_q \max_{j=1\ldots M} |y^{(q),j}_{\max} - y^{(q),j}_{\min}| =: \widetilde{m}_q$ and $|I_{\alpha\,\boldsymbol{n},m}(S_j)| \le \prod_q \widetilde{m}_q =: \mu$. Fig. 8.1 shows the index set $I^{(q)}_{\alpha\boldsymbol{n},m}(S_j)$.
The tensor $\mathcal{A}$ in (8.34) is now computed by only using the index sets $I_{\alpha\boldsymbol{n},m}(S_j)$ in $O(\mu M)$ operations:

- Initialize $\mathcal{A}$ with zeros
- For $j = 1\ldots M$ calculate the vector $\left(g_j \int_{S_j} \widetilde{\Psi}(\boldsymbol{l} \bullet \boldsymbol{n}^{-1} - \boldsymbol{y})\, \mathrm{d}\sigma(\boldsymbol{y})\right)_{\boldsymbol{l}\in I_{\alpha\,\boldsymbol{n},m}(S_j)}$ of length at most μ and add the corresponding components to $\mathcal{A}$.

Here, the integrals are precomputed, since they only depend on the given mesh. One may store the sparse matrix

$$\boldsymbol{A} := \Big(\int_{S_j} \widetilde{\Psi}(\boldsymbol{l} \bullet \boldsymbol{n}^{-1} - \boldsymbol{y})\, \mathrm{d}\sigma(\boldsymbol{y})\Big)_{j=1\ldots M,\, \boldsymbol{l}\in I_{\alpha\,\boldsymbol{n},m}(S_j)}. \tag{8.41}$$

Note that the above procedure for computing the tensor $\mathcal{A}$ is nothing else but a transposed sparse matrix-vector multiplication of $\mathbf{A}$ with the (column) vector $\boldsymbol{g} = (g_j)_{j=1\ldots M}$, i.e. $\mathrm{vec}(\mathcal{A}) = (\boldsymbol{g}^T\boldsymbol{A})^T$.
Nevertheless, since the integrals of (8.41) are smooth functions in the parameter $\boldsymbol{l}$, one can think of tensor compression for the rows, i.e. $\boldsymbol{A}(j,:) \in \bigotimes_{q=1}^{3} \mathbb{R}^{I^{(q)}_{\alpha\,\boldsymbol{n},m}(S_j)}$, reducing the storage to $\mu' M$ for $\mu' < \mu$ depending on the tensor format and the accuracy. Tab. 8.1 shows examples for

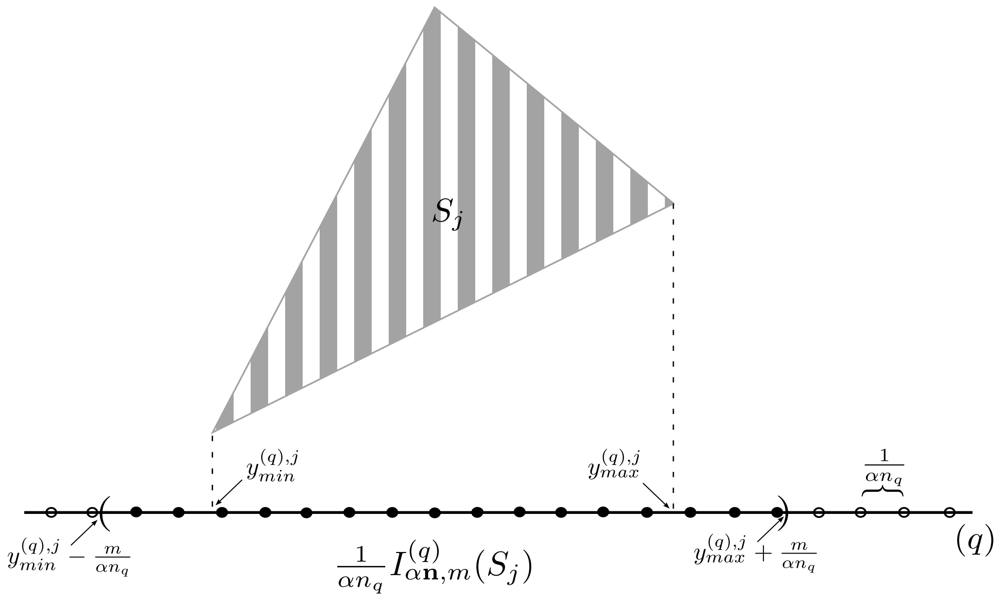
S_j
$y_{min}^{(q),j}$
$y_{max}^{(q),j}$
$\frac{1}{\alpha n_q}$
$y_{min}^{(q),j} - \frac{m}{\alpha n_q}$
$\frac{1}{\alpha n_q} I_{\alpha \mathbf{n},m}^{(q)}(S_j)$
$y_{max}^{(q),j} + \frac{m}{\alpha n_q}$
(q)

Figure 8.1: The q-th component of the index set $I_{\alpha \boldsymbol{n},m}(S_j)$ (filled dots). q denotes the space direction, S_j is one surface triangle, $\boldsymbol{n} = (n_1, n_2, n_3)$ the size of the tensor grid, α the oversampling factor, m the truncation parameter. $y_{\text{min/max}}^{(q),j}$ is the left and right most corner of the triangle, respectively. $1/(\alpha n_q)$ is the mesh size of the grid in the q-th direction. *Reprinted form [15]*

Table 8.1: Compression of $\boldsymbol{A}$ from surface mesh of a sphere by the Tensor Train format with accuracy 1e-8 measured in the relative Frobenius norm. $n_q \equiv 72$, $\alpha = 2$.

# surface elements	m	full (mb)	compressed (mb)
1.3e3	5	48	14
2.6e3	5	77	20
1.3e3	6	73	15
2.6e3	6	120	21

compression rates using tensor train (TT) approximation [68].

A further possibility to reduce storage is the usage of low rank tensor interpolation for a parameterized version of the integral in (8.41). More precise, the coordinates of the vertices of a general triangle give nine parameters and $\boldsymbol{l}$ additional three. Allowing this parameters to vary in an interval, one gets a (smooth) multivariate function defined on a tensor product domain. Multivariate Lagrange interpolation of this function with (black box) tensor compression for the coefficient tensor would give one the possibility to efficiently evaluate this interpolation. Hence, instead of precomputing the sparse matrix $\boldsymbol{A}$ one could precompute the structured coefficent tensor of the interpolation (e.g. in canonical format this leads to $O(Rm)$ storage costs for m interpolation points in one dimension and rank R.) and calculate the entries of $\boldsymbol{A}$ on the fly. The idea of tensor interpolation of parameterized intergrals in connection with fast computation of

BEM matrices is the core of a recent PhD thesis [86].

8.5.2 Window Functions

In [87, 88] it was shown that, in the case of *Gaussian, Sinc, cardinal B-splines* or *Kaiser-Bessel* window functions, the error for (adjoint) NFFT decays exponentially in the cut-off parameter m.

Hereby, Kaiser-Bessel functions have the fastest decaying error bound. For $n \in 2\mathbb{N}$ one defines the univariate Kaiser-Bessel function

$$\varphi(x) := \begin{cases} \frac{\sinh(b\sqrt{m^2-(\alpha n)^2x^2})}{\pi\sqrt{m^2-(\alpha n)^2x^2}}, & |x| \leq \frac{m}{\alpha n} \\ \frac{b}{\pi}, & |x| = \frac{m}{\alpha n} \\ \frac{\sin(b\sqrt{(\alpha n)^2x^2-m^2})}{\pi\sqrt{(\alpha n)^2x^2-m^2}}, & \text{else}, \end{cases} \tag{8.42}$$

where $b := \pi(2 - 1/\alpha)$. The Fourier coefficients are given by

$$c(\varphi)(k) = \begin{cases} \frac{1}{\alpha n} I_0(m\sqrt{b^2 - (\frac{2\pi k}{\alpha n})^2}), & |k| \leq \alpha n(1 - \frac{1}{2\alpha}) \\ 0, & \text{else}, \end{cases} \tag{8.43}$$

where I_0 is the *modified zero order Bessel-function of the first kind*.

For the univariate setting a bound for the relative error produced by NFFT is [88]

$$C(\alpha, m) = 4\pi(\sqrt{m} + m)(1 - 1/\alpha)^{1/4}\exp(-2\pi m\sqrt{1 - 1/\alpha}), \tag{8.44}$$

which already indicates small errors for m about 4 and $\alpha = 2$, see Fig. 8.2.

Note that this error bound is independent of n and M.

Since the method for computing $\mathcal{B}$ is mathematically equivalent to a NFFT if just accurate enough quadrature is used (compare with (8.33)), it makes sense to compare with the theoretical error bound (8.44) for standard NFFT. In this context, also note that the computation of $\mathcal{B}$ is stable regarding round off errors [88]. Fig.8.2 shows the cut-off parameter m versus the relative error in the maximum-norm, i.e. $\max_l |\mathcal{B} - \mathcal{B}_{\text{exact}}| / \max_l |\mathcal{B}_{\text{exact}}|$, for a triangular mesh of the surface of a sphere, randomly chosen values $g_j \in [-1, 1]$ and $\alpha = 2$.

8.5.3 Kernel Approximation

Focus now on the approximation of the Newtonian kernel $\mathcal{N}$ in a region $[\epsilon, \beta]$, $\beta > \epsilon > 0$, where one sets $\beta = 1/2$ due to the scaling convention $\Omega \subset (-1/4, 1/4)^3$.

As described in [76] the kernel $N(x) = 1/|x|$ can be approximated by exponential sums in an

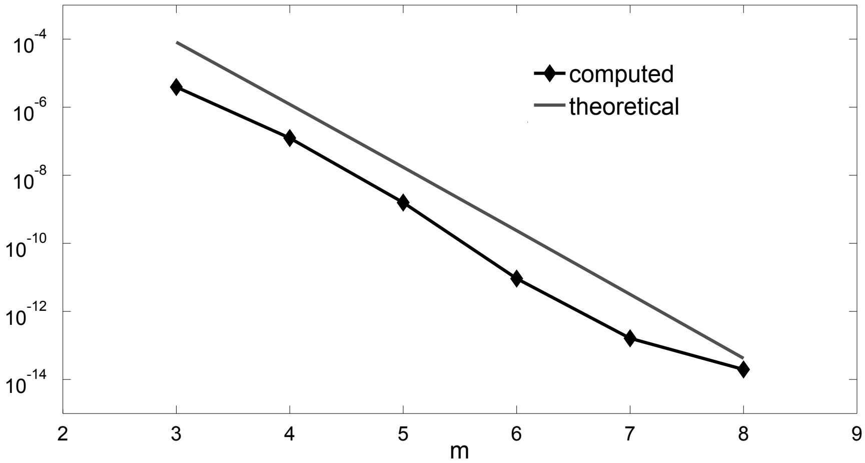

Figure 8.2: Relative error for the computation of $\mathcal{B}$ using Kaiser-Bessel functions, as defined in (8.42), on a triangular mesh ($M \sim 3e2$) of the surface of a sphere and randomly chosen values $g_j \in [-1, 1]$, $\alpha = 2$ and $n_q \equiv 32, q = 1 \ldots 3$. *Reprinted form [15]*

interval $[1, R]$, i.e.

$$N(x) \approx N_s(x) := \sum_{k=1}^{S} \omega_k e^{-\gamma_k x^2} \equiv \sum_{k=1}^{S} \omega_k N_s^{(k)}(x), \tag{8.45}$$

where the weights ω_k and nodes γ_k were computed for several configurations of the parameters R, S and uniform absolute error bound err. A simple transformation of the weights and nodes yields a corresponding approximation on the desired interval $[\epsilon, 1/2]$, i.e.

$$\omega_{\text{trans}} = \omega / h_{\text{min}} \tag{8.46}$$

$$\gamma_{\text{trans}} = \gamma / h_{\text{min}}^2 \tag{8.47}$$

$$err_{\text{trans}} = err / h_{\text{min}}, \tag{8.48}$$

where $h_{\text{min}} := 1/(2\sqrt{R})$.

For the numerical tests the computed values for $1/\sqrt{x}$ with $S = 21$ and $R = 7e4$ from [89] are chosen, yielding a uniform error of $5.79e-06$ in $[1.89e-03, 5.00e-01]$. Depending on the actual near field ϵ, the expansion (8.45) is truncated, taking only $S' \leq S$ terms, in order to have an accurate approximation only in the sub-interval $[\epsilon, 1/2]$. Fig. 8.3 shows the smooth approximation $N_s(.)$ for different number of terms in the expansion (8.45). Also see Fig. 8.6 in section 8.7 for the dependence of ϵ on S'.

