universität wien

# MASTERARBEIT | MASTER'S THESIS

Titel | Title

The minimization of Brown's energy bounds for the magnetostatic energy with the Finite Element Method

verfasst von | submitted by

Sebastian Schmuckermayr BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2026

| | |
|---|---|
| Studienkennzahl lt. Studienblatt \| Degree programme code as it appears on the student record sheet: | UA 066 821 |
| Studienrichtung lt. Studienblatt \| Degree programme as it appears on the student record sheet: | Masterstudium Mathematik |
| Betreut von \| Supervisor: | Dipl.-Ing. Dr.techn. Lukas Exl Privatdoz. |

# Abstract

Micromagnetics as the transition between classical continuum theory and quantum mechanical frameworks provides an important tool for analyzing static magnetic energy equations. This thesis addresses the computational aspects of micromagnetics, especially the minimization of the total Gibbs free energy in three dimensions with respect to a magnetization under a nonlinear unit norm constraint within the magnetic domain. Focus is on the optimization through classical numerical approaches based on mesh-dependent discretization methods, in particular the Finite Element Method (FEM). A central aspect is the replacement of parts of the total energy, the magnetostatic stray field energy, by an upper bound proposed by Brown, to simplify the optimization by reducing non-locality. This reformulation introduces an independent vector potential, resulting in an objective function depending on two three-dimensional variables.
In this thesis, a detailed overview of the P1-FEM is given, including the discretization of the energy terms and the upper bound with respect to both the magnetization and the vector potential. The resulting linear systems are solved with micromagnetic adaptions of the penalty method, the augmented Lagrangian method and projected conjugate gradient methods. Particular emphasis is placed on the novel approach of comparing a joint minimization of the objective with respect to both variables with an alternating optimization procedure. The performance of these methods is tested on various benchmark problems as well as on a variation of the NIST $\mu$MAG standard problem #3.

# Zusammenfassung

Mikromagnetismus, als eine Brücke zwischen klassischer Kontinuumstheorie und quantenmechanischen Ansätzen, stellt ein wichtiges Werkzeug zur Analyse statischer magnetischer Energiegleichungen dar. Diese Arbeit befasst sich mit den rechnerischen Aspekten des Mikromagnetismus, vor allem mit der Minimierung der Gibbs'schen freien Energie in drei Dimensionen bezüglich der Magnetisierung unter Berücksichtigung einer nicht-linearen Einheitsnormbedingung innerhalb des magnetischen Gebiets.
Der Fokus dieser Arbeit liegt auf der Optimierung mittels klassischer numerischer Verfahren, insbesondere auf gitterbasierten Diskretisierungsmethoden, wie der Finite Elemente Methode (FEM). Ein zentraler Aspekt besteht darin, Teile der totalen Energie, die sogenannte magnetostatische Streufeldenergie, durch eine von Brown eingeführte obere Schranke zu ersetzen, wodurch nichtlokale Effekte reduziert werden und die Optimierung vereinfacht wird. Diese Formulierung führt ein unabhängiges Vektorpotential ein, sodass die Zielfunktion von zwei dreidimensionalen Variablen abhängt.
Eine detaillierte Darstellung der P1-FEM wird gegeben, einschließlich der Diskretisierung der Energieterme und der oberen Schranke sowohl in Bezug auf die Magnetisierung als auch auf das Vektorpotential. Die daraus resultierenden linearen Gleichungssysteme werden mit Varianten von Penalty-Verfahren, Augmented-Lagrange-Methoden und projizierten Konjugierte Gradienten Verfahren gelöst.
Ein besonderer Schwerpunkt liegt auf dem Vergleich zwischen einer alternierenden Optimierungsstrategie und dem Ansatz, das Zielfunktional simultan bezüglich beider Variablen zu minimieren. Beide Methoden werden anhand von Benchmark-Problemen sowie einer Variante des NIST $\mu$MAG Standardproblem #3 getestet.

# Acknowledgment

First and foremost, I would like to express my sincere gratitude to Priv.-Doz. Dr. Lukas Exl for his continuous encouragement and for the time and care he invested in supervising this thesis. His thoughtful feedback and willingness to engage with my questions greatly contributed to my understanding of the subject. Furthermore, I also want to thank Dr. Sebastian Schaffer for his constant input and his constructive feedback throughout this work. Lastly, I thank my family and especially my girlfriend for their unwavering support during my entire studies and the writing of this thesis.

In addition, I gratefully acknowledge financial support by the Austrian Science Foundation (FWF) through the projects of Doz. Dr. Exl at WPI and Univ. Wien: 'Data-driven Reduced Order Approaches for Micromagnetism' (Data-ROAM) (Grant-DOI: 10.55776/PAT7615923) at WPI, 'Design of Nanocomposite Magnets by Machine Learning' (DeNaMML)(Grant-DOI: 10.55776/P35413) at Univ. Wien research platform MMM 'Mathematics–Magnetism–Materials'.

## Declaration

The large language model *ChatGPT* [1] was used in this work as an assistive tool for linguistic enhancement, as English is not the native language of the author. The author has reviewed and edited all AI-generated content and takes full responsibility for the claims and references.

# Contents

# 1 Introduction

Magnetism plays a central role in modern technology. The construction of electric motors, wind turbines, storage systems and different kinds of generators has increased interest in modeling magnetic structures over the last decades. In particular, technological advances such as renewable energy, smartphones, air conditioning systems or electric cars and bikes further intensified the need for high performance permanent magnets. In this context, high performance refers to magnets which can create a strong magnetic field as well as have a high resistance to external, opposing magnetic fields. These properties are fulfilled by a variety of rare-earth permanent magnets, such as neodymium-iron-bor (NdFeB) or samarium-cobalt (SmCo) compounds, leading to them dominating the global market [2]. Especially, the demand for NdFeB-magnets grows rapidly as the predicted annual growth rates between 2019 and 2022, depicted in Figure 1.1, show. In particular,

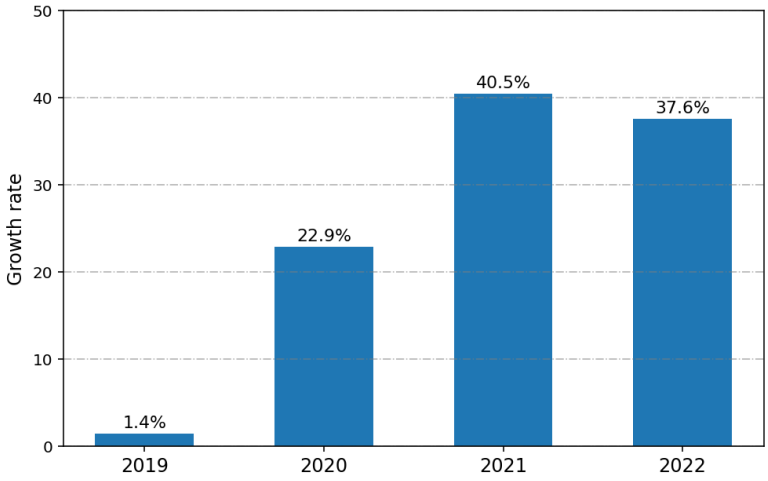


Figure 1.1: Annual growth rate of the demand for NdFeB magnets worldwide (in percent) in the years 2019-2022 (the values from 2020-2022 are based on a forecast) [3].

the automotive and the wind energy sector are the main driving forces for this trend as Figure 1.2 highlights. It shows a comparison of the demand for rare earth permanent magnets in tons in the years 2010 and 2015 with a forecast for 2025. Whereas the need for permanent magnets in wind turbines was expected to triple over the last ten years, the demand in the automotive industry was even predicted to be five times higher than

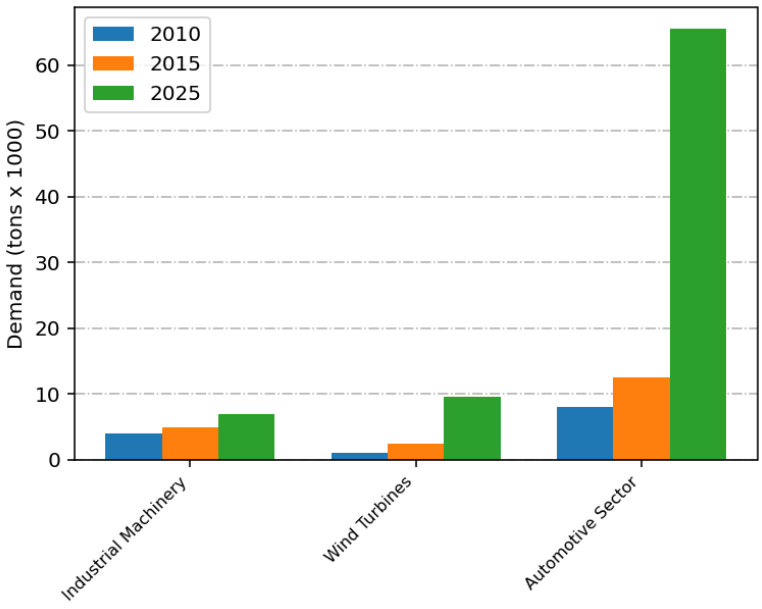


Figure 1.2: Comparison of the global demand for rare-earth permanent magnets in the years 2010, 2015 and 2025 in different economic sectors. The values for 2025 are based on a forecast [4], [5], [6].

in 2015. However, rare-earth elements are highly toxic, and their mining and extraction pose significant risks to both ecosystems and human health. In addition, China supplies over 80% of the global rare-earth market, which introduces potential geopolitical vulnerabilities [7]. This motivates the search for rare-earth reduced or even rare-earth free high performing magnets. To do so, a framework is needed to measure and observe the stability of magnets of a given structure and material. This is provided by the theory of micromagnetism.

Micromagnetism combines classical continuum theory with a quantum mechanical approach and is therefore a useful tool for modeling magnetic fields and the behavior of the magnetization within materials [8]. In micromagnetism one observes magnets on a scale of nanometers to micrometers. This has several advantages. On the one hand, it is a large enough scale to neglect discrete atom spins within the magnet and replace them with continuous functions. As a result, it is ensured, that the numerical computations can be done in reasonable time. On the other hand, macromagnetic scales governed by Maxwell's equations neglect interactions within the magnet, which are of quantum mechanical nature. These effects are taken into account with a length scale small enough, as in the case of micromagnetics [9]. A part of micromagnetics is the field of magnetostatics which covers the study of magnetic fields with steady currents. Here, the magnetic state of a ferromagnet is described by a classical magnetization vector field [10]. Consequently, it is possible to model the magnetization dynamics by analytical

expressions.
For example, the stable/metastable state of a magnet is described as the minimum of the total Gibbs free energy

$$E(\mathbf{m}) = E_d(\mathbf{m}) + E_a(\mathbf{m}) + E_{ex}(\mathbf{m}) + E_{zee}(\mathbf{m}) \tag{1.1}$$

under the constraint $\|\mathbf{m}\| = 1$ [11]. This total energy will be observed in more detail in the course of this thesis, but in general $E(\mathbf{m})$ depends on the shape and the material of the magnet. Thus, minimizing $E$ makes it possible to predict the behavior of this given magnet, making it an effective tool in the search of high performing magnets.

One challenge in minimizing (1.1) arises from the so-called *magnetostatic (stray field) energy* $E_d$. This energy exhibits a highly nonlocal behavior [12]. Therefore, in [13], W.F. Brown proposed sharp lower and upper bounds for the magnetostatic energy, which reduce the non-locality to purely local interactions by introducing new field variables [14]. While the lower bound depends on a scalar potential which generates the stray field, the upper bound involves a vector potential generating the magnetic induction and is of greater interest to this thesis.
Whereas it is an already established method to use the scalar potential for optimization, the key aspect of this work is observing the performance of the upper bound replacing the magnetostatic energy during the minimization process, leading to a min-min problem in both the magnetization and the vector potential. An additional focus of this thesis is on comparing an alternating minimization approach with a joint optimization strategy for this problem.

In recent years, data-driven methods, such as unsupervised Physics-Informed Neural Networks (PINNs) [15] have been used to minimize expression (1.1). These methods have the advantage of being flexible during the training process and are efficient in terms of conditional parameters [16]. However, classical numerical schemes remain highly relevant, both as robust, interpretable baselines and as a tool for benchmarking modern algorithms. Therefore, this work adopts a more traditional perspective, based on using Finite Element Methods (FEM). Hereby, the quality of such computations strongly depends on the right discretization of the domain as well as on efficient methods to solve the resulting linear systems. To this end, this thesis also covers iterative algebraic solvers specifically tailored to the aims of micromagnetics. The main objective of this paper is to demonstrate that this classical approach provides an efficient and reliable foundation for more modern algorithms.

The thesis is structured as follows:
In Chapter 2 the theoretical foundations of micromagnetics are explained and the essential energy equations are derived. In Chapter 3 the mathematical background of the FEM is elaborated. This includes handling variational formulations and developing discrete function spaces on a meshed domain. Chapter 4 deals with the application of the previously described theory as the discretization of the domain and the different energy terms is explained. Methods for solving the established linear systems are discussed in

Chapter 5 as well as various ways to deal with the nonlinear constraint occurring in the problem. Finally, in Chapter 6 these methods are evaluated by performing energy minimization in the context of different subproblems. The results of these calculations are presented in this section. In the Appendix explicit computations of analytical examples and a description of the used Python libraries including the most important code snippets can be found.

# 2 Theoretical Foundations of Micromagnetism

Micromagnetism as the attempt to connect the macroscopic Maxwell theory of electromagnetic fields with microscopic quantum mechanical models made its first appearance in the 1930s with the works of Landau and Lifshitz [17], who derived continuum expressions of various energy terms [18].
The numerical study of micromagnetism has begun in the 1950s when W. F. Brown formalized this concept in his works [19], followed by papers of Kittel, Stoner–Wohlfarth, Néel, Aharoni, Strikman and Treves [18]. Their theory describes magnetic structures with a continuum approach, assigning magnetization vectors to different materials [20].
A key benefit of micromagnetism is that it operates on a length scale exceeding the range of nanometers. Thus, it is possible to omit the use of discrete spins to describe the ferromagnetic state. Instead, one can use continuous vector fields. This approach, which typically is in the range of nanometers to micrometers, ensures that the length scale is sufficiently large to keep the computations in reasonable time but also small enough to model core physical concepts from quantum mechanics [9].
The primary assumption is to describe the state of a magnetic material by a continuous vector field $\mathbf{M} : \Omega \to \mathbb{R}^3$, the *magnetization*. Here, $\Omega \subset \mathbb{R}^3$ denotes the magnetic domain. The output of the function is given in SI-units $[\mathbf{M}] = Am^{-1}$ and the length of the magnetization is constant at a given temperature, $|\mathbf{M}| = M_s > 0$, the so-called *saturation magnetization* [21]. For the remainder of this work we will always assume a constant temperature, so a change of length of $\mathbf{M}$ is not permitted. This allows us to consider the normalized magnetization

$$\mathbf{m}(\mathbf{x}) \coloneqq \frac{\mathbf{M}(\mathbf{x})}{M_s}.$$

This relation only holds inside of $\Omega$, outside the magnetic domain we set $\mathbf{m} \equiv \mathbf{0}$ [14].

## 2.1 The Gibbs Free Energy

A key concept in static micromagnetic simulations is the total Gibbs free energy of a magnetic domain given by [11]:

$$E(\mathbf{m}) = E_d(\mathbf{m}) + E_{zee}(\mathbf{m}) + E_a(\mathbf{m}) + E_{ex}(\mathbf{m}). \tag{2.1}$$

The Gibbs free energy describes the dependence of the magnetization on the material and is given in SI-units $[E] = J$. It consists of four fundamental energy terms, the magnetostatic self-energy or stray field energy $E_d$, which quantifies the energy stored in the magnetic field with respect to the material's magnetization, the Zeeman energy $E_{zee}$ modeling the interaction of the magnet with an external field, the magnetocrystalline anisotropy energy $E_a$ describing the work to magnetize in certain directions and the exchange energy $E_{ex}$ which is a quantum mechanical measure to describe the relations between particles [9]. A more thorough description of the energy terms is given in the following sections. For numerical computations one uses the reduced form of the energy $e = \frac{1}{M_s^2 \mu_0} E$ with $\mu_0 = 4\pi 10^{-7}$ Tm/A being the *vacuum permeability* [15]. The reduced energy is given in units of $[e] = m^3$.
The main objective of the numerical computations in this work is finding a minimum of this reduced energy in terms of the magnetization, which leads to an equilibrium configuration for $\mathbf{m}$. One can model the evolution of the magnetic state and also create hysteresis loops by starting at an initial magnetization and carry out the minimization process [11]. By minimizing different energy configurations arising from various material compositions, one can model the stability of the corresponding magnet, which delivers important theoretical insight in the search for high performing magnets.
Hereby, a significant challenge is the nonlinear unit norm constraint

$$\|\mathbf{m}\|^2 = 1 \quad \text{in } \Omega,$$

which has to be considered during optimization.
In the following sections, we will describe the contributing energy terms in more details. For this we refer to [9] and [15].

## 2.2 The Exchange Energy

We start by deriving the exchange energy, which originates from quantum mechanical laws. It is based on the Pauli exclusion principle which states that two electrons must have different spins to be at the same place. However, in the context of ferromagnets two particles with parallel spin tend to spatially separate, thereby lowering the electrostatic energy leads to the parallel state being the preferred one. The following computations can be found in more detail in [9].
Let $\Omega \subset \mathbb{R}^3$ be the domain of interest and let $i$ and $j$ represent two neighboring atoms within $\Omega$. Then, the quantum mechanical formulation of the exchange energy between the two spins $\mathbf{S}_i$ and $\mathbf{S}_j$ can be expressed by a modification of the Heisenberg exchange Hamiltonian [22]

$$E_{ex}(i,j) = -2J_{ij}\mathbf{S}_i \cdot \mathbf{S}_j,$$

where $J_{ij}$ denotes the exchange integral between the atoms $i$ and $j$. For highly symmetric crystal structures such as cubic lattices, all nearest-neighbor interactions are equivalent, and thus $J_{ij}$ can be replaced by a constant $J$. In the micromagnetic setting this model is

simplified by replacing the spins with classical vectors, in particular with the unit vector of the magnetic moment $\mathbf{m}_i$, which is the unit vector in the direction $-\mathbf{S}_i$.
To generalize further, we introduce the continuous magnetization $\mathbf{m} : \Omega \to \mathbb{R}^3$, which fulfills at every position $\mathbf{x}_i$ of an atom $i$, that

$$\mathbf{m}(\mathbf{x}_i) = -\mathbf{S}_i.$$

After expanding $\mathbf{m}$ in a Taylor series around such a point $\mathbf{x}_i$ and summing over all atoms within a unit cell around $i$, one obtains in the limit the integral formulation of the total exchange energy

$$E_{ex}(\mathbf{m}) = \int_\Omega A_{ex}[\|\nabla m_x\|^2 + \|\nabla m_y\|^2 + \|\nabla m_z\|^2]\, d\mathbf{x} = \int_\Omega A_{ex}\|\nabla \mathbf{m}\|_F^2\, d\mathbf{x}.$$

Here, $\mathbf{m} = (m_x, m_y, m_z)^T$ and $A_{ex}$ is the *exchange constant*, fulfilling

$$A_{ex} = \frac{JS^2}{a} n,$$

with $a$ being the interatomic distance, $n$ the number of atoms per unit cell and $S = |\mathbf{S}_i|$ the spin quantum number, which is determined by the electronic structure of the ferromagnetic material.
By scaling with a reference element we arrive at the reduced exchange energy [15]

$$e_{ex}(\mathbf{m}) = \int_\Omega \tilde{A}_{ex}\|\nabla \mathbf{m}\|_F^2\, d\mathbf{x}, \tag{2.2}$$

with $\tilde{A}_{ex} \coloneqq A_{ex}/\mu_0 M_s^2$. This approach reduces the complexity of the model to a great extent, while retaining its essential characteristics. Nonetheless, it has its limitations. The formulation is often too simple, because it assumes a homogeneous and continuous medium, which is sometimes not true. Especially when the interactions happen on an atomic level the underlying structure cannot be described accurately enough [23]. Despite these simplifications, this model will be sufficiently accurate for our calculations.