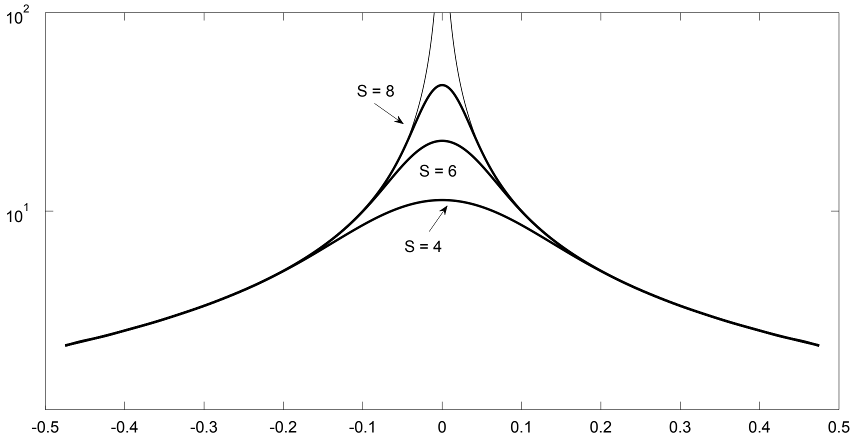

Figure 8.3: Approximation of $N(x) = 1/|x|$ by exponential sums. *Reprinted form [15]*

Note that $\mathcal{N}_s$ is a sum of separable functions, i.e.

$$\mathcal{N}_s(\boldsymbol{x}) = \sum_{k=1}^{S'} \omega_k \, N_s^{(k)}(x_1) \, N_s^{(k)}(x_2) \, N_s^{(k)}(x_3). \tag{8.49}$$

Thus, the Fourier coefficients of $\mathcal{N}_s$ can be computed by the trapezoidal rule, i.e.

$$\begin{aligned} c_{\boldsymbol{l}}(\mathcal{N}_s) &= \int_{\mathbb{T}} \mathcal{N}_s(\boldsymbol{x}) \, e^{-2\pi \mathrm{i} \boldsymbol{x} \cdot \boldsymbol{l}} \, \mathrm{d}\boldsymbol{x} \approx \frac{1}{\prod_{q=1}^{3} n_q} \sum_{\boldsymbol{r} \in I_{\boldsymbol{n}}} \mathcal{N}_s(\boldsymbol{r} \bullet \boldsymbol{n}^{-1}) \, e^{-2\pi \mathrm{i} \boldsymbol{l} \cdot (\boldsymbol{r} \bullet \boldsymbol{n}^{-1})} \\ &= \sum_{k=1}^{S'} \omega_k \prod_{q=1}^{3} \underbrace{\Big(\frac{1}{n_q} \sum_{r_q \in I_{\boldsymbol{n}}^{(q)}} N_s^{(k)}(r_q/n_q) \, e^{-2\pi \mathrm{i} l_q \, r_q/n_q} \Big)}_{=:\widetilde{c}_{l_q}(N_s^{(k)})} = \sum_{k=1}^{S'} \omega_k \, \widetilde{c}_{l_1}(N_s^{(k)}) \widetilde{c}_{l_2}(N_s^{(k)}) \widetilde{c}_{l_3}(N_s^{(k)}), \end{aligned} \tag{8.50}$$

where the $\widetilde{c}_{l_q}(N_s^{(k)})$ are the approximations (trapezoidal rule) of the Fourier coefficients $c_{l_q}(N_s^{(k)}) := \int_{-1/2}^{1/2} N_s^{(k)}(x_q) \, e^{-2\pi \mathrm{i} x_q l_q} \, \mathrm{d}x_q$, which are computed using one-dimensional FFT.

When discretized on a tensor grid, (8.49) and (8.50) are *canonical tensors of rank S'* (previously denoted as $\mathcal{C}_{\boldsymbol{n},S'}$, cf. section 4.2). This fact allows one to store only $S' \sum_{q=1}^{3} n_q$ complex numbers, instead of $\prod_{q=1}^{3} n_q$ for all Fourier coefficients of the multivariate function $\mathcal{N}_s$. However, additional $\mathcal{O}(S')$ operations have to be performed on runtime to calculate one entry of the tensor $c_{\boldsymbol{l}}(\mathcal{N}_s)$ from its factorized representation (8.50).

8.5.4 Near field correction

The near field correction is determined by (cf. (8.17))

$$\phi_2^{NF}(\boldsymbol{x}_i) \approx \sum_{j=1}^{M} g_j \int_{S_j} \mathcal{N}_{\mathrm{NF}}(\boldsymbol{x}_i - \boldsymbol{y}) \,\mathrm{d}\sigma(\boldsymbol{y}). \tag{8.51}$$

Since $\mathcal{N}_{\mathrm{NF}}$ has small support, (8.51) only has to be computed for surface elements that have less or equal distance than ϵ to the target point $\boldsymbol{x}_i$, i.e. for summation only the *index sets*

$$I_{\mathrm{NF}_\epsilon}(\boldsymbol{x}_i) := \{j \in \{1, \dots, M\} \mid \mathrm{d}(\boldsymbol{x}_i, S_j) \leq \epsilon\}, \tag{8.52}$$

are used, i.e.

$$\phi_2^{NF}(\boldsymbol{x}_i) \approx \sum_{j \in I_{\mathrm{NF}_\epsilon}(\boldsymbol{x}_i)} g_j \int_{S_j} \mathcal{N}_{\mathrm{NF}}(\boldsymbol{x}_i - \boldsymbol{y}) \,\mathrm{d}\sigma(\boldsymbol{y}). \tag{8.53}$$

It can be verified in a straight forward manner that there holds the following set relation:

$$\bigcup_{i=1}^{N} I_{\mathrm{NF}_\epsilon}(\boldsymbol{x}_i) \times i = \bigcup_{j=1}^{M} j \times I_{\mathrm{NF}_\epsilon}(S_j), \tag{8.54}$$

where

$$I_{\mathrm{NF}_\epsilon}(S_j) := \{i \in \{1, \dots, N\} \mid \mathrm{d}(\boldsymbol{x}_i, S_j) \leq \epsilon\}. \tag{8.55}$$

By using the relation above, one can sum up (8.51) in a similar way like the gridding tensor in $\mathcal{O}(M)$ operations, cf. Sec. 8.5.1, i.e.

- Initialize ϕ_2^{NF} with zeros
- For $j = 1 \dots M$ calculate the vector $\left(g_j \int_{S_j} \mathcal{N}_{\mathrm{NF}}(\boldsymbol{x}_i - \boldsymbol{y}) \,\mathrm{d}\sigma(\boldsymbol{y})\right)_{i \in I_{\mathrm{NF}_\epsilon}(S_j)}$ and add the corresponding components to ϕ_2^{NF}.

The integrals are again precomputed and stored in a sparse matrix. For reasonably uniform distribution of nodes nearby the boundary, one may assume that $\nu := \max |I_{\mathrm{NF}_\epsilon}(S_j)|$ is much smaller than N. The complexity of the calculation of ϕ_2^{NF} is therefore at most $\mathcal{O}(\nu M)$.
For the (weakly) singular cases in (8.51), i.e. $\boldsymbol{x}_i \in S_j$, one can use the following substitutions. Assume S_j has the vertices $\boldsymbol{x}_1, \boldsymbol{x}_2$ and $\boldsymbol{x}_3$ and one wants to evaluate at $\boldsymbol{x}_2$. The triangle S_j is parameterized by

$$p : \Delta_0 \to \mathbb{R}^3, (s, t) \mapsto \boldsymbol{x}_2 + s(\boldsymbol{x}_1 - \boldsymbol{x}_2) + t(\boldsymbol{x}_3 - \boldsymbol{x}_1), \tag{8.56}$$

where Δ_0 is the unit triangle in the plane. After the substitution $s \to s$ and $t \to st$ with Jacobian determinant $|J| = s$ the integration domain gets the unit square in the plane and the integral gets non-singular. Thus, one can treat it by tensor product Gaussian quadrature.
For more information the reader is referred to [90].

8.6 Numerics

The tests were taken on a Linux Workstation with a hexa-core AMD Phenom II X6 1090T processor and 16 GB RAM. Matlab 7.13.0 and the C library NFFT 3 [27] were used.
Alg. 14 shows a pseudo-code of the described method for solving problem (2.22) by the ansatz (8.5) and (8.6) using the proposed fast evaluation scheme for the single layer potential. Whereby, the total algorithm is divided into a setup and a computation phase. In any micromagnetic solver the computation phase is part of the effective field evaluation, which has to be done at every step of the iterative solution procedure. The setup phase only depends on the geometry of the problem and thus has to be done only once for a given problem. In the following, it is shown that the computational effort for the computation phase scales linearly with the problem size. In Alg. 14 for computing the magnetic scalar potential the first step of the computation phase is the solution of a Dirichlet problem for ϕ_1. Since the problem is sparse and the LU decomposition is done in a setup phase the complexity is linear. The numerical experiments in this section also show linear complexity for the computation of ϕ_2.

First the method for computing the single layer potential is tested for a cube. Fig. 8.4 shows the cpu-times in seconds of the different parts of the algorithm for randomly chosen values $g_j \in [-1, 1]$. The parts *gridding* and *fft* correspond to the computation of the tensor $\mathcal{B}$, compare with (8.32), where times for the element-wise division with the precomputed Fourier coefficients of the window function were included in the times for the FFT. The times for the element-wise multiplication of the Fourier coefficients of $\mathcal{N}_s$ and $\mathcal{B}$ are suppressed, since they are negligible. For the NFFT the C library NFFT 3 was used. Moreover, the setting $m = 5$, $\alpha = 2$ and $n_q \equiv 48$, both, in the gridding method as well as in the NFFT was used. Further, ϵ was chosen such that ν in the complexity of the near field correction was below 3e2. The smooth approximation of $\mathcal{N}$ was truncated after $S' = 6$ terms. One can observe linear complexity of all parts except the FFT that is constant for constant n_q. Note that the NFFT is linear in the number of nodes of the mesh.

Next the method is compared for the case of uniform magnetization, i.e. $\boldsymbol{m} = (0, 0, 1)^T$, $M_s = 1$,

Algorithm 14 FEM/BEM-NFFT for the scalar potential

```
Require: $\boldsymbol{m} \in (H^1(\Omega))^3$, mesh $\mathcal{T}$ of $\Omega \subset (-1/4, 1/4)^3 \subset \mathbb{T}$, $\boldsymbol{n} \in 2\mathbb{N}^3$, $\epsilon > 0$, $m \in \mathbb{N}$,
    $\alpha \in \mathbb{N}, \alpha \geq 2$
Ensure: $\phi^{int} \in H^1(\Omega)$
  Setup
    • Compute the LU decomposition of the stiffness matrix and the linear operator for the
      RHS, cf. Sec. 8.3
    • Compute the matrix $\boldsymbol{A}$ from (8.41), cf. Sec. 8.5.1
    • Compute the Fourier coefficients of the window functions from (8.43), cf. Sec. 8.5.2
    • Compute the Fourier coefficients of the Kernel approximation, i.e. $(c_{\boldsymbol{l}}(\mathcal{N}_s))_{\boldsymbol{l} \in I_{\boldsymbol{n}}}$,
      cf. Sec. 8.5.3
    • Compute the integrals of the near field correction, cf. Sec. 8.5.4
  Actual computation
    • Solve the linear system (8.13) for $\phi_1^{int}$
    • Compute $\phi_2^s$:
      – Compute the tensor $\mathcal{A}$ in (8.34)
      – Compute the multivariate FFT of the tensor $\mathcal{A}$, cf. (8.28)
      – Compute the tensor $\mathcal{B}$ by formula (8.32)
      – Compute $\mathcal{D} := (c_{\boldsymbol{l}}(\mathcal{N}_s))_{\boldsymbol{l} \in I_{\boldsymbol{n}}} \bullet \mathcal{B}$
      – Compute the NFFT of $\mathcal{D}$ to obtain $\phi_2^s$, cf. (8.21)
    • Compute $\phi_2^{NF}$ as described in Sec. 8.5.4
  $\phi_2^{int} \leftarrow \phi_2^s + \phi_2^{NF}$
  $\phi^{int} \leftarrow \phi_1^{int} + \phi_2^{int}$
```

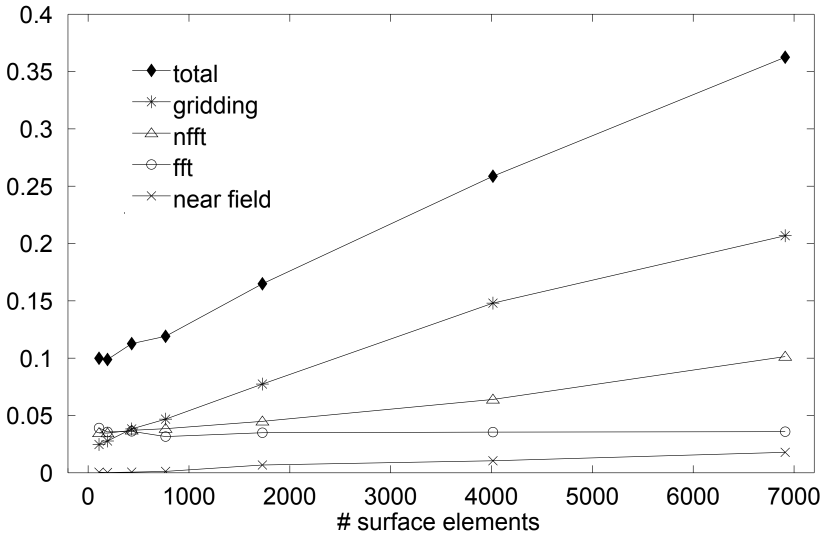

Figure 8.4: Cpu-times (sec) versus number of surface elements for the calculation of the single-layer potential in a cube. $n_q \equiv 48$, $\alpha = 2$ and $m = 5$. *Reprinted form [15]*

in a sphere with radius R and center at zero, were the exact solution is given as

$$\phi^{int}(\boldsymbol{x}) = \frac{x^{(3)}}{3}, \tag{8.57}$$

$$\phi^{ext}(\boldsymbol{x}) = R^3 \frac{x^{(3)}}{3\,\|\boldsymbol{x}\|^3}, \tag{8.58}$$

which can easily be verified by inserting into (2.22). The same parameters as in the first experiment were used. Fig. 8.5 shows number of nodes versus the maximum of the point-wise absolute error at the nodes of the mesh, i.e. l_∞-error, of the computed solution in $\overline{\Omega}$ compared to the analytical value. One observes linear error decay.