## 2.3 The Magnetocrystalline Anisotropy Energy

Next, we describe the magnetocrystalline anisotropy energy, an intrinsic property of the material of $\Omega$. This energy arises from the crystal-like structure of the magnet, which determines the arrangement of the electronic orbits. The spin-orbital interactions force the spins to align with certain axes - the so-called *easy axes* [24]. This phenomenon propagates to the magnetization and the energy gets larger, the further away the direction of the vector $\mathbf{m}$ is from the easy axes. Consequently, to minimize the anisotropy energy, one likes to align the magnetization with the one of these preferred directions.
In cubic systems, the anisotropy energy is influenced by the directional cosines of the

normalized magnetization with respect to the Cartesian axes of the lattice. Let $\mathbf{u}_x, \mathbf{u}_y, \mathbf{u}_z$ denote the unit vectors along these axes and define

$$\alpha_i := \mathbf{m} \cdot \mathbf{u}_i \quad i = x, y, z$$

as the directional cosines. Then the anisotropy energy is given by [24]

$$E_a(\mathbf{m}) = K_0 + K_1(\alpha_x^2\alpha_y^2 + \alpha_y^2\alpha_z^2 + \alpha_x^2\alpha_z^2) + K_2\alpha_x^2\alpha_y^2\alpha_z^2.$$

The values $K_0, K_1$ and $K_2$ are the magnetocrystalline anisotropy constants and depend in general on temperature [9]; however, under the assumption of constant temperature, they are treated as constants.
In this work only a special case of the energy above is considered, namely the *uniaxial anisotropy*. Here, there is only one single easy axis and we consider the angle $\theta$ between the magnetization and this axis. Without loss of generality, we choose the z-axis as the easy axis and can write the energy as [9]

$$E_a(\mathbf{m}) = K_0 + K_1 \sin^2(\theta) + K_2 \sin^4(\theta).$$

We can simplify this expression by using $\sin^2(\theta) = 1 - (\mathbf{m} \cdot \mathbf{u}_z)^2$, omitting the constant $K_0$, truncating the sum earlier and integrating over the whole domain to arrive at

$$E_a(\mathbf{m}) = \int_\Omega K_1(1 - (\mathbf{m} \cdot \mathbf{u}_z)^2)\, d\mathbf{x}.$$

Finally, in dimensionless form the uniaxial anisotropy energy can be written as

$$e_a(\mathbf{m}) = \int_\Omega Q(1 - (\mathbf{m} \cdot \mathbf{u}_z)^2)\, d\mathbf{x}, \tag{2.3}$$

with $Q = K_1/\mu_0 M_s^2$ being the reduced anisotropy constant.

## 2.4 The Zeeman Energy

The Zeeman energy describes the behavior of the magnet when an external magnetic field $\mathbf{H}_{ext}$ is applied. It is stated as [25]

$$E_{zee}(\mathbf{M}) = -\mu_0 \int_\Omega \mathbf{M} \cdot \mathbf{H}_{ext}\, d\mathbf{x}.$$

It follows that

$$E_{zee}(\mathbf{m}) = -\mu_0 M_s^2 \int_\Omega \mathbf{m} \cdot \mathbf{h}_{ext}\, d\mathbf{x} =: -\mu_0 M_s^2 e_{zee}(\mathbf{m}),$$

with a dimensionless external energy $\mathbf{h}_{ext} = \mathbf{H}_{ext}/M_s$.
The Zeeman energy is minimized when the magnetization aligns with the external field, which means this energy is in contrast to the other energy terms and is therefore responsible for multiple complex dynamics [23]. However, in the calculations of this work the external field will almost always be neglected which further simplifies the model.

## 2.5 The Magnetostatic Self-Energy

The magnetostatic self-energy or demagnetizing energy is formed from a stray field $\mathbf{H}_d$ which is generated by the magnetization itself [23]. $\mathbf{H}_d$ is therefore a function of $\mathbf{M}$ and also scale invariant. However, for the sake of a simpler notation we will still write $\mathbf{H}_d = \mathbf{H}_d(\mathbf{M})$. The field is measured in units of $A/m$. The associated energy depends on material properties like the shape or the magnetic configuration of the magnetic object. It is given by the interaction of each local magnetic moment $\boldsymbol{\mu}_i$ at every position $\mathbf{x}_i$ with the stray field [9]:

$$E_d(\mathbf{M}) = -\frac{\mu_0}{2} \sum_i \boldsymbol{\mu}_i \cdot \mathbf{H}_d(\mathbf{M}(\mathbf{x}_i)).$$

Since pairs of atoms are considered, a factor $1/2$ is needed to avoid double-counting. The magnetization can be expressed as the integral over all local magnetic moments [26]

$$\mathbf{M} = \int_\Omega \boldsymbol{\mu}_i \, d\mathbf{x},$$

so in the limit the magnetostatic self-energy is

$$E_d(\mathbf{M}) = -\frac{\mu_0}{2} \int_\Omega \mathbf{M} \cdot \mathbf{H}_d \, d\mathbf{x}. \tag{2.4}$$

In dimensionless form the stray field can be written as

$$\mathbf{h}_d = \frac{\mathbf{H}_d}{M_s}, \tag{2.5}$$

so from (2.4) it follows

$$E_d(\mathbf{m}) = -\frac{M_s^2 \mu_0}{2} \int_\Omega \mathbf{m} \cdot \mathbf{h}_d \, d\mathbf{x} =: \frac{M_s^2 \mu_0}{2} e_d(\mathbf{m}), \tag{2.6}$$

with

$$e_d(\mathbf{m}) := \int_\Omega -\mathbf{m} \cdot \mathbf{h}_d \, d\mathbf{x}.$$

Combining all the energies calculated above, the Gibbs free energy in dimensionless form reads as

$$\begin{aligned} e(\mathbf{m}) = \frac{1}{2} \int_\Omega -\mathbf{m} \cdot \mathbf{h}_d \, d\mathbf{x} + \int_\Omega -\mathbf{m} \cdot \mathbf{h}_{ext} \, d\mathbf{x} \\ + \int_\Omega Q(1 - (\mathbf{m} \cdot \mathbf{u}_z)^2) \, d\mathbf{x} + \int_\Omega \tilde{A}_{ex} \|\nabla \mathbf{m}\|_F^2 \, d\mathbf{x}. \end{aligned} \tag{2.7}$$

## 2.6 Calculation of the Stray Field

One of the most important and challenging numerical tasks in micromagnetics is the computation of the stray field $\mathbf{h}_d$. Thus, in this section the physical derivation of this vector field is examined, following the discussions in [15] and [27].
We start with the electrostatic Maxwell's equations, assuming vanishing currents:

$$\nabla \cdot \mathbf{b}_d = 0, \tag{2.8}$$
$$\nabla \times \mathbf{h}_d = 0. \tag{2.9}$$

Here $\mathbf{b}_d = \mathbf{B}_d/\mu_0 M_s$ is the dimensionless *magnetic induction*, which depends linearly on $\mathbf{m}$ and satisfies the relation

$$\mathbf{b}_d = \mathbf{m} + \mathbf{h}_d \tag{2.10}$$

and the $L^2$ orthogonality condition

$$\int_{\mathbb{R}^3} \mathbf{h}_d \cdot \mathbf{b}_d \, d\mathbf{x} = 0. \tag{2.11}$$

This allows for reformulations of the magnetostatic self-energy in terms of the magnetic induction.

**Lemma 2.6.1.** [15]: *The following formulations are equivalent descriptions of the magnetostatic self-energy:*

$$e_d(\mathbf{m}) = \int_{\mathbb{R}^3} \|\mathbf{h}_d\|^2 \, d\mathbf{x},$$
$$e_d(\mathbf{m}) = \int_{\Omega} \|\mathbf{m}\|^2 - \mathbf{m} \cdot \mathbf{b}_d \, d\mathbf{x},$$
$$e_d(\mathbf{m}) = \int_{\Omega} \|\mathbf{m}\|^2 \, d\mathbf{x} - \int_{\mathbb{R}^3} \|\mathbf{b}_d\|^2 \, d\mathbf{x}.$$

*Proof.* The proof of this Lemma follows from the definition of the dimensionless stray field energy and the relations (2.10)-(2.11). □

Since the curl of the stray field vanishes according to (2.9), the vector field has to be the gradient of some scalar-valued function $\phi_d : \mathbb{R}^3 \to \mathbb{R}$, the so-called *scalar potential*. This reduces the problem to solving a Poisson equation for $\phi_d$ :

$$\Delta \phi_d = \nabla \cdot \mathbf{m}, \tag{2.12}$$
$$\mathbf{h}_d = -\nabla \phi_d. \tag{2.13}$$

On the other hand, from (2.8) it follows that the magnetic induction $\mathbf{b}_d$ is the curl of a vector field $\mathbf{A}_d : \mathbb{R}^3 \to \mathbb{R}^3$, namely:

$$\mathbf{b}_d = \nabla \times \mathbf{A}_d.$$

$\mathbf{A}_d$ is called the *vector potential*. Therefore, (2.9) yields:

$$\nabla \times (\nabla \times \mathbf{A}_d) = \nabla \times \mathbf{b}_d = \nabla \times \mathbf{m}. \tag{2.14}$$

Since for an arbitrary scalar-valued function $\psi$ the curl of its gradient is zero, we can add its gradient field without changing the magnetic induction $\mathbf{b}_d = \nabla \times (\mathbf{A}_d + \nabla\psi)$. This is called *freedom of gauge*. A common choice is the *Coulomb gauge*

$$\nabla \cdot \mathbf{A}_d = 0.$$

With this, (2.14) simplifies to

$$\Delta \mathbf{A}_d = -\nabla \times \mathbf{m}, \tag{2.15}$$

$$\mathbf{b}_d = \nabla \times \mathbf{A}_d, \tag{2.16}$$

which is a Poisson equation for the vector potential.
Numerical solutions of (2.12)-(2.13) and (2.15)-(2.16) can be obtained in various ways. However, the computation of $\mathbf{h}_d$ and $\mathbf{b}_d$ remains highly nontrivial, because of their non-local nature. The most common approach in literature is to calculate the stray field via the scalar potential. An analytical solution of (2.12) is given by [27]

$$\phi_d(\mathbf{x}) = \frac{1}{4\pi}\left( -\int_\Omega \frac{\nabla \mathbf{m}(\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|}\, d\mathbf{x}' + \int_{\partial\Omega} \frac{\mathbf{n}' \cdot \mathbf{m}(\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|}\, dS \right). \tag{2.17}$$

There are different ways to tackle these problems. Naive solvers which directly evaluate the integrals in (2.17) numerically tend to be very slow and costly, leading to computational costs of $\mathcal{O}(N^2)$, where $N$ is the number of grid points in the discretization [10]. Several improvements can be made to this first approach. Finding the solution of (2.15) with the Finite Element Method in combination with multigrid preconditioning is capable of a complexity of $\mathcal{O}(N)$, but only if the boundary conditions are given, which they are normally not [28]. Therefore, one can apply a coupling between FEM and Boundary Element Methods, but they introduce an additional complexity of $\mathcal{O}(M^2)$, where $M$ is the number of boundary nodes [29]. A linear rate of $\mathcal{O}(N \log N)$ is achieved by the means of Fast Fourier Transforms [30], whereas more complex approaches such as Nonuniform Grid Methods [31] or Fast Multipole Methods [32] are $\mathcal{O}(N)$. More modern techniques even reach sublinear complexity such as Tensor Grid Methods [33] with a computational effort of up to $\mathcal{O}(N^{2/3})$.

However, this work follows a different path and tries to replace the non-local stray field and the magnetic induction by simpler expressions and solve them instead. To this end, we use the upper and lower bounds for the demagnetizing energy proposed by W. F. Brown in [13].

## 2.7 Brown's Energy Bounds for the Demagnetizing Energy

The goal of this section is to derive bounds for the demagnetizing energy $e_d$ which only use variables independent of the magnetization $\mathbf{m}$. More details can be found in the

works of Brown [19], [13], the computations in this chapter follow the derivations in [14]. Let $\mathbf{h} \in L^2(\mathbb{R}^3)^3$ be an arbitrary *irrotational* field, which means $\nabla \times \mathbf{h} = 0$, and $\mathbf{b} \in L^2(\mathbb{R}^3)^3$ an arbitrary *solenoidal* field, i.e. $\nabla \cdot \mathbf{b} = 0$. Then the following upper and lower bounds for $e_d$ hold:

$$\int_\Omega \|\mathbf{m}\|^2 \, d\mathbf{x} - \int_{\mathbb{R}^3} \|\mathbf{h} + \mathbf{m}\|^2 \, d\mathbf{x} \leq e_d(\mathbf{m}) \leq \int_{\mathbb{R}^3} \|\mathbf{b} - \mathbf{m}\|^2 \, d\mathbf{x}.$$

We begin by proving the upper bound. Define

$$U(\mathbf{m}, \mathbf{b}) \coloneqq e_d(\mathbf{m}) + \int_{\mathbb{R}^3} \|\mathbf{b} - \mathbf{b}_d\|^2 \, d\mathbf{x}.$$

This functional attains its minimum at $(\mathbf{m}^*, \mathbf{b}^*)$ if $\mathbf{m}^*$ is the minimum point of $e_d$ and the magnetic induction generated by $\mathbf{m}^*$ is $\mathbf{b}^* = \mathbf{b}_d$. In this case, it holds

$$e_d(\mathbf{m}^*) = U(\mathbf{m}^*, \mathbf{b}^*). \tag{2.18}$$

In order to rewrite $U$ we use the alternative formulations for the demagnetizing energy from Lemma 2.6.1 and expand the norm $\|\mathbf{b} - \mathbf{b}_d\|^2$ to get:

$$\begin{aligned} U(\mathbf{m}, \mathbf{b}) = &\int_\Omega \|\mathbf{m}\|^2 \, d\mathbf{x} - \int_{\mathbb{R}^3} \|\mathbf{b}_d\|^2 \, d\mathbf{x} \\ &+ \int_{\mathbb{R}^3} \|\mathbf{b}\|^2 \, d\mathbf{x} - 2 \int_{\mathbb{R}^3} \mathbf{b} \cdot \mathbf{b}_d \, d\mathbf{x} + \int_{\mathbb{R}^3} \|\mathbf{b}_d\|^2 \, d\mathbf{x}. \end{aligned}$$

Using that $\mathbf{b}$ is solenoidal and therefore its $L^2$-orthogonality with $\mathbf{h}_d$, we simplify this to:

$$\begin{aligned} U(\mathbf{m}, \mathbf{b}) &= \int_\Omega \|\mathbf{m}\|^2 \, d\mathbf{x} - 2 \int_{\mathbb{R}^3} \mathbf{b} \cdot \mathbf{m} \, d\mathbf{x} + \int_{\mathbb{R}^3} \|\mathbf{b}\|^2 \, d\mathbf{x} \\ &= \int_{\mathbb{R}^3} \|\mathbf{b} - \mathbf{m}\|^2 \, d\mathbf{x}. \end{aligned}$$

Hence, from (2.18) it follows that $\int_{\mathbb{R}^3} \|\mathbf{b} - \mathbf{m}\|^2 \, d\mathbf{x}$ is an upper bound for $e_d(\mathbf{m})$ under the constraints

$$\nabla \cdot \mathbf{b} = 0, \quad \|\mathbf{m}\| = \begin{cases} 1 & \text{in } \Omega, \\ 0 & \text{outside.} \end{cases}$$

However, the solenoidal condition can be enforced by the use of a vector potential that generates the magnetic induction, i.e. $\mathbf{b} = \nabla \times \mathbf{A}$, for some vector field $\mathbf{A} : \mathbb{R}^3 \to \mathbb{R}^3$. Expanding the norm gives the upper bound as a function depending on the magnetization and the vector field:

$$e_{\mathbf{A}}(\mathbf{m}, \mathbf{A}) \coloneqq \int_\Omega \|\mathbf{m}\|^2 \, d\mathbf{x} + \int_{\mathbb{R}^3} \|\nabla \mathbf{A}\|^2 \, d\mathbf{x} - 2 \int_\Omega \mathbf{m} \cdot (\nabla \times \mathbf{A}) \, d\mathbf{x}. \tag{2.19}$$

Here $\|\nabla \mathbf{A}\|^2 = \nabla \mathbf{A} : \nabla \mathbf{A}$, where

$$\nabla \mathbf{A} : \nabla \mathbf{A} = Tr[\nabla(\mathbf{A})^T \nabla \mathbf{A}]$$

is the double contraction.
Next, we derive the lower bound in a similar manner. Analogously to the calculations above, define

$$L(\mathbf{m}, \mathbf{h}) \coloneqq e_d(\mathbf{m}) - \int_{\mathbb{R}^3} \|\mathbf{h} - \mathbf{h}_d\|^2 \, d\mathbf{x}.$$

The tuple $(\mathbf{m}^*, \mathbf{h}^*)$ is the maximum point of L if $\mathbf{m}^*$ maximizes $e_d$ and the stray field induced by $\mathbf{m}^*$ is $\mathbf{h}^* = \mathbf{h}_d$. Thus,

$$e_d(\mathbf{m}^*) = L(\mathbf{m}^*, \mathbf{h}^*). \tag{2.20}$$

By expanding the norm, using Lemma 2.6.1 and the irrotational form of $\mathbf{h}$ we find

$$L(\mathbf{m}, \mathbf{h}) = \int_{\mathbb{R}^3} \|\mathbf{h} + \mathbf{m}\|^2 \, d\mathbf{x} + \int_{\Omega} \|\mathbf{m}\|^2 \, d\mathbf{x}.$$

This definition, together with (2.20), gives again the statement about the lower bound under the constraints that $\mathbf{h}$ has to be generated by a scalar potential and the unit norm constraint of the magnetization on the magnetic domain. The lower bound functional reads as:

$$e_\phi(\mathbf{m}, \phi) \coloneqq \int_{\mathbb{R}^3} \|\nabla \phi\|^2 \, d\mathbf{x} + 2 \int_{\Omega} \nabla \phi \cdot \mathbf{m} \, d\mathbf{x}.$$

Instead of directly computing $e_d$, one may now maximize the lower bound or minimize the upper bound with respect to $\mathbf{m}$ and an independent scalar or vector potential.

The aim of this thesis is now to minimize the Gibbs free energy using Brown's upper bound, in particular

$$\min_{\mathbf{m},\mathbf{A}} \left[ \frac{1}{2} e_{\mathbf{A}}(\mathbf{m}, \mathbf{A}) + e_{zee}(\mathbf{m}) + e_a(\mathbf{m}) + e_{ex}(\mathbf{m}) \right], \tag{2.21}$$

with respect to the unit norm constraint

$$\|\mathbf{m}\| = \begin{cases} 1 & \text{in } \Omega, \\ 0 & \text{outside.} \end{cases} \tag{2.22}$$

By replacing the stray field energy with expression (2.19), which involves only local differential operators, the Finite Element discretization will lead to sparse system matrices, reducing the involved matrix-vector products to scaling linear with the number of grid points.
In this work the upper bound is chosen over the lower bound because of two properties of problem (2.21): on the one hand, the use of the lower bound would lead to a min-max problem, which corresponds to a saddle point problem in optimization theory. These are

known to be significantly more challenging than pure minimization problems. In particular, min–max problems often require nested optimization procedures and specialized algorithms, leading to increased computational cost and reduced numerical stability [34]. On the other hand, the min-min formulation allows for a coupling of the involved variables and therefore for a joint minimization procedure, which will be a central concept of this thesis.

## 2.8 The NIST $\mu$MAG standard problem #3

However, for the remainder of this thesis, we will not work with the full problem (2.21), but instead consider the framework of the NIST $\mu$MAG standard problem #3 [35], which allows for direct comparison with previously published results and simplifies some calculations. The main objective of problem #3 is to determine the *single domain limit* (SDL) of a soft ferromagnetic cube, i.e. the length of the cube $L$ in which the energies of two characteristic magnetic configurations, the so-called *flower* and *vortex states* are equal [36].
The flower state represents a quasi-homogeneous magnetization distribution. However, a perfectly uniform state cannot be an equilibrium state, since the stray field energy favors non-uniform magnetization near the boundaries. Consequently, the magnetization tilts outward near the surface of the cube [37].
The exact profile of the state depends on the material and the geometry of the magnet, so an a priori analytical expression cannot be given. Therefore, we proceed as in [38], where the authors used polynomial expansions to describe the state. As a consequence, it is sufficient for this work to depict the flower state as the normalized version of

$$\mathbf{m}_f(x,y,z) = \left(\frac{1}{a}xz, \frac{1}{c}yz + \frac{1}{b^3}y^3z^3, 1\right)^T \tag{2.23}$$

for given parameters $a, b, c \in \mathbb{R}^+$.
The vortex state evolves from the flower state with increasing size of the cube. By contrast, it has a twisted magnetization configuration around an axis. The spins at the center are aligned with this axis, whereas the others rotate clockwise or counterclockwise in circles, creating a symmetric magnetization pattern. Again we adopt a normalized approximation of an analytic ansatz from [10]:

$$\mathbf{m}_v(x,y,z) = \left(-\frac{y}{r}\sqrt{1-\exp\left(-4\frac{r^2}{r_c^2}\right)}, \frac{x}{r}\sqrt{1-\exp\left(-4\frac{r^2}{r_c^2}\right)}, \exp\left(-2\frac{r^2}{r_c^2}\right)\right)^T, \tag{2.24}$$

with $r = \sqrt{x^2+y^2}$ and $r_c = 0.14$. The corresponding magnetization dynamics of both variants are depicted in Figure 2.1.