Note that $\phi_1^{int} \equiv 0$ (compare with (8.5)), since $\nabla \cdot \boldsymbol{m} \equiv 0$ in Ω. Hence, this example only tests the computation of ϕ_2, cf. (8.6).

In order to include the computation of ϕ_1 in the tests, take the example $\boldsymbol{m}(\boldsymbol{x}) = \boldsymbol{x}/\|\boldsymbol{x}\|$ in a sphere with radius R and center at zero with exact solution

$$\phi^{int}(\boldsymbol{x}) = \|\boldsymbol{x}\| - R, \tag{8.59}$$

$$\phi^{ext}(\boldsymbol{x}) = 0. \tag{8.60}$$

Since ϕ is zero at the boundary, there holds $\phi^{int} = \phi_1^{int}$ in (8.5) and $[\frac{\partial \phi_2}{\partial \boldsymbol{n}}] = 0$, hence $\phi_2 = 0$ in (8.6). Nevertheless, in the numerical test also the computation of ϕ_2 through (8.7) were

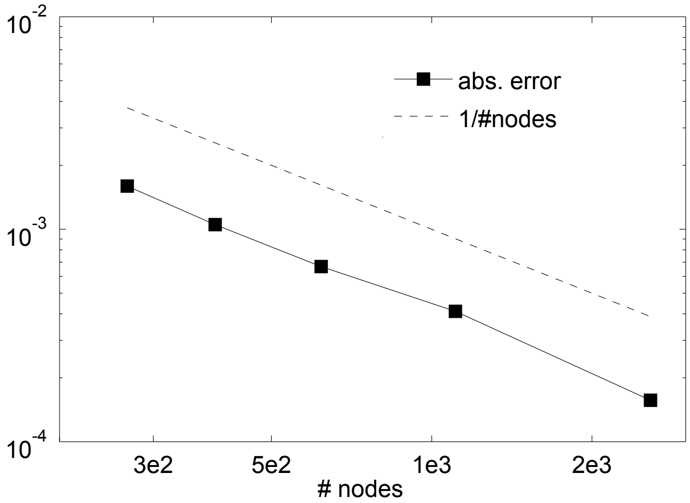

Figure 8.5: Maximum of the absolute error for uniform magnetization in a sphere with radius 0.2 and center at zero. *Reprinted form [15]*

Table 8.2: Errors for magnetization $\boldsymbol{m}(\boldsymbol{x}) = \boldsymbol{x}/\,\|\boldsymbol{x}\|$ in a sphere with radius 0.2 and center at zero measured in the $L^2(\Omega)$-norm, $H^1(\Omega)$-semi-norm and $H^1(\Omega)$-norm.

# elements	# nodes	$\|.\|_{L^2(\Omega)}$	$\lvert.\rvert_{H^1(\Omega)}$	$\|.\|_{H^1(\Omega)}$
3058	678	1.0e-3	9.9e-3	1.0e-2
7188	1490	8.9e-4	7.7e-3	7.7e-3
14169	2232	7.2e-4	3.0e-3	3.1e-3

included. Tab.8.2 shows the errors in the $L_2(\Omega)$-norm, $H^1(\Omega)$-semi-norm and $H^1(\Omega)$-norm, i.e.

$$\left\|\phi - \phi_{\mathrm{appr}}\right\|_{L^2(\Omega)} = \Big(\int_\Omega (\phi - \phi_{\mathrm{appr}})^2(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}\Big)^{1/2}, \tag{8.61}$$

$$|\phi - \phi_{\mathrm{appr}}|_{H^1(\Omega)} = \Big(\sum_{q=1}^{3} \left\|\partial_q(\phi - \phi_{\mathrm{appr}})\right\|^2_{L^2(\Omega)}\Big)^{1/2}, \tag{8.62}$$

$$\left\|\phi - \phi_{\mathrm{appr}}\right\|_{H^1(\Omega)} = \Big(\left\|\phi - \phi_{\mathrm{appr}}\right\|^2_{L^2(\Omega)} + |\phi - \phi_{\mathrm{appr}}|^2_{H^1(\Omega)}\Big)^{1/2}, \tag{8.63}$$

respectively, which was calculated by taking the nodal interpolations of the exact and computed solutions. The parameter setting $n = 72$, $\alpha = 2$ and cut-off parameters $m = 5$ was used. Note that the $H^1(\Omega)$-semi-norm and thus the $H^1(\Omega)$-norm take the errors of the stray field $\boldsymbol{h}_s = -\nabla\phi$ into account.

8.7 A Closer Look at Errors and Complexity

Remember that the approximation for the single layer potential (cf. section 8.4) is split into a near field correction and a smooth part, i.e.

$$\phi_2(\boldsymbol{x}_i) \approx \sum_{j=1}^{M} g_j \int_{S_j} \mathcal{N}_{\mathrm{NF}}(\boldsymbol{x}_i - \boldsymbol{y})\,\mathrm{d}\sigma(\boldsymbol{y}) + \sum_{j=1}^{M} g_j \int_{S_j} \mathcal{N}_s(\boldsymbol{x}_i - \boldsymbol{y})\,\mathrm{d}\sigma(\boldsymbol{y}) =: \phi_2^{\mathrm{NF}}(\boldsymbol{x}_i) + \phi_2^s(\boldsymbol{x}_i). \tag{8.64}$$

The scheme for the smooth part written in a compact form (also compare with (8.21) and (8.32)) reads

$$\phi_2^s \approx \mathrm{NFFT}\Big((c_{\boldsymbol{l}}(\mathcal{N}_s))_{\boldsymbol{l}\in I_{\boldsymbol{n}}} \bullet (\mathrm{FFT}(\mathcal{A})/c_{\boldsymbol{l}}(\widetilde{\Phi}))_{\boldsymbol{l}\in I_{\boldsymbol{n}}} \Big). \tag{8.65}$$

As pointed out in section 8.5 and also numerically tested in section 8.5.2, the error that arises from approximating the tensor $\mathcal{B}$ with entries $b_{\boldsymbol{l}} = \sum_{j=1}^{M} g_j \int_{S_j} e^{-2\pi \mathrm{i} \boldsymbol{y}\cdot\boldsymbol{l}}\,\mathrm{d}\sigma(\boldsymbol{y})$, i.e.

$$\mathcal{B} \approx (\mathrm{FFT}(\mathcal{A})/c_{\boldsymbol{l}}(\widetilde{\Phi}))_{\boldsymbol{l}\in I_{\boldsymbol{n}}}, \tag{8.66}$$

behaves like that for the standard NFFT. The error bound in section 8.5.2 shows that this error decays exponentially with increasing cut-off parameter m and is independent of the tensor grid size $|I_{\boldsymbol{n}}|$.

In order to be able to analyze the error dependence on $\boldsymbol{n}$ of the whole scheme (8.64), one has to look at the kernel splitting in more detail, i.e.

$$\mathcal{N} = (\mathcal{N} - \mathcal{N}_s) + \mathcal{F}\mathcal{N}_s + (\mathcal{N}_s - \mathcal{F}\mathcal{N}_s). \tag{8.67}$$

In the scheme (8.64) with (8.65) for the smooth part, the contribution of $\mathcal{N}_s - \mathcal{F}\mathcal{N}_s$ is neglected. Thus, the error occurring from the approximation of the smooth kernel approximation $\mathcal{N}_s$ by its Fourier series approximation $\mathcal{F}\mathcal{N}_s$ has to be analyzed. Moreover, in order to get linear complexity in the near field correction, $(\mathcal{N} - \mathcal{N}_s)(\boldsymbol{x}) = 0$ is assumed for $\|\boldsymbol{x}\| > \epsilon$. Due to the approximation by exponential sums, compare with section 8.5.3, this yields a (uniform) error in the interval $[\epsilon, \beta]$, which is denoted as E_{NF} in the following estimate. Overall, for the essential error arising in the summation in (8.64) holds

$$|\phi_2^{\mathrm{NF}}(\boldsymbol{x}_i) + \phi_2^s(\boldsymbol{x}_i) - (\widetilde{\phi_2^{\mathrm{NF}}}(\boldsymbol{x}_i) + \widetilde{\phi_2^s}(\boldsymbol{x}_i))| \leq |\partial\Omega|\ \|\boldsymbol{g}\|_1 \left(E_{\mathrm{NF}} + \max_{\|\boldsymbol{x}\|<\frac{1}{2}} |\mathcal{N}_s(\boldsymbol{x}) - \mathcal{F}\mathcal{N}_s(\boldsymbol{x})| \right), \tag{8.68}$$

where $\widetilde{\phi_2^{\mathrm{NF}}}(\boldsymbol{x}_i) + \widetilde{\phi_2^s}(\boldsymbol{x}_i)$ denotes the computed values and $\|\boldsymbol{g}\|_1 := \sum_{j=1}^{M} |g_j|$.
Due to the tensor product structure of the Fourier coefficients of $\mathcal{N}_s$ (compare with section 8.5.3), also the Fourier series approximation has this structure, i.e.

$$\mathcal{F}\mathcal{N}_s(\boldsymbol{x}) = \sum_{k=1}^{S'} \omega_k \mathcal{F}N_s^{(k)}(x_1)\, \mathcal{F}N_s^{(k)}(x_2)\, \mathcal{F}N_s^{(k)}(x_3). \tag{8.69}$$

It follows

$$\begin{aligned}\max_{\|\boldsymbol{x}\|<\frac{1}{2}} |\mathcal{N}_s(\boldsymbol{x}) - \mathcal{F}\mathcal{N}_s(\boldsymbol{x})| &= \sum_{k=1}^{S'} |\omega_k| \max_{\|\boldsymbol{x}\|<\frac{1}{2}} |N_s^{(k)}(x_1)\, N_s^{(k)}(x_2)\, N_s^{(k)}(x_3) - \mathcal{F}N_s^{(k)}(x_1)\, \mathcal{F}N_s^{(k)}(x_2)\, \mathcal{F}N_s^{(k)}(x_3)| \\ &\leq \sum_{k=1}^{S'} C_k |\omega_k| \sum_{q=1}^{3} \max_{\|\boldsymbol{x}\|<\frac{1}{2}} |N_s^{(k)}(x_q) - \mathcal{F}N_s^{(k)}(x_q)|, \end{aligned} \tag{8.70}$$

where an telescoping sum like $abc - \tilde{a}\tilde{b}\tilde{c} = (a-\tilde{a})bc + (b-\tilde{b})\tilde{a}c + (c-\tilde{c})\tilde{a}\tilde{b}$ was used and C_k is an upper bound for the products $bc, \tilde{a}c$ and $\tilde{a}\tilde{b}$.
Adapting the proof of Th. 3.4 in [91] for the univariate case, the error $\max_{x_q<\frac{1}{2}} |N_s^{(k)}(x_q) - \mathcal{F}N_s^{(k)}(x_q)|$ for $N_s^{(k)}(x_q) = e^{-\gamma_k x_q^2}$ can be estimated by

$$\max_{x_q<\frac{1}{2}} |N_s^{(k)}(x_q) - \mathcal{F}N_s^{(k)}(x_q)| \leq A(\gamma_k, \eta_k^{(q)}) + B(\gamma_k, \eta_k^{(q)}), \tag{8.71}$$

where $\eta_k^{(q)} := \frac{\pi n_q}{2\sqrt{\gamma_k}}$ and $A(\gamma_k, \eta_k^{(q)}) \sim e^{-(\eta_k^{(q)})^2}$ and $B(\gamma_k, \eta_k^{(q)}) \sim e^{-\gamma_k/4}/\eta_k^{(q)}$.
The consequences of (8.71) are twofold. First, for small γ_k the term $B(\gamma_k, \eta_k^{(q)})$ only gets small for large n_q, whereas for large γ_k this term is negligible. In the first case (small γ_k) one can use boundary regularization or further scaling the domain Ω into, e.g., $(-0.2, 0.2)^3$. This reduces the error $N_s^{(k)} - \mathcal{F}N_s^{(k)}$ in general, [91].
On the other hand (8.71) suggests to choose n_q in the order of $\sqrt{\gamma_k}$, i.e. $n_q \sim \sqrt{\gamma_k}$, such that $\eta_k^{(q)} \geq 1$ and thus $A(\gamma_k, \eta_k^{(q)})$ is small.

By reducing the number of terms in the exponential sum of N_s, one can observe an exponential increase of ϵ (the left border of the interval of validity for the uniform approximation, cf. Fig. 8.6), e.g. linear fitting gives, for the certain choice of the coefficients ω_k and γ_k in section 8.5.3, $\log\epsilon \sim -0.28311S' - 0.15471$. Moreover, $\gamma_{S'}$ increases exponentially with increasing S'; linear fitting gives $\log\gamma_{S'} \sim 0.57352S' + 6.5187$. Thus, one gets approximately $\gamma_{S'} \sim 1/\epsilon^{2.0258}$. Fitting with coefficients from the precomputed list with $S = 28$, the same $R = 7e4$ (yields the same interval of validity for the uniform approximation as that from section 8.5.3 but with the lower error 2.34e-08) gives the similar estimate $\gamma_{S'} \sim 1/\epsilon^{2.0449}$. Finally, fitting with coefficients corresponding to lists with different R (means different interval of va-

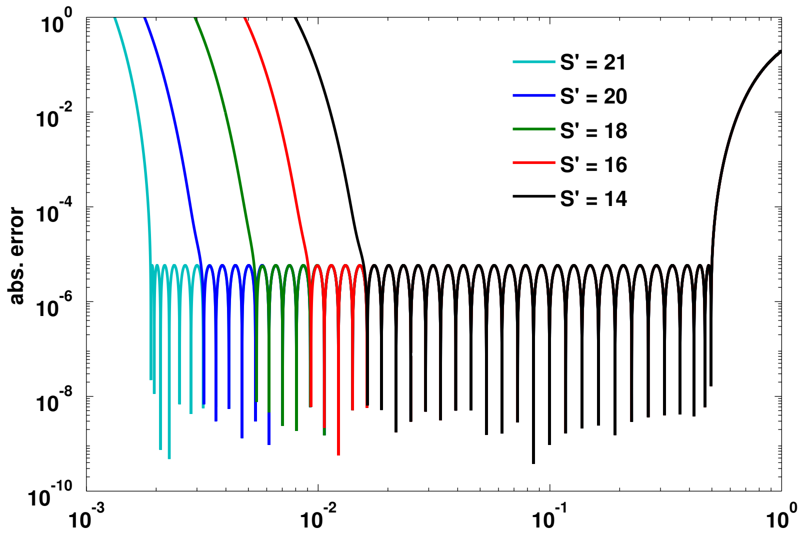

Figure 8.6: Dependence of the absolute error on the number of terms of the approximation of the function $1/|x|$ by exponential sums with nodes/weights from section 8.5.3.

lidity for the approximation with all S terms) confirm the trend $\gamma_{S'} \sim 1/\epsilon^2$.
This, in connection with $n_q \sim \sqrt{\gamma_k}$, gives an approximate asymptotic relation between the tensor grid size n_q and the 'near field' ϵ which is about

$$n_q = O(\epsilon^{-1}). \tag{8.72}$$

Now, linear complexity of the near field computation requires that $\nu := \max |I_{\mathrm{NF}_\epsilon}(S_j)|$ is much smaller than the total number of nodes, i.e. N, cf. section 8.5.4. Assuming that the nodes near the boundary are reasonably uniformly distributed, means that the 'ϵ–balls' $I_{\mathrm{NF}_\epsilon}(S_j)$ contain about the same number of nodes, namely ν. If the even more idealistic assumption is made that the whole mesh is roughly uniform, then the volume of an ϵ-ball is proportional to the ratio ν/N, i.e. there should hold approximately $\epsilon \sim (\nu/N)^{1/3}$.
Together with (8.72) this combines to

$$n_q = O(N^{1/3}). \tag{8.73}$$

Since the complexity of the proposed scheme for (8.64) is $O(M + N + (\prod_{q=1}^3 n_q)(\log \prod_{q=1}^3 n_q))$, the assumption of a roughly uniform mesh, together with the error investigation above, gives rise to the scaling

$$O(M + N + N \log N). \tag{8.74}$$

8.8 Computing the Stray Field

Here is briefly described how to derive the stray field from the finite element solution of the scalar potential of the previous sections.

The P1 finite element method yields the approximated values u_i of the scalar potential at the nodes of the mesh. This defines a unique (P1) nodal interpolation, i.e.

$$\phi(\boldsymbol{x}) \approx \sum_{i=1}^{N} u_i \, \eta_i(\boldsymbol{x}), \tag{8.75}$$

where the η_i are the nodal basis, compare with section 8.3.

From nodal interpolation:

The approximation of the *demagnetizing field* (stray field) $\boldsymbol{h}_d = -\nabla\phi$ is therefore the element-wise constant function (P0)

$$\boldsymbol{h}_d(\boldsymbol{x}) \approx -\sum_{i=1}^{N} u_i \, \nabla\eta_i(\boldsymbol{x}). \tag{8.76}$$

Within one tetrahedron T_j, the (approximate) scalar potential is the affine function

$$\phi(\boldsymbol{x}) \,|_{T_j} \approx \sum_{\alpha=1}^{4} u_\alpha \, \varphi_\alpha^{(T_j)}(\boldsymbol{x}), \tag{8.77}$$

where for the affine element basis functions holds $\varphi_\alpha^{(T_j)}(\boldsymbol{x}_\beta) = \delta_{\alpha\beta}$ (*Kronecker*-δ). The stray field approximation in each element has therefore the constant value

$$\boldsymbol{h}_d(\boldsymbol{x}) \,|_{T_j} \approx \sum_{\alpha=1}^{4} u_\alpha \, \nabla\varphi_\alpha^{(T_j)}. \tag{8.78}$$

From mass-lumping and midpoint rule:

In order to get a P1-approximation of the stray field one can use the approach from [72]: Two different approximations of the demagnetizing energy

$$E_d = -\frac{\mu_0}{2} \int_\Omega M_s^2 \boldsymbol{m} \cdot \boldsymbol{h}_d, \tag{8.79}$$

are compared in order to derive a gradient operator (matrix). Here physical units for the energy are used (in contrast to reduced units, see Sec. 2.2).The first approximation ansatz is

$$E_d \approx -\frac{\mu_0 M_s}{2} \boldsymbol{\mu}^T \boldsymbol{h}_d, \tag{8.80}$$

where $\boldsymbol{h}_d$ is assumed to be a vector of length N containing the values of the stray field at the nodes and $\boldsymbol{\mu}$ a vector of length N containing the volume- and spacial averaged magnetic moments at the nodes. The latter one is assembled in a usual local to global process by a loop over the elements

$$\boldsymbol{\mu}(\boldsymbol{k}) \mathrel{+}= \frac{1}{4} M_s^{(T_j)} |T_j|, \tag{8.81}$$

where $\boldsymbol{k}$ are the global indices of the nodes of element T_j.
This approximation can be rewritten as

$$E_d \approx -\frac{\mu_0 M_s}{2} \boldsymbol{m}^T \boldsymbol{L} \boldsymbol{h}_d, \tag{8.82}$$

where $\boldsymbol{m} = (m_1^{(x)}, m_1^{(y)}, m_1^{(z)}, \ldots, m_N^{(x)}, m_N^{(y)}, m_N^{(z)})^T \in \mathbb{R}^{3N\times 1}$ is the mesh vector of the unit magnetization and $\boldsymbol{L}$ a $3N \times 3N$ diagonal matrix which consists itself of 3×3 diagonal blocks $\mathrm{diag}(\boldsymbol{\mu}_i, \boldsymbol{\mu}_i, \boldsymbol{\mu}_i)$.
The second approximation is done by midpoint integration of (8.79) over all elements and $-\nabla\phi$ inserted for the stray field, i.e.

$$E_d \approx \frac{\mu_0 M_s^2}{2} \sum_{T_j \in \mathcal{T}} \int_{T_j} \boldsymbol{m} \cdot \nabla\phi, \tag{8.83}$$

which yields a similar equation like (8.82), namely

$$E_d \approx \frac{\mu_0 M_s^2}{2} \boldsymbol{m}^T \boldsymbol{G} \boldsymbol{u}, \tag{8.84}$$

where $\boldsymbol{G}$ is a sparse $3N \times N$ matrix (it is the transposed matrix of the divergence matrix which is used for the right hand side in section 8.3) and $\boldsymbol{u} = (u_i)_{i=1\ldots N}$ the vector of node values of the scalar potential.
Comparing (8.82) and (8.84) yields

$$\boldsymbol{h}_d = -\boldsymbol{L}^{-1} \boldsymbol{G} \boldsymbol{u}. \tag{8.85}$$

Table 8.3 shows the errors of the stray field for the flower state in the unit cube in [18] with the same reference value and computed by (8.85) and (8.82). The scalar potential is calculated by the FEM/BEM-NFFT algorithm described in this chapter.

From variational formulation

Table 8.3: Errors of the stray field energy (reduced units) for a flower state in the unit cube computed by (8.85) and (8.82).

# elements	# nodes	energy	error in percent
6000	1331	0.157	2.4
10368	2197	0.156	2.2
24576	4913	0.155	1.2
93750	17576	0.152	0.7

The method above is a special way to solve the variational (projection) problem

$$\int_\Omega \boldsymbol{h}_d \cdot \boldsymbol{m} = -\int_\Omega \nabla\phi \cdot \boldsymbol{m}, \tag{8.86}$$

for all $\boldsymbol{m} \in H^1(\Omega)$. Namely, within a P1 discretization of (8.86) the RHS is treated by the midpoint rule, yielding $-\boldsymbol{m}^T \boldsymbol{G}\boldsymbol{u}$, where the LHS is approximated using (8.81), yielding $\boldsymbol{m}^T \boldsymbol{L}\boldsymbol{h}_d$. Alternatively, one can use a different Galerkin FE ansatz (e.g. higher order) and use appropriate quadrature, for instance by using the finite element package FEniCs [92].

8.9 Summary and Discussion

A P1 finite element method for the computation of the micromagnetic scalar potential was introduced, which is based on the ansatz of García-Cervera and Roma. The potential is computed by a splitting $\phi = \phi_1 + \phi_2$, where ϕ_1 is solved by a Poisson equation with zero Dirichlet boundary conditions and ϕ_2 by evaluation of the single layer potential. The contribution is the development of a method to compute the single layer potential at all nodes of a tetrahedral mesh in linear time (or almost linear, see section 8.7) by means of Fourier approximation of a smoothed kernel and near field correction.

The discretized integral operator splits into a part with smooth and singular kernel. The latter one has small support and therefore allows a computation by sparse summation, while for the smooth part Fourier techniques can be applied. Due to the unstructured FE-mesh, generalizations of discrete Fourier transforms arise, which can be implemented efficiently.

Overall the method scales both linear in the number of surface elements and nodes, whereas the usage of an auxiliary tensor grid gives an additional almost linear dependence on the tensor grid size. The considerations from section 8.7 indicate an almost linear complexity with respect to the number of nodes of the mesh for this part.

Similar, the storage requirements are linear in the number of surface elements, where further tensor train compression was introduced in order to reduce the constant in the storage estimate for the gridding procedure.

Exponential sums were used to obtain an entirely smooth and separable approximation of the Fourier coefficients of the Newtonian potential. As a consequence of the above mentioned splitting, which includes a near field correction, the only essential error of the method (within the P1 FEM framework) is due to this approximation, cf. section 8.7. Nevertheless, numerical experiments for test cases with known analytical solutions show accurate approximations.

Future work could deal with a possible extension to higher order finite element and boundary element methods. Also an application of the proposed method could be used to accelerate a $P0$ approximation of the integral representation of the potential (2.23), i.e.

$$\phi(\boldsymbol{x}) \approx \sum_{T_j} \oint_{\partial T_j} \frac{\boldsymbol{m}_j \cdot \boldsymbol{n}(\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|} \, \mathrm{d}\sigma(\boldsymbol{y}), \tag{8.87}$$

Also $P1$ or higher order approximation for (2.23) itself is conceivable.

An interesting possibility for future work is the parallelization of the scheme (8.89) by the parallel FFT (PFFT) package [93].

Also the application of the novel sub-linearly scaling *sparse FFT* [94, 95] could be investigated, especially in the context of discrete Fourier transform of the gridding tensor (cf. Sec. 8.5.1).

Low-Rank Tensor Version

Here an idea concerning cost-reduction of the FFT in Alg. 14 is briefly discussed, but not fully analyzed yet. The key point is to use tensor compression for the tensor $\mathcal{A} \in \mathbb{R}^{I_{\alpha \boldsymbol{n}}} := \bigotimes_{q=1}^{3} \mathbb{R}^{I_{\alpha \boldsymbol{n}}^{(q)}}$ in (8.34) followed by FFT and NFFT for structured tensors (compare with section 4.5). Since the tensor grid parameter $\boldsymbol{n}$ controls the accuracy of the method (amongst others, e.g. cut-off m or near field ϵ), it is desirable to choose it as large as possible. On the other hand, also remember that this parameter has no connection to the underlying problem like geometry or magnetization. So the choice of $\boldsymbol{n}$ is a trade-off between accuracy and complexity/storage. The gridding process

$$a_{\boldsymbol{r}} := \sum_{j=1}^{M} g_j \int_{S_j} \widetilde{\Psi}(\boldsymbol{r} \bullet (\alpha\, \boldsymbol{n})^{-1} - \boldsymbol{y}) \, \mathrm{d}\sigma(\boldsymbol{y}), \tag{8.88}$$

smooths the data on a regular tensor grid and constructs the tensor $\mathcal{A}$, compare with section 8.5. It seems (somehow) natural and within the scope of this thesis to ask whether the 'smooth' tensor $\mathcal{A}$ permits a 'low-rank' tensor representation, if only for certain parameters (e.g. cut-off m, mesh parameters/properties or variation of the sources g_j). In order to see what implications this would have for the computation of ϕ_2^s, the scheme is here rewritten in a compact form (also compare with (8.21) and (8.32)), i.e.

$$\phi_2^s = \mathrm{NFFT}\Big(\big(c_l(\mathcal{N}_s)\big)_{l\in I_n} \bullet \big(\mathrm{FFT}(\mathcal{A})/c_l(\widetilde{\Phi})\big)_{l\in I_n}\Big). \tag{8.89}$$

Remember that $(c_l(\mathcal{N}_s))_{l\in I_n} \in \mathcal{C}_{\boldsymbol{n},S'}$ and $(c_l(\widetilde{\Phi}))_{l\in I_n} \in \mathcal{C}_{\boldsymbol{n},1}$, meaning they are canonical tensors of rank S' and 1, respectively.

Assume now $\mathcal{A} \in \mathcal{C}_{\alpha\boldsymbol{n},R}$. From chapter 4 section 4.5 Lemma 3 one gets $\mathrm{FFT}(\mathcal{A}) \in \mathcal{C}_{\alpha\boldsymbol{n},R}$ and hence $(\mathrm{FFT}(\mathcal{A})/c_l(\widetilde{\Phi}))_{l\in I_n} \in \mathcal{C}_{\boldsymbol{n},R}$. From the Hadamard product of canonical tensors (compare with section 4.2) one finally has

$$\text{If } \mathcal{A} \in \mathcal{C}_{\alpha\boldsymbol{n},R} \Longrightarrow (c_l(\mathcal{N}_s))_{l\in I_n} \bullet (\mathrm{FFT}(\mathcal{A})/c_l(\widetilde{\Phi}))_{l\in I_n} \in \mathcal{C}_{\boldsymbol{n},S'R}. \tag{8.90}$$

The costs for forming (8.90) are $O(R\,\alpha\,n\log\alpha\,n)$ for the FFT and $O(R\,n{+}R\,S'n)$ for the Hadamard products.