The length $L$ is expressed in units of an intrinsic length $l_{ex} = \sqrt{A_{ex}/K_m}$ to be independent of the material of the cube. Here, $K_m := \mu_0 M_s^2$ is the magnetic energy density. In

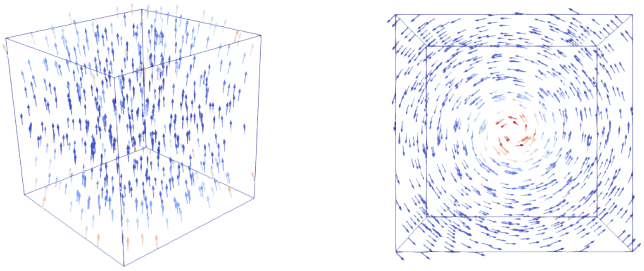

Figure 2.1: Flower state (left) and vortex state (right) of a magnetized cube.

the following, all the energy terms are given in units of $K_m$. In this setting, the total Gibbs free energy (with Brown's upper bound replacing the stray field energy) reads

$$\begin{aligned} e_{NIST}(\mathbf{m}, \mathbf{A}) = \frac{1}{2}\left( \int_\Omega \|\mathbf{m}\|^2 \, d\mathbf{x} + \int_{\mathbb{R}^3} \nabla \mathbf{A} : \nabla \mathbf{A} \, d\mathbf{x} - 2 \int_\Omega \mathbf{m} \cdot (\nabla \times \mathbf{A}) \, d\mathbf{x} \right) \\ + \int_\Omega Q(1 - (\mathbf{m} \cdot \mathbf{a})^2) \, d\mathbf{x} + \int_\Omega \tilde{A}_{ex} \|\nabla \mathbf{m}\|_F^2 \, d\mathbf{x}. \end{aligned} \tag{2.25}$$

A key aspect of working in this framework is the neglect of the Zeeman energy, as we only consider magnetic states with zero external field. The dimensionless parameters of the remaining energy terms are given through the problem specifications. The reduced uniaxial anisotropy constant $Q := \frac{K_1}{K_m}$ is given as $Q = 0.1$ and the easy axis $\mathbf{a}$ should be parallel to the z-axis, therefore we take $\mathbf{a} = (0, 0, 1)^T$. The reduced exchange stiffness constant $\tilde{A}_{ex} := \frac{A_{ex}}{K_m}$ can be computed from the length of the cube via $\tilde{A}_{ex} = \frac{1}{L^2}$ after scaling the domain to a unit cube [39].

# 3 The Finite Element Method

Finding solutions of problem (2.21) usually results in solving Partial Differential Equations (PDEs). The Finite Element Method (FEM) is a widely used approach to solve PDEs numerically, especially in physics, engineering and applied mathematics [40]. Unlike other variational methods, it uses a partition of the computational domain in finitely many, but simpler subdomains, called *finite elements* [41]. On each of these elements the functions occurring in the problem are expressed by a local basis. This gives rise to a system of linear equations that can be solved numerically.
A main advantage of the FEM over other approaches like Finite Difference Methods is the accurate description of the boundary behavior of complex geometries. Moreover, the FEM is based on variational formulations which is backed by a rigorous functional analytical theory, leading to well-defined discrete models.
In the following sections this theory is outlined, as well as the construction of a finite element space and the specific choice of basis functions employed in this work.

## 3.1 Variational Formulations

The essential idea of variational methods is to reformulate a PDE such that suitable approximations can be derived by the FEM in a flexible manner. The so-called *weak formulation* of the PDE is obtained by integrating the equations after multiplying with certain test functions. This relaxes the requirements on the smoothness of the solution and transforms the problem into an integral formulation, which makes it easier to handle when assembling a global system of equations from the discretizations in every element. By choosing appropriate function spaces the formulation guarantees that the solution possesses the right integrability conditions needed. In the context of the FEM one typically uses Sobolev spaces extending the concept of $L^p$ spaces. We will briefly discuss the theoretical concepts in the next chapters for scalar-valued functions and then generalize for vector fields used in the applications. The main theory explained in this section originates from [42], [43] and [44].

We start with the notion of weak derivatives. Let $\Omega \in \mathbb{R}^n$ be the domain of interest and denote by $C_0^\infty(\Omega)$ the space of infinitely differentiable, real valued functions with compact support. Consider $1 \leq p \leq \infty$ and $f \in L^p(\Omega)$ a real-valued, measurable function. For a multi-index $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}^n$ with length $|\alpha| = \alpha_1 + \cdots + \alpha_n$ we use the following

notation [44]

$$D^\alpha := \frac{\partial^\alpha}{\partial_1^\alpha x_1 \cdots \partial_n^\alpha x_n}.$$

Then, a function $v \in L^p(\Omega)$ is called the $|\alpha|$-*th weak derivative* of $f$ if

$$\int_\Omega D^\alpha \varphi f \, d\mathbf{x} = (-1)^{|\alpha|} \int_\Omega \varphi v \, d\mathbf{x}$$

holds for all $\varphi \in C_0^\infty(\Omega)$ [42]. In this case we identify $D^\alpha f := v$.
Now, we can define Sobolev spaces of scalar-valued functions.

**Definition 3.1.1.** [42]: Let $1 \leq p \leq \infty$, $\Omega \subset \mathbb{R}^n$ an open subset and $m$ be a nonnegative integer. We define the *Sobolev space* $W^{m,p}(\Omega)$ as

$$W^{m,p}(\Omega) = \{f \in L^p(\Omega) \,|\, D^\alpha f \in L^p(\Omega) \quad \forall \alpha \in \mathbb{N}^n, |\alpha| \leq m\}.$$

It is equipped with the norm

$$\|f\|_{W^{m,p}(\Omega)} := \left( \sum_{|\alpha| \leq m} \|D^\alpha f\|_{L^p(\Omega)}^p \right)^{1/p}.$$

Similar to $L^p$-spaces, the case $p = 2$ requires special attention.
We define $H^m(\Omega) := W^{m,2}(\Omega)$. The Sobolev space $H^m(\Omega)$ is a Hilbert space equipped with the inner product [44]

$$(u, v)_m := \sum_{|\alpha| \leq m} \int_\Omega (D^\alpha u)(D^\alpha v) \, d\mathbf{x}.$$

Consequently, we get the following norm on $H^m(\Omega)$ :

$$\|u\|_{H^m(\Omega)} = \sqrt{(u,u)_m} = \sqrt{\sum_{|\alpha| \leq m} \int_\Omega (D^\alpha u)^2 \, d\mathbf{x}},$$

and the $H^m$-seminorm

$$|u|_{H^m(\Omega)} = \sqrt{\sum_{|\alpha| = m} \int_\Omega (D^\alpha u)^2 \, d\mathbf{x}}.$$

For a bounded domain $\Omega$ we additionally define the closure of $C_0^\infty(\Omega)$ in $H^m(\Omega)$.

**Definition 3.1.2.** [44]: Let $\Omega \subset \mathbb{R}^n$ be a an open subset and $m \in \mathbb{N}$. Then:

$$H_0^m(\Omega) := \overline{C_0^\infty(\Omega)}^{\|\cdot\|_{H^m(\Omega)}}.$$

Equivalently,

$$H_0^m(\Omega) = \{u \in H^m(\Omega) \mid \exists (\varphi_j)_{j \in \mathbb{N}} \subset C_0^\infty(\Omega) : \lim_{j \to \infty} \|u - \varphi_j\|_{H^m(\Omega)} = 0\}.$$

Another topic which has to be addressed is the treatment of boundary conditions. Since functions in $H^m(\Omega)$ are in general not continuous nor defined on subsets of measure zero, it is not trivial which value a function $u \in H^m(\Omega)$ takes at the boundary of a domain $\Omega$. Therefore, the concept of trace operators is needed. The following Lemma is stated for $m = 1$, but it can be extended to higher order (see [43]).

**Lemma 3.1.1.** [42, Thm. 5.5.1]: *Let $\Omega \subset \mathbb{R}^n$ be a bounded domain with a $C^1$-smooth boundary $\partial\Omega$.*
*Then, there exists a continuous, linear operator*

$$T : H^1(\Omega) \to L^2(\partial\Omega)$$

*with*

$$Tu = u|_{\partial\Omega} \quad \forall u \in H^1(\Omega) \cap C(\bar{\Omega}).$$

*We call $Tu$ the trace of $u$ and write $Tu = u|_{\partial\Omega}$.*

*Proof.* A constructive proof of this result can be found in [42]. □

We now derive the weak formulation for a prototypical elliptic boundary-value problem of second order. Other types are treated in a similar manner, but are not considered in the course of this thesis. See [42] for further reading.
Let $\Omega \subset \mathbb{R}^n$ be open and bounded. We consider

$$Lu = f \quad \text{in } \Omega, \tag{3.1}$$
$$u = 0 \quad \text{on } \partial\Omega\,, \tag{3.2}$$

where $f \in L^2(\Omega)$ is given, $u = u(\mathbf{x}) \in C^2(\Omega) \cap C(\bar{\Omega})$ is the unknown function and $L$ is a second-order, elliptic differential operator of the following form

$$Lu := -\nabla \cdot (A\nabla u) + \mathbf{b} \cdot \nabla u + cu.$$

Here, $A = (a_{ij})_{i,j=1}^n \in (L^\infty(\Omega))^{n\times n}$, $\mathbf{b} = (b_1, \ldots, b_n)^T \in (L^\infty(\Omega))^n$ and $c \in L^\infty(\Omega)$ are given coefficient functions. Additionally, $A$ should fulfill uniform ellipticity: there exists a constant $\theta > 0$ such that

$$\sum_{i,j=1}^{n} a_{ij}(\mathbf{x})\xi_i\xi_j \geq \theta|\boldsymbol{\xi}|^2,$$

for all $\mathbf{x} \in \Omega$ and $\boldsymbol{\xi} \in \mathbb{R}^n$ [42]. We consider homogeneous Dirichlet boundary conditions (3.2), since these are the only type of boundary conditions needed in this thesis.
Let now $\varphi \in C_0^\infty(\Omega)$ be an arbitrary test function. Multiplying equation (3.1) with $\varphi$ and integrating partially yields

$$\int_\Omega (A\nabla u) \cdot \nabla\varphi \, d\mathbf{x} + \int_\Omega (\mathbf{b} \cdot \nabla u)\varphi \, d\mathbf{x} + \int_\Omega cu\varphi \, d\mathbf{x} = \int_\Omega f\varphi \, d\mathbf{x}. \tag{3.3}$$

Here, the boundary integrals vanish, since $\varphi$ has compact support. Equation (3.3) is called the *weak formulation* of the PDE, because the smoothness assumptions on $u$ are significantly weakened. Now it is only necessary that $u \in H_0^1(\Omega)$.
By extending the test space to $H_0^1(\Omega)$ and defining the bilinear operator

$$\begin{aligned} &B : H_0^1(\Omega) \times H_0^1(\Omega) \to \mathbb{R}, \\ &B(u,v) = \int_\Omega (A\nabla u) \cdot \nabla v \, d\mathbf{x} + \int_\Omega (\mathbf{b} \cdot \nabla u) v \, d\mathbf{x} + \int_\Omega cuv \, d\mathbf{x}, \end{aligned} \tag{3.4}$$

equation (3.3) can be reformulated as:
find $u \in H_0^1(\Omega)$ such that:

$$B(u,\varphi) = (f,\varphi)_{L^2(\Omega)} \quad \forall \varphi \in H_0^1(\Omega). \tag{3.5}$$

This expression is called *variational formulation* of the PDE and a function $u \in H_0^1(\Omega)$ satisfying (3.5) is called *weak solution*. Existence and uniqueness of such weak solutions follow from standard functional-analytic arguments. For a thorough analysis of this topic, we refer to [44, Chapter 3.4.1] and [45, Chapter 2.1.6]. Here, we only state the central proposition in a setting of a general Hilbert space.