Similar as for FFT for canonical tensors the NFFT for CP tensors is reduced to 1-dimensional NFFT. Namely, the following statement can be recalculated in a straight forward manner (for an idea of proof compare with the proof of Lemma 3 and 4).

Lemma 10. *For a canonical tensor* $\mathcal{A} = [\![\, \lambda;\, \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}\,]\!] \in \mathcal{C}_{\boldsymbol{n},r}$, $a_l = \sum_{s=1}^{r} \lambda_s\, u^{(1)}_{l_1 s}\, u^{(2)}_{l_2 s}\, u^{(3)}_{l_3 s}$ *the NFFT is given by*

$$NFFT(\mathcal{A}) = \Big[NFFT_{1d}(\boldsymbol{U}^{(1)}) \bullet NFFT_{1d}(\boldsymbol{U}^{(2)}) \bullet NFFT_{1d}(\boldsymbol{U}^{(3)})\Big]\,\lambda, \tag{8.91}$$

where the (1-d) NFFT is only taken along each column of a factor matrix.

The costs for the CP-NFFT in Lemma 8.91 are ($n_q \equiv n$) $O(r\,m\,N + r\,n\,\log n)$, compare with $O(m^3 N + n^3 \log n^3)$ for ordinary NFFT.

Hence, if the gridded tensor $\mathcal{A}$ is in canonical form with rank R, the costs for computing ϕ_2^s through (8.89) are $O(RS'\,m\,N{+}RS'\,n\,\log n){+}O(R\,\alpha\,n\log\alpha\,n)$, compare with $O(m^3N{+}n^3\log n^3{+}(\alpha n)^3\log(\alpha n)^3)$ for dense $\mathcal{A}$.

In addition, one had to add the costs for computing $\mathcal{A}$ as a canonical tensor. For that purpose, one could use *black box approximation* for canonical tensors [56] or cross approximation for the TT-format [96, 97] with subsequent conversion to Tucker tensors plus approximation of the core in the CP format (Tucker to CP approximation) by e.g. an ALS algorithm. These methods allow approximations without forming the dense/full tensor $\mathcal{A}$ explicitly.

Tab. 8.4 shows an example for compression of $\mathcal{A}$ in the case of a meshed sphere with radius 0.5 and center at zero and a flower magnetization [18]. For testing purposes the CP rank was determined by first pre-calculating $\mathcal{A}$ completely followed by an ALS based Tucker to CP approximation. The ranks are in the scale of the tensor grid parameter n, whereas the results indicate lower ranks for the cases where the cut-off m is larger and where the mesh is finer.

Table 8.4: CP-compression of $\mathcal{A}$ in the case of a meshed sphere with radius 0.5 and center at zero and a flower magnetization. The over-sampling factor α is 2. The compression error in the relative Frobenius norm was below one percent.

# surface elements	αn	m	CP-rank R
420	144	5	400
980	144	5	250
980	96	5	60
1794	96	5	50
1794	144	5	110
1794	144	3	250
1794	144	6	95

The same test but carried out with randomly chosen sources g_j, instead of those arising from a (parameterized) flower state, fails in terms of a clear increase of ranks (even a rank of 500 yields an error above one percent). Thus, the 'smoothness' of the underlying source function is a crucial factor whether low ranks can be achieved in principle.

A cylinder geometry (basis in the $x-y$-plane with radius 0.5, height 1) was tested together with the vortex magnetization from [18]. For the parameters $m = 5, \alpha n = 96$ and a very coarse surface mesh of only 300 elements the compression rank is 37 (error below one percent). Inserting the flower magnetization yields a compression rank of 23. It is very likely that the geometry (in connection with the magnetization configuration) plays an important role. At the current stage it is unclear whether the compression method yields any advantages over the 'plain' scheme, however, further investigations on that have to be done.

Chapter 9

Conclusions

This thesis starts with a brief summary of background information on micromagnetics. The *micromagnetic energy minimization problem* is formulated as constrained optimization problem in the continuous and the discrete setting. Several algorithms which address this problem are introduced, including a new variant of the steepest descent method, a *semi-implicit scheme*. This method is compared with a *quasi Newton method* applied to the unconstrained version of the micromagnetic energy minimization problem. Although the modified steepest descent method is simple and more efficient than ordinary steepest descent, the quasi Newton approach outperforms it in two test cases in terms of needed function evaluations. On the contrary, the computational costs of both methods are comparable. Moreover, *penalty approaches* from non-linear programming are applied to micromagnetics, as well as, *Newton's method to the Karush-Kuhn-Tucker (NKKT) conditions*. Both approaches are less efficient than modified steepest descent and quasi Newton on the unconstrained problem. Nevertheless, while penalty methods are re-used later in the context of *low-rank energy minimization*, the NKKT method turns out to be a generalization of the widely used *method of Alouges*.

In order to minimize the energy on large tensor grids, the data-sparse tensor formats are introduced. A detailed description of tensor formats and approximation of tensors is given. Incidentally, a FFT-based method to apply filtering of disturbed multi-way data is found.

A tensor grid method for computing the stray field for tensor structured input is described and mathematically analyzed. Kronecker product structure of the demagnetizing operator is proven, which later gives rise to a similar structure for the Hessian of a second order discretization of the total magnetic energy. This structure allows the efficient evaluation for tensor structured input. Later, the tensor grid stray field method is even accelerated by means of FFT.

A whole chapter is dedicated to approximation of magnetization configurations by the Tucker format. The tests indicate an asymptotically logarithmic rank-growth with respect to the side-length of a (rather) hard magnetic cube (with no external field). In the remanent case it is also shown that the minimization of the energy has a regularizing effect on randomly disturbed initial magnetization. In addition, the compression ranks corresponding to a prescribed tolerance during demagnetization are adaptively determined. The dependence of the ranks with respect to the tolerance ϵ for the compression is found to be approximately $\widetilde{r} \sim \mathcal{O}(\log \epsilon^{-1})$ if the external field strength is not near the coercive field. Right before and during switching of the magnetization the ranks 'explode' in an oscillating manner, while for the region away from the critical field ranks do not grow drastically.

Finally, the micromagnetic energy minimization problem subject to low-rank tensors is investigated and analyzed. An algorithm is introduced, which is based on low-rank updates and minimization within the canonical tensor representation. In principal, this method allows applying large grids due to the sublinear scaling in the volume size. This is useful for large ferromagnetic particles which demand a high resolution due to constraints related to the exchange length or domain wall width.

The final section is dedicated to a novel finite element/boundary element (FEM/BEM) algorithm, which benefits from *non-uniform FFT*, which is especially adapted to boundary integrals. The method, which calculates the scalar potential, scales quasi optimal in the number of volume and surface elements.

List of Algorithms

1 Barzilai-Borwein switch; `BBswitch(`$\tau_{min}, \tau_{max}, \tau_n^1, \tau_n^2, \alpha$`)` 28
2 Semi-implicit steepest descent scheme; `SemiImplSD(`$\boldsymbol{m}^0$`,tol)` 29
3 Limited memory Quasi Newton; `lBFGS(`$\boldsymbol{m}^0$`,tol)` 30
4 Quadratic penalty method; `qPM(`$\boldsymbol{m}^0$`,tol,`μ_0`,`$\tau_n \to 0$`)` 32
5 Newton's method for KKT system; `NKKT(`$\boldsymbol{m}^0$`,`λ^0`,tol)` 35
6 HOOI (ALS); `HOOI(`$\mathcal{X}, \boldsymbol{r}_0, \epsilon$`)` 45
7 HOOI (ALS) for disturbed data; `HOOId(`$\mathcal{R}, \boldsymbol{r}_0, \epsilon, b$`)` 50
8 Sinc-Quadrature for (5.15); `sinc(`R, c_0`)` 55
9 Gaussian matrices; `gaussmat(`τ_l, p, h_1, h_2, h_3`)` 58
10 Scalar potential; `scpot(`$\mathcal{M}^{(1)}, \mathcal{M}^{(2)}, \mathcal{M}^{(3)}, h_1, h_2, h_3, R, c_0$`)` 58
11 Scalar potential DFFT; `scpotDFFT(`$\mathcal{M}^{(1)}, \mathcal{M}^{(2)}, \mathcal{M}^{(3)}, \boldsymbol{h}, R, c_0,$ `tol` > 0`)` . . . 76
12 qPM for CP-magnetization; `qPM_CP(`$\mathcal{M}^0$`,`$\boldsymbol{p}(\mathcal{M}^0)$`,tol,` μ_0`,`$\tau_n \to 0$`)` 101
13 Successive rank-k update method; `qPM_CP_rk(`$\mathcal{M}^0$`,`$\boldsymbol{r}$`,`$\texttt{tol}_{1,2}$`,` μ_0`,`$\tau_n \to 0$`)` . . 104
14 FEM/BEM-NFFT for the scalar potential 127

List of Figures

3.1 Demagnetizing curve of 70nm $Nd_2Fe_{14}B$ cube, i.e. projection of magnetization onto field axis $(1, 0, 1)^T$ ($|m|$) as function of external field h. Uniaxial anisotropy axis is $(0, 0, 1)^T$, $\Delta h = 0.005$, opt-tol (rel. l^2) $=$ 1e-11. 31

5.1 Comparison of three quadrature methods for Eq. (5.6) with $\rho = x^2$: Block-Gauss-Legendre on 10 logarithmically scaled subintervals of the interval $[0, 2^{10}] = [0, 2] \cup [2, 4] \cup \ldots \cup [2^9, 2^{10}]$ with $q = 4$ and 6 quadrature nodes each, respectively; the *Sinc*-quadrature with a total of 40 nodes and $c_0 = 1.85$ and Gauss-Laguerre quadrature with 40 nodes. 56

5.2 Vector of tensor grid spacings, i.e. $\boldsymbol{h}_p$ 57

5.3 Computation time for the scalar potential by Alg. 10 for given magnetization in Tucker format (ranks constant $\equiv 10$); $R = 30$. 'Tucker full' indicates full ('exact') computation for Eqn. (5.23), whereas 'Tucker approx' are the times for approximately calculating Eqn. (5.23) within the Tucker format using Alg. 6 with `eps`= 1e-8 for approximate summation. Precomputation of the 'Gaussian matrices' is excluded. 60

5.4 Absolute error of collocation scheme (5.36). *Reprinted from [26].* 67

5.5 Relative error for sinc quadrature of (5.54) with substitution $\tau = \sinh(t)$. $c_0 = 2.15$. *Reprinted from [26].* 73

5.6 Complexity and relative error in the energy for Alg. 11 tested for uniform magnetization. *Reprinted from [26].* 76

5.7 Ranks of the computed stray field using Alg. 11 with accuracy 1e-12 and a flower-like magnetization state. *Reprinted from [26].* 77

6.1 Left: 'Staircase approximation' of a sphere. Right: Approximation on the left but smoothed by Gaussian filter. 79

6.2 Demagnetizing curve of 70nm $Nd_2Fe_{14}B$ cube, i.e. projection of magnetization onto field axis $(1, 0, 10)^T$ ($|m|$) as function of external field h. The brighter curve is calculated with Tucker compression with tolerance 1e-7, whereas the dark blue curve uses full magnetization. The calculated coercive field is in both cases $h_c = -2.767$. Uniaxial anisotropy axis is $(0, 0, 1)^T$, $\Delta h = 0.005$, opt-tol = 1e-6. 83

6.3 Average ranks and corresponding error (maximum- and l^2-norm) as a function of compression tolerance (tol) for the z-component of different states in the demagnetizing curve (cf. Fig. 6.2) for a 70nm $Nd_2Fe_{14}B$ cube. Axis of applied field is parallel to $(1, 0, 10)^T$, uniaxial anisotropy axis is $(0, 0, 1)^T$, $\Delta h = 0.005$, opt-tol = 1e-6. 84

6.4 Average ranks of the magnetization components in the demagnetizing curve (cf. Fig. 6.2 bright curve) as a function of external field. Compression tolerance used in HOOI is 1e-7. 84

6.5 Average ranks of the magnetization components in the demagnetizing curve (inset) of a 70nm $Nd_2Fe_{14}B$ cube as a function of external field. Compression tolerance used in HOOI is 1e-6. The stopping tolerance in the optimization is 1e-5. 85

6.6 Average ranks of the magnetization components in the demagnetizing curve (inset) of a 70nm $Nd_2Fe_{14}B$ cube as a function of external field. Compression tolerance used in HOOI is 1e-8. The stopping tolerance in the optimization is 1e-7. 85

6.7 Average ranks of the magnetization components in the demagnetizing curve (inset) of a 35nm $Nd_2Fe_{14}B$ cube as a function of external field. Compression tolerance used in HOOI is 1e-7. The stopping tolerance in the optimization is 1e-6. 86

6.8 Average ranks of the magnetization components in the demagnetizing curve (inset) of a 100nm $Nd_2Fe_{14}B$ cube as a function of external field. Compression tolerance used in HOOI is 1e-7. The stopping tolerance in the optimization is 1e-6. 86

6.9 Average ranks of the magnetization components in the demagnetizing curve (inset) of a soft magnetic cube as a function of external field. Compression tolerance used in HOOI is 1e-7. The stopping tolerance in the optimization is 1e-6. 87

8.1 The q-th component of the index set $I_{\alpha\boldsymbol{n},m}(S_j)$ (filled dots). q denotes the space direction, S_j is one surface triangle, $\boldsymbol{n} = (n_1, n_2, n_3)$ the size of the tensor grid, α the over-sampling factor, m the truncation parameter. $y^{(q),j}_{\min/\max}$ is the left and right most corner of the triangle, respectively. $1/(\alpha n_q)$ is the mesh size of the grid in the q-th direction. *Reprinted form [15]* 121

8.2 Relative error for the computation of $\mathcal{B}$ using Kaiser-Bessel functions, as defined in (8.42), on a triangular mesh ($M \sim 3e2$) of the surface of a sphere and randomly chosen values $g_j \in [-1, 1]$, $\alpha = 2$ and $n_q \equiv 32, q = 1 \ldots 3$. *Reprinted form [15]* 123

8.3 Approximation of $N(x) = 1/|x|$ by exponential sums. *Reprinted form [15]* . . . 124

8.4 Cpu-times (sec) versus number of surface elements for the calculation of the single-layer potential in a cube. $n_q \equiv 48$, $\alpha = 2$ and $m = 5$. *Reprinted form [15]* 128

8.5 Maximum of the absolute error for uniform magnetization in a sphere with radius 0.2 and center at zero. *Reprinted form [15]* 129

8.6 Dependence of the absolute error on the number of terms of the approximation of the function $1/|x|$ by exponential sums with nodes/weights from section 8.5.3. 132

List of Tables

3.1 The number of iterations and function evaluations (energy and gradient are evaluated simultaneously) for different mesh-sizes ($N = n^3$) and tolerance `tol=` 1e-7 of Alg. 2 and Alg. 3 for cube with $\lambda = 8.4515$ and $Q = 0.05$. Initial state is always uniform magnetization in z-direction. M denotes the limited memory parameter in Alg. 3. 30
3.2 The number of function evaluations (energy and gradient are evaluated simultaneously) of Alg. 2 and Alg. 3 for calculation of the demagnetizing curve in Fig. 3.1. 31

4.1 Results for Alg. 7 for the case of $\max_j |n_j| =: \|\mathcal{N}\|_\infty = 1\text{e-}06$ and $\epsilon = 1\text{e-}08$. $\mathcal{X}$ was chosen randomly with ranks $r_q \equiv 10$, whereas $n_q \equiv 40$. The over-sampling parameter was $a = 2$. The first column gives the parameter m from (4.42). . . . 50

5.1 Average abs./rel. errors of sinc-quadrature on ρ-interval $[5e-05, 1e-02]$. Left: c_0-dependence of approximation of (5.6) ($R = 50$). Right: R-dependence of approximation of (5.6) ($c_0 = 1.85$). 55
5.2 Comparison of accuracy of Eqs. (5.29) and Alg. 10 ($R = 50$). Absolute ($Err_{10/50}$) and relative ($Relerr_{10/50}$) l^2-errors for a $N = 10^3$ (exact errors) and $N = 50^3$ (errors for 200 randomly chosen mesh-points) tensor product grid, g^{ten} indicates the order of Gaussian quadrature used for the integrals in (5.20), the integrals in (5.29) are evaluated exactly. 62
5.3 Absolute error in the energy between results for direct tensor integration algorithm and FEM/BEM for uniform magnetization distribution in the unit cube. N indicates the number of total nodes in the mesh. In the fourth column the rel. deviation of the energy values computed by the two numerical schemes is given; percentage is based on the true value, i.e. $e_d = 1/6\,[\mu_0^{-1} M_s^{-2}]$. 63

5.4 Absolute and relative deviation in the energy between results for direct tensor integration algorithm and finite element reference-value for magnetization distribution of a vortex in the unit cube given by Eqs. (5.32). In order to resolve the vortex, a non-uniform grid (geometrically refined towards the center of the cube) is used. The columns to the right show the minimal grid size, i.e. $\min h_j$, in the center of the cube and respectively the maximal value, i.e. $\max h_j$, next to the boundaries. 63

5.5 Absolute and relative deviation between results for direct tensor integration algorithm and finite element reference-value for magnetization distribution in the unit cube given by Eqs. (5.33). 64

6.1 Tucker approximation of indicator function of a 'staircase approximation' of a sphere by Alg. 6 with tolerance 1e-06. 79

6.2 Remanent states for uniform initial magnetization. Ranks $\bar{r}$ (average of ranks, i.e. $1/9 \sum_{p=1}^{3} \sum_{q=1}^{3} r_p^q$) determined with tolerance 1e-14. $Q = 1$ and $n = 2.5\lambda$. . . 80

6.3 Average ranks for Tucker approximation with tolerance ϵ for remanent states of randomly disturbed uniform initial magnetization for different stopping tolerance (opt-tol) of the maximum norm of the projected gradient. Grid size $n = 50$, $\lambda = 35$ and $Q = 1/2$. 81

7.1 Minimization w.r.t. fixed CP format with Alg. 12 with $\mu_0 = 10$ and $M_n = 1$ or 5. For each line in the table 15 loops in Alg. 12 were performed. Results for a nearly uniform flower (5.33) $[a = c = 5, b = 2]$ in the unit cube as initial magnetization with different CP ranks. Parameters are $Q = 0.05$ (z-direction) and $\lambda = 8.45[l_{ex}]$. Grid spacing is $h = 1/200$ and no external field is applied. In the first column the ranks are shown, columns 2-5 show the energies, column 6 shows the average z-component of the magnetization, column 7 shows the constraint violation measured by the function P_n from (7.10), column 8 shows the 2- norm of the projected gradient and the last column shows the number of function evaluations (function and gradient evaluations are counted as one evaluation). 101

7.2 Minimization with rank-1 update with Alg. 13. Results for a nearly uniform flower (5.33) [$a = c = 5$, $b = 2$] in the unit cube as initial magnetization with different CP ranks. Parameters are $Q = 0.05$ (z-direction) and $\lambda = 8.45[l_{ex}]$. Grid spacing is $h = 1/20$ and no external field is applied. The first column gives the iteration number in the outer loop and the corresponding ranks in the brackets. In each outer iteration 8 iterations in `qPM_CP` were performed, while no other stopping criterion was used. The subproblems in Alg. 12 are solved with a quasi-Newton method; $\mu_0 = 10$ and $M_n = 1$ or 5. In total 350 function evaluations (function and gradient evaluations are counted as one evaluation) were performed. Columns 2-5 show the energies, column 6 shows the average z-component of the magnetization, column 7 shows the constraint violation measured by the function P_n from (7.10), column 8 shows the 2- norm of the projected gradient. 105

7.3 Minimization with rank-1 update with Alg. 13. The same configurations as in Tab. 7.2 but with grid spacing $h = 1/200$. 106

7.4 Minimization with rank-1 update with Alg. 13. Results for the vortex configuration (5.32) [$r_c = 0.25$] in the unit cube as initial magnetization with different CP ranks. Parameters are $Q = 0.05$ (z-direction) and $\lambda = 8.45[l_{ex}]$. Grid spacing is $h = 1/20$ and no external field is applied. The first column gives the iteration number in the outer loop and the corresponding ranks in the brackets. In each outer iteration 8 iterations in `qPM_CP` were performed, while no other stopping criterion was used. In total 519 function evaluations (function and gradient evaluations are counted as one evaluation) were performed. Columns 2-5 show the energies, column 6 shows the average y-component of the magnetization, column 7 shows the constraint violation measured by the function P_n from (7.10), column 8 shows the 2- norm of the projected gradient. 106

7.5 Minimization with rank-2 update with Alg. 13. Results for the vortex configuration (5.32) [$r_c = 0.25$] in the unit cube as initial magnetization with different CP ranks. Parameters are $Q = 0.05$ (z-direction) and $\lambda = 8.45[l_{ex}]$. Grid spacing is $h = 1/20$ and no external field is applied. The first column gives the iteration number in the outer loop and the corresponding ranks in the brackets. In each outer iteration 8 iterations in `qPM_CP` were performed, while no other stopping criterion was used. In total 309 function evaluations (function and gradient evaluations are counted as one evaluation) were performed. Columns 2-5 show the energies, column 6 shows the average y-component of the magnetization, column 7 shows the constraint violation measured by the function P_n from (7.10), column 8 shows the 2- norm of the projected gradient. 107

7.6 Minimization with rank-1 update with Alg. 13 with external field in direction $(1,1,1)$ with strength $\|\boldsymbol{h}_{ext}\|_2 = 1$. Results for uniform magnetization in z-direction of the unit cube as initial magnetization with different CP ranks. Parameters are $Q = 0$ and $\lambda = 11.30[l_{ex}]$. Grid spacing is $h = 1/40$ and no anisotropy field is assumed. The first column gives the iteration number in the outer loop and the corresponding ranks in the brackets. In each outer iteration 15 iterations in `qPM_CP` were performed. The final stopping criterion was a projected gradient norm of less than 1e-5. In total 301 function evaluations (function and gradient evaluations are counted as one evaluation) were performed. Columns 2-5 show the energies, column 6-8 shows the average x, y, z-component of the magnetization, column 9 shows the constraint violation measured by the function P_n from (7.10), column 10 shows the 2- norm of the projected gradient. 107

8.1 Compression of $\boldsymbol{A}$ from surface mesh of a sphere by the Tensor Train format with accuracy 1e-8 measured in the relative Frobenius norm. $n_q \equiv 72$, $\alpha = 2$. . 121

8.2 Errors for magnetization $\boldsymbol{m}(\boldsymbol{x}) = \boldsymbol{x}/\|\boldsymbol{x}\|$ in a sphere with radius 0.2 and center at zero measured in the $L^2(\Omega)$-norm, $H^1(\Omega)$-semi-norm and $H^1(\Omega)$-norm. . . . 129

8.3 Errors of the stray field energy (reduced units) for a flower state in the unit cube computed by (8.85) and (8.82). 135

8.4 CP-compression of $\mathcal{A}$ in the case of a meshed sphere with radius 0.5 and center at zero and a flower magnetization. The over-sampling factor α is 2. The compression error in the relative Frobenius norm was below one percent. 138

Bibliography

[1] O Gutfleisch, M Willard, E Brück, C H Chen, S G Sankar, and J P Liu. Magnetic materials and devices for the 21st century: stronger, lighter, and more energy efficient. *Advanced materials*, 23(7):821–42, February 2011. ISSN 1521-4095. doi: 10.1002/adma.201002180. URL `http://www.ncbi.nlm.nih.gov/pubmed/21294168`.

[2] R Skomski, P Manchanda, P K Kumar, B. Balamurugan, A Kashyap, and D J Sellmyer. Predicting the Future of Permanent-Magnet Materials. *IEEE Transactions on Magnetics*, 49(7):3215–3220, July 2013. ISSN 0018-9464. doi: 10.1109/TMAG.2013.2248139. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6558995`.

[3] H Sepehri-Amin, T Ohkubo, S Nagashima, M Yano, T Shoji, A Kato, T Schrefl, and K Hono. High-coercivity ultrafine-grained anisotropic Nd–Fe–B magnets processed by hot deformation and the Nd–Cu grain boundary diffusion process. *Acta Materialia*, 2013. URL `http://www.sciencedirect.com/science/article/pii/S1359645413005697`.

[4] H Kronmüller. General micromagnetic theory. *Handbook of Magnetism and Advanced Magnetic Materials*, 2007.

[5] David S. Kinderlehrer and Ling Ma. Simulation of hysteresis in nonlinear systems. In H. Thomas Banks, editor, *North American Conference on Smart Structures and Materials*, pages 78–87, May 1994. doi: 10.1117/12.174200. URL `http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=958612`.

[6] L Grasedyck, D Kressner, and C Tobler. A literature survey of low-rank tensor approximation techniques. *arXiv preprint arXiv:1302.7121*, 2013.

[7] W F Brown. Micromagnetics. *Interscience Publishers, Wiley & Sons, New York*, 1963.

[8] A Aharoni. *Introduction to the Theory of Ferromagnetism*, volume 109. Oxford University Press, 2000.

[9] H.-A. Engel, E. I. Rashba, and B. I. Halperin. *Handbook of Magnetism and Advanced Magnetic Materials*. John Wiley & Sons Ltd., Chichester, UK, 2007.

[10] J. D. Jackson. Classical electrodynamics, 3rd ed. *Am. J. Phys*, 67(9), 1999. ISSN 00029505. doi: 10.1119/1.19136. URL `http://link.aip.org/link/?AJP/67/841/2&Agg=doi`.

[11] W F Brown. Magnetostatic principles in ferromagnetism. *Amsterdam*, 112, 1962.

[12] R.D. McMichael. Standard problem number 3, problem specification and reported solutions. *Micromagnetic Modeling Activity Group*, 1998. URL `http://www.ctcms.nist.gov/~rdm/mumag.html`.

[13] J E Miltat and M J Donahue. Numerical micromagnetics: Finite difference methods. *Handbook of Magnetism and Advanced Magnetic Materials*, 2007.

[14] W Rave, K Fabian, and A Hubert. Magnetic states of small cubic particles with uniaxial anisotropy. *Journal of Magnetism and Magnetic Materials*, 190(3):332–348, 1998.

[15] L Exl and T Schrefl. Non-uniform FFT for the finite element computation of the micromagnetic scalar potential. *arXiv preprint arXiv:1305.3162*, 2013.

[16] M. Aurada, M. Feischl, T. Führer, M. Karkulik, J. M. Melenk, and Dirk Praetorius. Classical fem-bem coupling methods: nonlinearities, well-posedness, and adaptivity. *Computational Mechanics*, pages 1–21, 2012.

[17] C. Carstensen and E. P. Stephan. Adaptive coupling of boundary elements and finite elements. *ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique*, 29(7):779–817, 1995.