**Lemma 3.1.2.** (**Lax-Milgram**) [44, Lem. 3.1, Def. 2.2]: *Let $H$ be a Hilbert space and $B : H \times H \to \mathbb{R}$ a bilinear form, satisfying the following conditions:*

- *$B$ is continuous, i.e. there exists a constant $M > 0$ such that*

$$|B(u,v)| \leq M \|u\|_H \|v\|_H \quad \forall u, v \in H.$$

- *$B$ is coercive, i.e. there exists a constant $\alpha > 0$ such that*

$$B(v,v) \geq \alpha \|v\|_H^2. \quad \forall v \in H.$$

*Additionally, let $f : H \to \mathbb{R}$ be a linear and continuous functional. Then there exists a unique solution to the problem:*
*find $u \in H$ such that:*

$$B(u,v) = f(v) \quad \forall v \in H. \tag{3.6}$$

*Moreover, the norm estimate*

$$\|u\|_H \leq \frac{1}{\alpha} \|f\|_{H'}$$

*holds. Here, $H'$ is the dual space of $H$.*

*Proof.* The following proof repeatedly makes use of *Riesz' Representation Theorem* [42], which states that for a linear and continuous functional $f \in H'$, there exists a unique function $u \in H$, such that

$$(u,v)_H = f(v) \quad \forall v \in H.$$

Let now $T, T' : H \to H$ be two linear operators defined by

$$(Tu, v)_H = B(u, v) \quad \forall v \in H,$$
$$(T'u, v)_H = B(v, u) \quad \forall v \in H.$$

By Riesz' Representation Theorem both $Tu$ and $T'u$ exist and are uniquely defined. It holds,

$$(Tu, v)_H = B(u, v) = (T'v, u)_H = (u, T'v)_H,$$

and therefore $T'$ is the adjoint operator of $T$. Furthermore, for all $u \in H$,

$$\|Tu\|_H^2 = (Tu, Tu)_H = B(u, Tu) \leq M\|u\|_H\|Tu\|_H$$

and thus, $T$ is continuous. The same holds for $T'$ as well.
Now, define the bilinear form $\hat{B} : H \times H \to \mathbb{R}$ as

$$\hat{B}(u, v) = (TT'u, v)_H = (T'u, T'v)_H \quad \forall u, v \in H.$$

By construction, $\hat{B}$ is symmetric and also coercive and continuous, since

$$\begin{aligned}\alpha^2\|v\|_H^4 &\leq (B(v, v))^2 = ((T'v, v)_H)^2 \\ &\leq \|v\|_H^2\|T'v\|_H^2 = \|v\|_H^2(T'v, T'v)_H = \|v\|_H^2\hat{B}(v, v)\end{aligned}$$

and

$$|\hat{B}(u, v)| = |(T'u, T'v)_H| \leq \|T'u\|_H\|T'v\|_H \leq M^2\|u\|_H\|v\|_H.$$

Therefore, $\hat{B}(\cdot, \cdot)$ defines an inner product in $H$. Using Riesz' Representation Theorem for this inner product concludes that $\exists! w \in H$, such that

$$\hat{B}(w, v) = f(v) \quad \forall v \in H.$$

Let $u \coloneqq T'w$. Then,

$$B(u, v) = B(T'w, v) = (TT'w, v)_H = \hat{B}(w, v) = f(v) \quad \forall v \in H.$$

Thus, the existence of a solution of (3.6) follows. Moreover, assume that there exist two solutions $u_1$ and $u_2$. Then,

$$B(u_1 - u_2, v) = 0 \quad \forall v \in H \Rightarrow 0 = B(u_1 - u_2, u_1 - u_2) \geq \alpha\|u_1 - u_2\|^2 \Rightarrow u_1 = u_2.$$

The norm estimate holds, since

$$\|u\|_H^2 \leq \frac{1}{\alpha}B(u, u) = \frac{1}{\alpha}f(u) \leq \frac{1}{\alpha}\|f\|_{H'}\|u\|_H.$$

□

## 3.2 Galerkin Methods

The weak formulation (3.3) is defined on a subspace of the infinite-dimensional space $H^1(\Omega)$. However, for numerical methods it is necessary to approximate this continuous formulation using discrete functions. Therefore, the aim of this section is to develop finite-dimensional subspaces of $H^1(\Omega)$ suitable for this purpose. The following descriptions are based on [41] and [44].
We start with a general formulation of the problem derived in the previous section. Let $n \geq 1$ and $\Omega \subseteq \mathbb{R}^n$ be a domain. Moreover, let $V$ be an appropriate subspace of $H^1(\Omega)$. Then, the elliptic boundary value problem (3.2) can be written in weak form as

$$\text{find } u \in V: \quad B(u,v) = (f,v)_{L^2(\Omega)} \quad \forall v \in V, \tag{3.7}$$

where $B(\cdot,\cdot)$ denotes the bilinear form defined in (3.4).
Now, assume there exist finite-dimensional subspaces $V_h \subset V$ with $dim V_h = N_h < \infty$, depending on a parameter $h > 0$. The approximate problem reads as:

$$\text{find } u_h \in V_h: \quad B(u_h,v_h) = (f,v_h)_{L^2(\Omega)} \quad \forall v_h \in V_h. \tag{3.8}$$

Since $V_h$ is finite-dimensional, there exists a basis $(\varphi_j)_{j=1}^{N_h} \subset V_h$ and $u_h$ can be expressed as

$$u_h(x) = \sum_{j=1}^{N_h} c_j \varphi_j(x),$$

where $c_j \in \mathbb{R}$, $j = 1,\dots,N_h$ are unknown coefficients.
Substituting this expression into (3.8) yields

$$\text{find } c_j \in \mathbb{R}: \quad \sum_{j=1}^{N_h} c_j B(\varphi_j, v_h) = (f,v_h)_{L^2(\Omega)} \quad \forall v_h \in V_h.$$

Testing against the basis functions leads to

$$\text{find } c_j \in \mathbb{R}: \quad \sum_{j=1}^{N_h} c_j B(\varphi_j, \varphi_i) = (f,\varphi_i)_{L^2(\Omega)} \quad \forall i = 1,\dots,N_h. \tag{3.9}$$

This is called *Galerkin formulation.*
Equation (3.9) can now be written as the linear system

$$A\mathbf{c} = \mathbf{b}, \tag{3.10}$$

where the *stiffness matrix* $A \in \mathbb{R}^{N_h \times N_h}$ and the *load vector* $\mathbf{b} = (b_1,\dots,b_{N_h})^T \in \mathbb{R}^{N_h}$ are defined by

$$A_{ij} = B(\varphi_j,\varphi_i), \quad b_i = (f,\varphi_i)_{L^2(\Omega)}.$$

We now give some general properties of the stiffness matrix, but it has to be noted that additional characteristics of $A$ and $\mathbf{b}$ such as sparsity or the condition number depend on the choice of the subspace $V_h$ and its basis $(\varphi_j)_{j=1}^{N_h}$.

**Proposition 3.2.1.** [44, Thm. 4.1, Property 4.1]: *Let $A \in \mathbb{R}^{N_h \times N_h}$ be the stiffness matrix defined in (3.9).*

- *$A$ is symmetric if and only if the bilinear form $B(\cdot,\cdot)$ is symmetric.*
- *If $B(\cdot,\cdot)$ is coercive, then $A$ is positive definite.*

*Proof.* The symmetry follows directly from the definition of the entries of $A$, since

$$A_{ij} = B(\varphi_j, \varphi_i) = B(\varphi_i, \varphi_j) = A_{ji}.$$

For positive definiteness let $\mathbf{v} \in \mathbb{R}^{N_h}$ be an arbitrary vector. From the bilinearity of $B(\cdot,\cdot)$ it follows

$$\begin{aligned}
\mathbf{v}^T A \mathbf{v} &= \sum_{j=1}^{N_h} \sum_{i=1}^{N_h} v_j A_{ij} v_i \\
&= \sum_{j=1}^{N_h} \sum_{i=1}^{N_h} v_j B(\varphi_j, \varphi_i) v_i \\
&= B(\sum_{j=1}^{N_h} v_j \varphi_j, \sum_{i=1}^{N_h} v_i \varphi_i) \\
&= B(v_h, v_h).
\end{aligned}$$

Here, $v_h \in V_h$ is the vector spanned by the basis functions $(\varphi_j)_{j=1}^{N_h}$ and the coefficients $v_i \in \mathbb{R} \quad i = 1, \ldots, N_h$. Furthermore, coercivity of the bilinear form induces, that there exists a constant $\alpha > 0$ such that

$$B(v_h, v_h) \geq \alpha \|v_h\|^2 \geq 0.$$

Moreover, $B(v_h, v_h) = 0 \Leftrightarrow \|v_h\|^2 = 0 \Leftrightarrow v_h = 0 \Leftrightarrow \mathbf{v} = 0$. Hence, positive definiteness of $A$ follows. □

The properties of the bilinear form also provide convergence results.

**Lemma 3.2.2.** [44, Lem. 4.1]: *Let $u \in V$ be the solution of*

$$B(u, v) = f(v) \quad \forall v \in V \tag{3.11}$$

*and $u_h \in V_h$ the solution of the respective Galerkin problem. Then the orthogonality relation*

$$B(u - u_h, v_h) = 0 \quad \forall v_h \in V_h$$

*holds.*

*Proof.* Since $V_h \subset V$ the exact solution $u$ also satisfies

$$B(u, v_h) = f(v_h) \quad \forall v_h \in V_h.$$

Therefore,

$$B(u - u_h, v_h) = B(u, v_h) - B(u_h, v_h) = f(v_h) - f(v_h) = 0.$$

□

This relations leads to an a priori error estimate for the Galerkin solution.

**Theorem 3.2.3.** (**Céa Lemma**) [44]: *Let $u$ be the solution of the exact problem (3.11) and $u_h$ the solution of the corresponding Galerkin problem. Additionally, let $M > 0$ and $\alpha > 0$ the continuity and coercivity constants of $B(\cdot, \cdot)$. Then, it holds*

$$\|u - u_h\|_V \leq \frac{M}{\alpha} \inf_{v_h \in V_h} \|u - v_h\|_V.$$

*Proof.* By using the coercivity, the continuity of $B(\cdot, \cdot)$ and the orthogonality property from Lemma 3.2.2 twice, the result is obtained directly by:

$$\begin{aligned} \alpha \|u - u_h\|_V^2 &\leq B(u - u_h, u - u_h) = B(u - u_h, u - v_h) \\ &\leq M \|u - u_h\|_V \|u - v_h\|_V, \end{aligned}$$

for all $v_h \in V_h$. □

If additionally a certain 'density' of $V_h$ is assumed, we obtain a result in the limit.

**Corollary 3.2.4.** [44]: *If $V_h$ fulfills the approximation property*

$$\lim_{N_h \to \infty} \inf_{v_h \in V_h} \|u - v_h\|_V = 0,$$

*then*

$$\lim_{N_h \to \infty} \|u - u_h\|_V = 0.$$

## 3.3 Approximation by Piecewise Polynomial Functions

We proceed by constructing concrete spaces for the problems we face in our applications. Before that, we discuss how to mesh the domain. For the content of this section we refer to [44], [45] and [46]. Even though this thesis only deals with problems in $\mathbb{R}^3$, the theoretical concepts of this section are also explained in two dimensions for the sake of comprehensibility. Therefore, let $\Omega \subseteq \mathbb{R}^n$ with $n \in \{2, 3\}$ denote the domain of interest. The essential concept of the FEM is the partitioning of $\Omega$ into simpler subdomains. We employ triangles in two dimensions and tetrahedra in three dimensions to build them.

**Definition 3.3.1.** [44],[45]: A subset $T \subset \mathbb{R}^2$ is called a *triangle* if there exist points $x_1, x_2, x_3 \in \mathbb{R}^2$ such that $T = conv\{x_1, x_2, x_3\}$ and $|T| > 0$.
Similarly, a subset $T \subset \mathbb{R}^3$ is called *tetrahedron* if there exist points $y_1, y_2, y_3, y_4 \in \mathbb{R}^3$ such that $T = conv\{y_1, y_2, y_3, y_4\}$ and $|T| > 0$.
The *diameter* of a triangle or a tetrahedron $T$ is expressed as

$$h_T \coloneqq diam(T),$$

where $diam(T) = \max_{x,y \in T} |x - y|$.
Moreover, we denote with the *sphericity* $\rho_T$ the diameter of its inscribed circle or sphere.

An important example is the so-called *reference element*

$$T_r = \begin{cases} conv\{(0,0),(0,1),(1,0)\} & \text{in } \mathbb{R}^2, \\ conv\{(0,0,0),(1,0,0),(0,1,0),(0,0,1)\} & \text{in } \mathbb{R}^3. \end{cases} \tag{3.12}$$

We call the family of subdomains $\mathcal{T} = \{T_1, \ldots, T_M;\ M \in \mathbb{N}\}$ a *triangulation* or *mesh* of $\Omega$ and the individual triangles/tetrahedrons $T_k$ the *elements* of the mesh. Denoting their diameters as $h_k$, the *global mesh size* is defined by

$$h \coloneqq \max_{k \in \{1,\ldots,M\}} h_k.$$

To ensure a well-structured discretization, we induce the following regularity properties.

**Definition 3.3.2.** [44]: A triangulation $\mathcal{T} = \{T_1, \ldots, T_M\}$ is called *admissible* if:

- $\mathring{T}_k \neq \emptyset \quad \forall k \in \{1, \ldots, M\}$,
- $\mathring{T}_k \cap \mathring{T}_l = \emptyset \quad \forall k, l \in \{1, \ldots, M\},\ k \neq l$,
- $\bigcup_{i=1}^{n} T_i = \bar{\Omega}$,
- For any $F = T_k \cap T_l$ with $k \neq l$, one of the following holds:

1. $F = \emptyset$,
2. $F$ is an entire edge (or face) shared by $T_k$ and $T_l$ or,
3. F is a single vertex shared by $T_k$ and $T_i$.

Moreover, a triangulation is called *(shape) regular* if there exists a constant $\delta > 0$ such that

$$\frac{h_T}{\rho_T} \leq \delta \quad \forall T \in \mathcal{T},$$

where $h_T$ and $\rho_T$ denote the diameter and the sphericity of an element $T$ respectively.

Shape regularity of a mesh ensures that ill-shaped elements are excluded, meaning the inner angles of any triangle or tetrahedron cannot be too wide or too narrow. An example of a two-dimensional admissible and regular mesh can be seen in Figure 3.1.

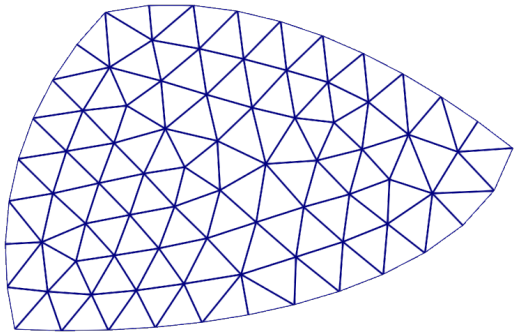

Figure 3.1: Admissible and regular triangulation of a two-dimensional domain.

In the following we will always assume that we work with admissible and regular meshes. It is also sometimes useful to impose the additional property of quasi-uniformity to the mesh:

**Definition 3.3.3.** [47, Def. 6.3.1]: A triangulation $\mathcal{T}$ with global mesh size $h > 0$ is called *quasi-uniform* if there exists a $\tau > 0$ such that

$$\min_{T_k \in \mathcal{T}} h_k > \tau h.$$

In practice one chooses finite element spaces as subspaces of $H^1(\Omega)$. To this end, we show first that continuity but not necessarily differentiability of functions is sufficient.

**Theorem 3.3.1.** [46, Thm. 5.2]: *Let $k \in \mathbb{N}$, $n \in \{2, 3\}$, $\Omega \subset \mathbb{R}^n$ bounded and $\mathcal{T}$ be a mesh of $\Omega$. Moreover, let $v : \bar{\Omega} \to \mathbb{R}$ be infinitely differentiable on every element of $\mathcal{T}$. Then,*

$$v \in H^k(\Omega) \Leftrightarrow v \in C^{k-1}(\bar{\Omega}).$$

*Proof.* We will prove both directions only for $k = 1$. The case of higher regularity follows from induction.
Let $k = 1$ and $v \in C(\bar{\Omega})$ be infinitely often differentiable on every element of $\mathcal{T} = \{T_j\}_{j=1}^M$. For $i = 1, \dots, n$ define the piecewise function $w_i : \Omega \to \mathbb{R}$, such that for every $j = 1, \dots, M$ it holds

$$w_i|_{T_j} \coloneqq \partial_{x_i} v|_{T_j}.$$

On the edges of the elements $w_i$ takes one of the limiting values.
Furthermore, let $\varphi \in C_0^\infty(\Omega)$ be a test function. Then,

$$\int_\Omega \varphi w_i \, d\mathbf{x} = \sum_{j=1}^M \int_{T_j} \varphi(\partial_{x_i} v) \, d\mathbf{x} = \sum_{j=1}^M \Big[ - \int_{T_j} (\partial_{x_i} \varphi) v \, d\mathbf{x} + \int_{\partial T_j} \varphi \, v \, \nu_i^{(j)} \, dS \Big],$$

where $\boldsymbol{\nu}^{(j)} = (\nu_1^{(j)}, \dots, \nu_n^{(j)})$ denotes the outward pointing unit normal vector on $T_j$.
Since $\varphi$ has compact support the boundary integral vanishes and by continuity of $v$ we have

$$\int_\Omega \varphi w_i \, d\mathbf{x} = - \int_\Omega (\partial_{x_i} \varphi) v \, d\mathbf{x},$$

so $w_i$ is the weak derivative of $v$ with respect to $x_i$. Furthermore, $v$ is piecewise smooth on a bounded domain and therefore its classical derivatives are bounded on each element $T_j$, hence $w_i \in L^2(\Omega)$. From this it follows that $v \in H^1(\Omega)$.
For the converse direction assume $v \in H^1(\Omega)$. We have $v|_{T_j} \in C^\infty(\bar{T}_j)$ for $j = 1, \dots, M$, but it is still possible that there is a jump between two neighboring elements $T^+$ and $T^-$ in the interior of $\Omega$. Denote that jump by

$$[v](\mathbf{x}) \coloneqq v|_{T^+}(\mathbf{x}) - v|_{T^-}(\mathbf{x}) \quad \forall \mathbf{x} \in T^+ \cap T^-.$$

We show that $[v] = 0$ almost everywhere on every interior interface and thus $v \in C(\bar{\Omega})$. Let $T^+$ and $T^-$ be two arbitrary neighboring elements and $\Sigma \coloneqq T^+ \cap T^-$. Choose $\varphi \in C_0^\infty(U)$ on a neighborhood $U$ of $\Sigma$ such that $U$ intersects with no other intersections between elements or the boundary. Since $v \in H^1(\Omega)$, we have for all $i = 1, \dots, n$, that

$$\int_\Omega v(\partial_{x_i} \varphi) \, d\mathbf{x} = - \int_\Omega (\partial_{x_i} v) \varphi \, d\mathbf{x}. \tag{3.13}$$

On the other hand, on the intersection $T^\pm \cap U$ we can perform classical integration by parts to get

$$\int_{T^\pm \cap U} v(\partial_{x_i} \varphi) \, d\mathbf{x} = - \int_{T^\pm \cap U} (\partial_{x_i} v) \varphi \, d\mathbf{x} + \int_{\Sigma \cap U} v|_{T^\pm} \, \varphi \, \nu_i^\pm \, dS,$$

where $\boldsymbol{\nu}^{\pm}$ are the outward unit normal vectors of $T^{\pm}$.
Summing the identities for $T^+$, respectively $T^-$ gives

$$\begin{aligned}\int_U v(\partial_{x_i}\varphi)\,d\mathbf{x} &= -\int_U (\partial_{x_i}v)\varphi\,d\mathbf{x} + \int_{\Sigma\cap U}(v|_{T^+}\nu_i^+ + v|_{T^-}\nu_i^-)\varphi\,dS \\ &= -\int_U (\partial_{x_i}v)\varphi\,d\mathbf{x} + \int_{\Sigma\cap U}[v]\,\nu_i^+\,\varphi\,dS.\end{aligned}$$

The last inequality follows from $\boldsymbol{\nu}^+ = -\boldsymbol{\nu}^-$.
Using (3.13), we get

$$\int_{\Sigma\cap U}[v]\,\nu_i^+\,\varphi\,dS = 0 \quad \forall\varphi\in C_0^\infty(U) \quad \forall i = 1,\dots,n,$$

which proves the claim. □

Another important property is that locally regular functions on a mesh are globally well-defined.