[18] C Abert, L Exl, G Selke, A Drews, and T Schrefl. Numerical methods for the stray-field calculation: A comparison of recently developed algorithms. *Journal of Magnetism and Magnetic Materials*, 326:176–185, 2012.

[19] M. J. Donahue and R. D. McMichael. Micromagnetics on curved geometries using rectangular cells: Error correction and analysis. *IEEE Trans. Magn.*, 43(6):2878 – 2880, June 2007. doi: 10.1109/TMAG.2007.892865.

[20] J.L. Blue and M.R. Scheinfein. Using multipoles decreases computation time for magnetostatic self-energy. *IEEE Trans. Magn.*, 27(6):4778 –4780, nov 1991. ISSN 0018-9464. doi: 10.1109/20.278944.

[21] H.H. Long, E.T. Ong, Z.J. Liu, and E.P. Li. Fast fourier transform on multipoles for rapid calculation of magnetostatic fields. *IEEE Trans. Magn.*, 42(2):295 – 300, feb. 2006. ISSN 0018-9464. doi: 10.1109/TMAG.2005.861505.

[22] B Van de Wiele, F Olyslager, and L Dupré. Application of the fast multipole method for the evaluation of magnetostatic fields in micromagnetic computations. *Journal of Computational Physics*, 227(23):9913–9932, 2008.

[23] B Livshitz, A Boag, H N Bertram, and V Lomakin. Nonuniform grid algorithm for fast calculation of magnetostatic interactions in micromagnetics. *J. Appl. Phys.*, 105(7): 07D541, 2009. doi: 10.1063/1.3076048. URL `http://link.aip.org/link/?JAP/105/07D541/1`.

[24] A.V. Goncharov, G. Hrkac, J.S. Dean, and T. Schrefl. Kronecker product approximation of demagnetizing tensors for micromagnetics. *Journal of Computational Physics*, 229(7):2544–2549, April 2010. ISSN 0021-9991. doi: 10.1016/j.jcp.2009.12.004. URL `http://www.sciencedirect.com/science/article/B6WHY-4XX1606-3/2/29863d27f2c27f31627061a6cc252899`.

[25] L. Exl, W. Auzinger, S. Bance, M. Gusenbauer, F. Reichel, and T. Schrefl. Fast stray field computation on tensor grids. *Journal of Computational Physics*, 231(7): 2840–2850, 2012. URL `http://www.sciencedirect.com/science/article/pii/S0021999111007510`.

[26] L Exl, C Abert, N J Mauser, T Schrefl, H P Stimming, and D Suess. FFT-based Kronecker product approximation to micromagnetic long-range interactions. *(accepted for publication in Mathematical Models and Methods in Applied Sciences) arXiv preprint arXiv:1212.3509*, 2012.

[27] J. Keiner, S. Kunis, and D. Potts. Using nfft 3—a software library for various nonequispaced fast fourier transforms. *ACM Transactions on Mathematical Software (TOMS)*, 36 (4):19, 2009.

[28] E. Kritsikis, J.-C. Toussaint, O. Fruchart, H. Szambolics, and L. Buda-Prejbeanu. Fast computation of magnetostatic fields by nonuniform fast fourier transforms. *Applied Physics Letters*, 93(13):132508–132508, 2008.

[29] X. Brunotte, G. Meunier, and J.F. Imhoff. Finite element modeling of unbounded problems using transformations: a rigorous, powerful and easy solution. *IEEE Trans. Magn.*, 28(2):1663 –1666, mar 1992.

[30] D.R. Fredkin and T.R. Koehler. Hybrid method for computing demagnetizing fields. *IEEE Trans. Magn.*, 26(2):415 –417, mar 1990. ISSN 0018-9464. doi: 10.1109/20.106342.

[31] A. Knittel, M. Franchin, G. Bordignon, T. Fischbacher, S. Bending, and H. Fangohr. Compression of boundary element matrix in micromagnetic simulations. *J. Appl. Phys.*, 105(7):07D542, 2009. doi: 10.1063/1.3072032. URL `http://link.aip.org/link/?JAP/105/07D542/1`.

[32] C. J Garcia-Cervera and A. M Roma. Adaptive mesh refinement for micromagnetics simulations. *Magnetics, IEEE Transactions on*, 42(6):1648–1654, 2006.

[33] L Exl, S Bance, F Reichel, T Schrefl, H P Stimming, and N J Mauser. LaBonte's method revisited: An effective steepest descent method for micromagnetic energy minimization. *(accepted for publication in the Journal of Applied Physics) arXiv preprint arXiv:1309.5796*, 2013.

[34] A. E. LaBonte. Two-Dimensional Bloch-Type Domain Walls in Ferromagnetic Films. *Journal of Applied Physics*, 40(6):2450, 1969. ISSN 00218979. doi: 10.1063/1.1658014. URL `http://link.aip.org/link/?JAP/40/2450/1&Agg=doi`.

[35] K Kosavisutte and N Hayaslii. A Numerical Study of LaBonte ' s Iteration : An Approach to Acceleration. *IEEE Transactions on Magnetics*, 32(5):4243–4245, 1996.

[36] R Cohen, S-Y Lin, and M Luskin. Relaxation and gradient methods for molecular orientation in liquid crystals. *Computer Physics Communications*, 53(1):455–465, 1989.

[37] A Viallix, F Boileau, R Klein, JJ Niez, and P Baras. A new method for finite element calculation of micromagnetic problems. *Magnetics, IEEE Transactions on*, 24(6):2371–2373, 1988.

[38] M Luskin and L Ma. Numerical optimization of the micromagnetics energy. *Mathematics in Smart Materials*, pages 19–29, 1993.

[39] F Alouges, S Conti, A DeSimone, and Y Pokern. Energetics and switching of quasi-uniform states in small ferromagnetic particles. *ESAIM: Mathematical Modelling and Numerical Analysis*, 38(02):235–248, 2004.

[40] M J Donahue and D G Porter. Exchange energy formulations for 3d micromagnetics. *Physica B: Condensed Matter*, 343(1):177–183, 2004.

[41] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, New York, 1999. ISBN 9780387987934.

[42] Jonathan Barzilai and Jonathan M Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988.

[43] Donald Goldfarb, Zaiwen Wen, and Wotao Yin. A curvilinear search method for p - harmonic flows on spheres. *SIAM Journal on Imaging Sciences*, 2(1):84–109, 2009.

[44] Ignace Loris, Mario Bertero, Christine De Mol, Riccardo Zanella, and Luca Zanni. Accelerating gradient projection methods for l1-constrained signal recovery by steplength selection rules. *Applied and computational harmonic analysis*, 27(2):247–254, 2009.

[45] A Ramage and Eugene C Gartland Jr. A preconditioned nullspace method for liquid crystal director modeling. *SIAM Journal on Scientific Computing*, 35(1):B226–B247, 2013.

[46] T G Kolda and B W Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3): 455–500, August 2009. ISSN 0036-1445. doi: 10.1137/07070111X. URL http://csmr.ca.sandia.gov/~tgkolda/pubs/bibtgkfiles/TensorReview-preprint.pdf.

[47] W. Hackbusch. Tensor spaces and numerical tensor calculus. *Springer-Verlag Berlin Heidelberg*, 2012. doi: 10.1007/978-3-642-28027-6.

[48] G. Beylkin and M.J. Mohlenkamp. Numerical operator calculus in higher dimensions. *Proc Natl Acad Sci U S A*, 99(16):10246–10251, 2002.

[49] B W Bader and T G Kolda. Efficient MATLAB computations with sparse and factored tensors. *SIAM J. Sci. Comput.*, 30(1):205, 2008. ISSN 10648275. doi: 10.1137/060676489. URL http://link.aip.org/link/SJOCE3/v30/i1/p205/s1&Agg=doi.

[50] I.V. Oseledets. Tensor-train decomposition. *SIAM J. Sci. Comput.*, 33(5):2295–2317, 2011.

[51] E Acar, D M Dunlavy, and T G Kolda. A scalable optimization approach for fitting canonical tensor decompositions. *Journal of Chemometrics*, 25(2), jan 2011. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.172.5541.

[52] J. Håstad. Tensor rank is np-complete. *Journal of Algorithms*, 11(4), dec 1990.

[53] Vin de Silva and Lek-Heng Lim. Tensor rank and the Ill-Posedness of the best Low-Rank approximation problem. *SIAM Journal on Matrix Analysis and Applications*, 30: 1084–1127, September 2008. ISSN 0895-4798. doi: 10.1137/06066518X. URL http://portal.acm.org/citation.cfm?id=1461964.1461969.

[54] G. Tomasi and R. Bro. A comparison of algorithms for fitting the parafac model. *Computational Statistics & Data Analysis*, 50:1700–1734, 2006.

[55] M. Espig. Effziente Bestapproximation mittels Summen von Elementartensoren in hohen Dimensionen. Dissertation, Universität Leipzig, 2007.

[56] M Espig, L Grasedyck, and W Hackbusch. Black box low tensor-rank approximation using fiber-crosses. *Constructive approximation*, 30(3):557–597, 2009.

[57] W. Hackbusch and B. N. Khoromskij. Low-rank kronecker-product approximation to multi-dimensional nonlocal operators. part I. separable approximation of multi-variate functions. *Computing*, 76(3-4):177–202, 2006. ISSN 0010-485X. doi: 10.1007/s00607-005-0144-0. URL `http://www.springerlink.com/content/74v20851143034q1/`.

[58] E.E. Tyrtyshnikov. Tensor approximations of matrices generated ba asymptotically smooth functions. *Math. Sb.*, 194(6):147–160, 2003.

[59] L. Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM J. Matrix Anal. & Appl.*, 31(4), jan 2010.

[60] I V Oseledets, DV Savostyanov, and E E Tyrtyshnikov. Linear algebra for tensor problems. *Computing*, 85(3):169–188, 2009.

[61] D. Kressner. Low-rank tensor techniques for parametrized and high-dimensional linear algebra problems. lecture notes, The Max Planck Institute Magdeburg series of colloquia 2011, 2011.

[62] L.D. Lathauwer, B. De Moore, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4), 2000.

[63] P.M. Kroonenberg and J. De Leeuw. Principal component analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika*, 45(1), 1980.

[64] I V Oseledets, DV Savostianov, and E E Tyrtyshnikov. Tucker dimensionality reduction of three-dimensional arrays in linear time. *SIAM Journal on Matrix Analysis and Applications*, 30(3):939–956, 2008.

[65] C.A. Andersson and R. Bro. Improving the speed of multi-way algorithms: Part i. tucker3. *Chemometrics and Intelligent Laboratory Systems*, (42), 1998.

[66] L.D. Lathauwer, B. De Moore, and J. Vandewalle. On the best rank-1 and rank-(r1,r2,...,rn) approximation of higher- order tensors. *SIAM J. Matrix Anal. Appl.*, 21(4), 2000.

[67] P.K. Hopke, M. Leung, N. Li, and C. Navasca. Block tensor decomposition for source apportionment of air pollution. *arXiv:1110.4133*, 2011. doi: arXiv:1110.4133.

[68] I. Oseledets and E. Tyrtyshnikov. TT-cross approximation for multidimensional arrays. *Linear Algebra and its Applications*, 432, 2010.

[69] Julien Marot, Caroline Fossati, and Salah Bourennane. About advances in tensor data denoising methods. *EURASIP Journal on Advances in Signal Processing*, 2008(1):235357, 2008.

[70] J Jusélius and D Sundholm. Parallel implementation of a direct method for calculating electrostatic potentials. *The Journal of Chemical Physics*, 126(9):094101, 2007. ISSN 00219606. doi: 10.1063/1.2436880. URL `http://link.aip.org/link/JCPSA6/v126/i9/p094101/s1&Agg=doi`.

[71] http://suessco.com/simulations/, 2011. URL `http://suessco.com/simulations/`.

[72] T Schrefl, G Hrkac, S Bance, D Suess, O Ertl, and J Fidler. Numerical methods in micromagnetics (finite element method). *Handbook of magnetism and advanced magnetic materials*, 2007.

[73] Ernst Feldtkeller and Harry Thomas. Struktur und Energie von Blochlinien in dünnen ferromagnetischen Schichten. *Zeitschrift für Physik B Condensed Matter*, 4:8–14, 1965. ISSN 0722-3277. URL `http://dx.doi.org/10.1007/BF02423256`. 10.1007/BF02423256.

[74] R. P. Cowburn and M. E. Welland. Micromagnetics of the single-domain state of square ferromagnetic nanostructures. *Phys. Rev. B*, 58:9217–9226, Oct 1998. doi: 10.1103/PhysRevB.58.9217. URL `http://link.aps.org/doi/10.1103/PhysRevB.58.9217`.

[75] F. Steger. Numerical methods based on sinc and analytic functions. *Springer-Verlag*, 1993.

[76] D. Braess and W. Hackbusch. On the efficient computation of high-dimensional integrals and the approximation by exponential sums. In *Multiscale, nonlinear and adaptive approximation*, pages 39–74. Springer, 2009.

[77] C. Abert, G. Selke, B. Krüger, and A. Drews. A fast finite-difference method for micromagnetics using the magnetic scalar potential. *IEEE Trans. Magn.*, 48(3):1105 –1109, mar 2012. ISSN 0018-9464. doi: 10.1109/TMAG.2011.2172806.

[78] EA Muravleva and IV Oseledets. Fast low-rank solution of the poisson equation with application to the stokes problem. *arXiv preprint arXiv:1306.2150*, 2013.

[79] M Espig and W Hackbusch. A regularized Newton method for the efficient approximation of tensors represented in the canonical tensor format. *Numerische Mathematik*, 122(3): 489–525, 2012.

[80] A. Dutt and V. Rokhlin. Fast fourier transforms for nonequispaced data. *SIAM Journal on Scientific computing*, 14(6):1368–1393, 1993.

[81] G. Beylkin. On the fast fourier transform of functions with singularities. *Applied and Computational Harmonic Analysis*, 2(4):363–381, 1995.

[82] L Greengard and J-Y Lee. Accelerating the nonuniform fast fourier transform. *SIAM review*, 46(3):443–454, 2004.

[83] D. Potts, G. Steidl, and M. Tasche. Fast fourier transforms for nonequispaced data: A tutorial. In *Modern sampling theory*, pages 247–270. Springer, 2001.

[84] D. Potts and G. Steidl. Fast summation at nonequispaced knots by nfft. *SIAM Journal on Scientific Computing*, 24(6):2013–2037, 2003.

[85] S. Funken, D. Praetorius, and P. Wissgott. Efficient implementation of adaptive P1-FEM in MATLAB. *Comput. Methods Appl. Math.*, 11(4):460–490, 2011.

[86] J Ballani. *Fast evaluation of near-field boundary integrals using tensor approximations*. PhD thesis, Dissertation, Universität Leipzig, 2012.

[87] A. Elbel and G. Steidl. Fast fourier transforms for nonequispaced data,. In C.K. Chui and L.L. Schumaker, editors, *In: Approximation Theory IX,*, pages 39 –46, Vanderbuilt University Press,, 1998.

[88] D. Potts. Schnelle Fourier Transformationen für nichtäquidistante Daten und Anwendungen. Habilitationsschrift, Universität zu Lübeck, 2003.

[89] URL `http://www.mis.mpg.de/scicomp/EXP_SUM/1_sqrtx/`.

[90] J N Lyness and R Cools. A survey of numerical cubature over triangles. In *Proc. Symposia Appl. Math*, volume 48, pages 127–150, 1994.

[91] D. Potts, G. Steidl, and A. Nieslony. Fast convolution with radial kernels at nonequispaced knots. *Numerische Mathematik*, 98(2):329–351, 2004.

[92] Anders Logg, Kent-Andre Mardal, Garth N. Wells, et al. *Automated Solution of Differential Equations by the Finite Element Method*. Springer, 2012. ISBN 978-3-642-23098-1. doi: 10.1007/978-3-642-23099-8.

[93] Michael Pippig. PFFT: An extension of FFTW to massively parallel architectures. *SIAM Journal on Scientific Computing*, 35(3):C213–C236, 2013.

[94] H Hassanieh, P Indyk, D Katabi, and E Price. Nearly optimal sparse fourier transform. In *Proceedings of the 44th symposium on Theory of Computing*, pages 563–578. ACM, 2012.

[95] URL `http://groups.csail.mit.edu/netmit/sFFT/index.html`.

[96] I Oseledets and E Tyrtyshnikov. TT-cross approximation for multidimensional arrays. *Linear Algebra and its Applications*, 432(1):70–88, 2010.

[97] D Savostyanov and I Oseledets. Fast adaptive interpolation of multi-dimensional arrays in tensor train format. In *Multidimensional (nD) Systems (nDs), 2011 7th International Workshop on*, pages 1–8. IEEE, 2011.

– ◇ –

Dieser Mythos ist tragisch, weil sein Held bewußt ist. Worin bestünde tatsächlich seine Strafe, wenn ihm bei jedem Schritt die Hoffnung auf Erfolg neue Kraft gäbe? Heutzutage arbeitet der Werktätige sein Leben lang unter gleichen Bedingungen, und sein Schicksal ist genauso absurd. Tragisch ist es aber nur in den wenigen Augenblicken, in denen der Arbeiter bewußt wird. Sisyphos, der ohnmächtige und rebellische Prolet der Götter, kennt das ganze Ausmaß seiner unseligen Lage: über sie denkt er während des Abstiegs nach. Das Wissen, das seine eigentliche Qual bewirken sollte, vollendet gleichzeitig seinen Sieg.

Albert Camus, *Der Mythos des Sisyphos*

Acknowledgments

When I started working in Dieter's and Thomas' project I just had finished my maths thesis. At this time, I was rather unaware of how science (business) works. It is mostly due to my supervisor, Thomas Schrefl, that I now know a little more. I want to thank him for giving me the opportunity to participate on many conferences and workshops. Moreover, his tremendous knowledge in the relevant literature helped me a lot. I also want to appreciate his experienced advises concerning the topics around producing papers and getting them published. He really understands this business. Amazingly enough, Tom is also an excellent hands-on scientist, who earns every single credit point in his good reputation as a scientist.

Of course, I do not want to forget to thank Winfried Auzinger. Especially in the first year of my project, he supported me with his mathematical insight and sympathetic ear. Many discussions at the Vienna UT were helpful and interesting for me. Not least because of Winfried's sympathy and instinct of sensing problems of his young students concerning their development in the academic area.

I want to thank my colleagues from the FH St. Pölten, Tom, Frank, Simon, Gusi, Hari, Fisch and Alex. It was great fun and also instructive to work in such a lively and talented group. I found that it is important to have a surrounding which gives you the feeling of being a part of something. Especially, I want to appreciate Frank's organizational efforts concerning all the 'side-activities' at the Fachhochschule and his help with computer issues. Also, Thomas' liberal leadership style in our research group gave me the freedom to cope with many problems on my own.

Equally, I want to give thanks to my colleagues form the University of Vienna, especially Norbert Mauser and Hans Peter Stimming, for their help with our joint papers and the financial support. The latter gives me the opportunity to stay a few months within the environment of the Wolfgang Pauli Institut. This is a very valuable experience.

Many conferences and visits were great fun, especially because of the nice companionship of all my colleagues. I remember a great research visit in Hamburg thanks to Claas Abert and André Drews. Especially, I want to thank my generous hosts Claas and Hanna. During this visit

and Claas' visits in Wien I enjoyed many professional discussions with him and, of course, also the distractions from work.

Nevertheless, working and writing on my thesis was sometimes a very hard and lonesome time, pondering on something on my own. For this very reason, it was extremely important for me to rely on my best friends and family if things turned out to be too much. Many thanks to my girlfriend Tina, who supported me greatly and gave me fresh power where I needed it.

An dieser Stelle möchte ich meiner Familie speziellen Dank aussprechen, besonders meinen Eltern, welche mir das Rüstzeug, die notwendigen Freiheiten und die Verantwortung gaben, eigene Werte und Persönlichkeit entwickeln zu können.

Finally, I want to thank Dieter Suess and Thomas Schrefl for their efforts to apply for the project in which I could participate. Of course, because of the essential role for the project, the financial support by the Austrian Science Fund (FWF) SFB ViCoM (F4112-N13) is also gratefully acknowledged.

Lukas Exl
Wien, Januar 2014

Selected Publications

[1] L Exl, S Bance, F Reichel, T Schrefl, H P Stimming, and N J Mauser. LaBonte's method revisited: An effective steepest descent method for micromagnetic energy minimization. *(accepted for publication in the Journal of Applied Physics) arXiv preprint arXiv:1309.5796*, 2013.

[2] L. Exl and T. Schrefl, "Non-uniform FFT for the finite element computation of the micromagnetic scalar potential," *arXiv preprint arXiv:1305.3162*, 2013.

[3] L Exl, C Abert, N J Mauser, T Schrefl, H P Stimming, and D Suess. FFT-based Kronecker product approximation to micromagnetic long-range interactions. *(accepted for publication in Mathematical Models and Methods in Applied Sciences) arXiv preprint arXiv:1212.3509*, 2012.

[4] L. Exl, W. Auzinger, S. Bance, M. Gusenbauer, F. Reichel, and T. Schrefl. Fast stray field computation on tensor grids. *Journal of Computational Physics*, 231(7):2840–2850, 2012. http://www.sciencedirect.com/science/article/pii/S0021999111007510.

[5] C Abert, L Exl, G Selke, A Drews, and T Schrefl. Numerical methods for the stray-field calculation: A comparison of recently developed algorithms. *Journal of Magnetism and Magnetic Materials*, 326:176–185, 2012.

[6] L. Exl, A-posteriori Fehlerschätzung für Differentialgleichungen höherer Ordnung, *Master Thesis, TU Wien,* http://kitt.ub.tuwien.ac.at, 2010.

Conferences, Workshops, Talks and Posters

11/2013	7th Workshop ViCoM, Stadtschlaining, Austria
11/2013	58th Annual Conference on Magnetism and Magnetic Materials (MMM 2013), Denver (Col), USA Poster presentation: *'La Bonte's method revisited: An effective steepest descent method for micromagnetic energy minimization'*
09/2013	CECAM Workshop (*'Fast Methods for Long Range Interactions in Complex Particle Systems'*), FZ Jülich, Germany Poster presentation: *'Non-uniform FFT for the finite element computation of the magnetic scalar potential'*
04/2013	6th Workshop ViCoM, Vienna, Austria Talk: *'Fast convolution method for non-uniform data'*
11/2012	5th Workshop ViCoM, Stadtschlaining, Austria
08/2012	Joint European Magnetic Symposia (JEMS 2012), Parma, Italy Poster presentation: *'Micromagnetic energy minimization for low-rank tensor magnetization'*
07/2012	International Summer School (*'High Performance Computing'*), Waidhofen/Ypps, Austria
06/2012	Research visit (1 month) at University of Hamburg (GER), Group Nanostructure Physics (Institute of Applied Physics) Group Arbeitsbereich Technische Informatik Systeme (Department of Informatics)
05/2012	International Workshop on Advanced Micromagnetics (IWAM 2012),

	UCSD, San Diego (CA), USA Talk: *’Micromagnetic energy minimization for low-rank tensor magnetization’*
04/2012	4th Workshop ViCoM, Vienna, Austria Talk: *’Micromagnetic energy minimization for low-rank tensor magnetization’*
02/2012	7th Vienna Conference on Mathematical Modelling (MATHMOD VIENNA 2012) Talk: *’Low-Rank Tensors as possible Tool for Micromagnetics’*
11/2011	5th International Conference on Advanced COmputational Methods in ENgineering (ACOMEN 2011) Liège, Belgium, 14-17 November 2011 Talk: *’Magnetostatic Field Computation on Tensor Grids’*
10/2011	3rd Workshop ViCoM, Stadtschlaining, Austria
05/2011	8th International Symposium on Hysteresis Modelling and Micromagnetics (HMM), Levico (Trento), Italy Talk: *’Fast stray field computation by tensor representation’*
04/2011	2nd Workshop ViCoM, Vienna, Austria Talk: *’Stray field computation on tensor grids’*

Curriculum Vitae

DIPL.-ING. (M.SC.)
LUKAS SEBASTIAN EXL

Friedrich-Kaiser Gasse 26/23, A - 1160 Vienna

Date of Birth	Jan 12, 1985
Place of Birth	Vienna, Austria
Citizenship	Austria

Education

since 10/2010	PhD student Supervisor: Univ.Prof. FH-Prof. Dipl.-Ing. Dr. Thomas Schrefl Vienna University of Technology, Faculty of Physics
06/2010	M.Sc. in Mathematics (with honors) Supervisor: Ao.Univ.Prof. Dipl.-Ing. Dr. Winfried Auzinger, Institute for Analysis and Scientific Computing, E101, Vienna University of Technology
03/2008-06/2010	Studies of Mathematics for Natural Sciences, Vienna University of Technology
10/2004-02/2008	Studies of Technical Mathematics, Vienna University of Technology
10/2003-07/2004	Military service (corpsman), Military Academy Wr. Neustadt, Lower Austria
09/1995-06/2003	Abitur grammar school, Berndorf, Lower Austria

Scientific Employments and Projects

since 12/2013	University of Vienna, Department of Mathematics
since 12/2013	FWF SFB F4112-N13, Part6 *'Dynamical Correlated Systems'*
01/2011 - 11/2013	University of Applied Sciences St. Pölten GmbH, Department of Technology
01/2011 - 11/2013	FWF SFB F4112-N13, Part12 *'Multi-Scale Simulations of Magnetic Nanostructures'*

Lebenslauf

Dipl.-Ing. (M.Sc.)
Lukas Sebastian Exl

Friedrich-Kaiser Gasse 26/23, A - 1160 Wien

Geburtstag	Jan 12, 1985
Geburtsort	Wien, Österreich
Staatsbürgerschaft	Österreich

Ausbildung

seit 10/2010	Doktorand Betreuer: Univ.Prof. FH-Prof. Dipl.-Ing. Dr. Thomas Schrefl Technische Universität, Fakultät für Physik
06/2010	Diplom in Mathematik (mit Auszeichnung) Betreuer: Ao.Univ.Prof. Dipl.-Ing. Dr. Winfried Auzinger, Institut für Analysis and Scientific Computing, E101, Technische Universität Wien
03/2008-06/2010	Studium der Mathematik in den Naturwissenschaften, Technische Universität Wien
10/2004-02/2008	Studium der Technischen Mathematik, Technische Universität Wien
10/2003-07/2004	Präsenzdienst (Sanitäter), Militär Akademie Wr. Neustadt, Niederösterreich
09/1995-06/2003	Realgymnasium (mit Matura abgeschlossen), Berndorf, Niederösterreich

Wissenschaftliche Anstellungen und Projekte

seit 12/2013	Universität Wien, Fakultät für Mathematik
seit 12/2013	FWF SFB F4112-N13, Teil6 *'Dynamical Correlated Systems'*
01/2011 - 11/2013	Fachhochschule St. Pölten GmbH, Department für Technologie
01/2011 - 11/2013	FWF SFB F4112-N13, Teil12 *'Multi-Scale Simulations of Magnetic Nanostructures'*