**Theorem 3.3.2.** [44, Property 4.2]: *Let $v:\Omega\to\mathbb{R}$ and $\mathcal{T} = \{T_j\}_{j=1}^M$ be a mesh on $\Omega$. Then, the following statements are equivalent:*

1. $v\in H^1(\Omega)$.
2. $v|_{\mathring{T}_j}\in H^1(\mathring{T}_j) \quad \forall j = 1,\dots,M$ *and on* $F := T_l\cap T_k$, $l\neq k$, *both* $v|_{T_k}$ *and* $v|_{T_l}$ *have the same trace on* $F$.

*Proof.* The proof of this theorem follows a similar approach as the proof of Theorem 3.3.1, so we do not include it here. For a detailed explanation we refer to [47]. □

Now, we construct explicit function spaces for the Galerkin approximation.

**Definition 3.3.4.** [44]: Let $n\in\mathbb{N}$ and $\Omega\subset\mathbb{R}^n$. For $d\geq 1$ the space of *polynomials of degree* $\leq d$ is defined as

$$\mathcal{P}_d(\Omega) := \left\{ p(\mathbf{x}) = \sum_{\substack{0\leq\alpha_1,\dots,\alpha_n \\ \alpha_1+\cdots+\alpha_n\leq d}} c_{\alpha_1,\dots,\alpha_n} x_1^{\alpha_1}\dots x_n^{\alpha_n}, \quad c_{\alpha_1,\dots,\alpha_n}\in\mathbb{R} \right\}.$$

*Remark.* The dimension of $\mathcal{P}_d$ is given by

$$N_d := dim\mathcal{P}_d = \binom{d+n}{d}.$$

Based on this, we construct the finite-dimensional subspace of *piecewise polynomials* on a mesh $\mathcal{T}$.

**Definition 3.3.5.** [48]: Let $\mathcal{T}_h$ be a mesh of $\Omega$ with mesh size $h$ and $d \geq 1$. Define

$$V_h^d := \{v \in C(\Omega) \mid v|_T \in \mathcal{P}_d(T) \quad \forall T \in \mathcal{T}_h\}.$$

Moreover,

$$\mathring{V}_h^d := \{v \in V_d^h \mid v|_{\partial\Omega} = 0\}.$$

Here, the condition on the boundary is understood in the sense of Lemma 3.1.1.

By Theorems 3.3.1 and 3.3.2, the subspace relations $V_h^d \subset H^1(\Omega)$ and $\mathring{V}_h^d \subset H_0^1(\Omega)$ are well-defined.
In order to construct a basis, we note that an element $p \in \mathcal{P}_d$ is well-defined, whenever its value on $N_d$ chosen points is known. This holds also for functions $v \in V_h^d$. Therefore, we fix $N_d$ points on every element of the mesh to determine the basis functions. These points are called *degrees of freedom.*
Figure 3.2 illustrates these degrees of freedom for $d = 1$ and $d = 2$.

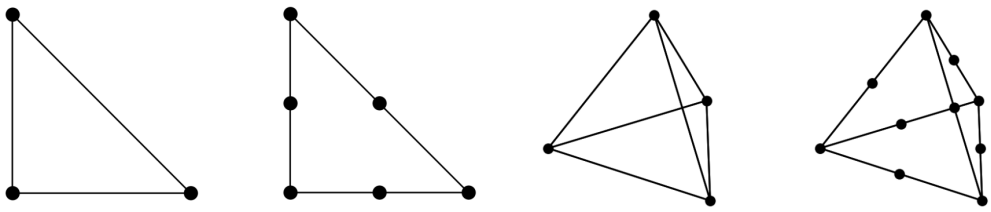

Figure 3.2: Degrees of freedom on a single element in two and three dimensions for $\mathcal{P}_1$ and $\mathcal{P}_2$ polynomials, respectively.

Let $\{\mathbf{z}_1, \ldots, \mathbf{z}_{N_d}\}$ denote these nodes. Then we define the *nodal basis* of $V_h^d$ by the set of the characteristic Lagrangian functions [44]

$$\mathcal{B} = \{\varphi_i\}_{i=1}^{N_d}, \quad \varphi_i(\mathbf{z}_j) = \delta_{ij} = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases} \qquad i, j = 1, \ldots, N_d.$$

Every function $v_h \in V_h^d$ can be expressed as a linear combination of the basis elements by

$$v_h(\mathbf{x}) = \sum_{i=1}^{N_d} c_i \varphi_i(\mathbf{x}),$$

with $c_i = v_h(\mathbf{z}_i)$.
Coming back to problem (3.8) we obtain the Galerkin formulation by expressing the unknown function $u_h \in V_h^d$ through the nodal basis,

$$u_h(\mathbf{x}) = \sum_{i=1}^{N_d} u_i \varphi_i(\mathbf{x}).$$

Setting $\mathbf{u} \coloneqq (u_1, \dots, u_{N_d})^T$, this leads again to a linear system

$$A\mathbf{u} = \mathbf{b},$$

with stiffness matrix $A \in \mathbb{R}^{N_d \times N_d}$ and load vector $\mathbf{b} \in \mathbb{R}^{N_d}$ as defined in (3.10). However, by the explicit choice of the basis, we can say something about the sparsity of $A$.
$A$ is indeed a sparse matrix, since a basis function $\varphi_i \in \mathcal{B}$ is compactly supported on the elements of the mesh that have the node $\mathbf{z}_i$ in common. In particular, $A_{ij} = B(\varphi_j, \varphi_i) \neq 0$ if and only if $\mathbf{z}_i$ and $\mathbf{z}_j$ are nodes of the same element. Exploiting this structure is one of the key computational advantages of the FEM.

## 3.4 Error Analysis

The aim of this section is to derive an estimate for the approximation error $\|u - u_h\|_{H^1}$ in the energy norm and establish a convergence rate of the $\mathcal{P}_d$-FEM for $d \geq 1$. We follow the approach of [44] and [47]. We start by estimating the interpolation error.

**Definition 3.4.1.** [44]: The operator $\Pi_h^d : C^0(\bar{\Omega}) \to V_h^d$ defined by

$$\Pi_h^d v = \sum_{i=1}^{N_d} v(\mathbf{z}_i)\varphi_i$$

is called the *(global) interpolation operator.*
For an element $T_k \in \mathcal{T}$ the associated *local interpolation operator* is defined as

$$\Pi_{T_k}^d v = \sum_{i=1}^{N_k} v(\mathbf{z}_{i,k})\varphi_i|_{T_k}$$

where $N_k$ is the number of degrees of freedom of $T_k$ and $\mathbf{z}_{i,k}$ are its nodal points.

We now introduce an affine transformation between the reference element (3.12) and any other element $T \in \mathcal{T}$:

$$\begin{aligned} F_T &: T_r \to T, \\ F_T(\hat{\mathbf{x}}) &= B_T \hat{\mathbf{x}} + \mathbf{b}_T \end{aligned} \tag{3.14}$$

with $B_T \in \mathbb{R}^{n\times n}$ and $\mathbf{b}_T \in \mathbb{R}^n$.
The strategy for the following derivations is to map to the reference element, estimate the relevant norms there, and then map back to $T$. In fact, it holds,

$$(\Pi_T^d v) \circ F_T = \Pi_{T_r}^d \hat{v},$$

where $\hat{v} = v \circ F_T$.
We begin by estimating the $H^1$-seminorms. The proof of the next statement can be found in [47].

**Proposition 3.4.1.** [47, Prop. 3.4.1, 3.4.2] *Let $T \in \mathcal{T}$ and $v \in H^1(T)$. Set $\hat{v} := v \circ F_T$. Then $\hat{v} \in H^1(T_r)$. Moreover, there exists a constant $C > 0$ such that*

$$|\hat{v}|_{H^1(T_r)} \leq C\|B_T\||detB_T|^{-\frac{1}{2}}|v|_{H^1(T)}$$

*and*

$$|v|_{H^1(T)} \leq C\|B_T^{-1}\||detB_T|^{\frac{1}{2}}|\hat{v}|_{H^1(T_r)}$$

*with $\|\cdot\|$ being the matrix norm associated to the Euclidean norm in $\mathbb{R}^n$. Furthermore, these norms can be estimated by*

$$\|B_T\| \leq \frac{h_T}{\rho_{T_r}}, \quad \|B_T^{-1}\| \leq \frac{h_{T_r}}{\rho_T}$$

*where $h_T, \rho_T$ and $h_{T_r}, \rho_{T_r}$ are the diameter and the sphericity of $T$ and $T_r$.*

The next two Lemmas provide the final step for first error measurements:

**Lemma 3.4.2.** (Bramble-Hilbert) [44, Lem. 4.4] *Let $d \geq 0$ and let $\hat{l} : H^{d+1}(T_r) \to H^1(T_r)$ be a linear, continuous operator which is polynomial preserving, i.e.*

$$\hat{l}(\hat{p}) = \hat{p} \quad \forall \hat{p} \in \mathcal{P}_d(T_r).$$

*Then, there exists a $C > 0$ such that*

$$\|\hat{v} - \hat{l}(\hat{v})\|_{H^1(T_r)} \leq C \inf_{\hat{p}\in\mathcal{P}_d(T_r)} \|\hat{v} - \hat{p}\|_{H^{d+1}(T_r)} \quad \forall \hat{v} \in H^{d+1}(T_r).$$

*Proof.* Let $\hat{v} \in H^{d+1}(T_r)$. Then, for any $\hat{p} \in \mathcal{P}_d(T_r)$ it holds

$$\begin{aligned}\|\hat{v} - \hat{l}(\hat{v})\|_{H^1(T_r)} &= \|\hat{v} - \hat{p} - \hat{l}(\hat{v}) + \hat{l}(\hat{p})\|_{H^1(T_r)} \\ &= \|(id - \hat{l})(\hat{v} - \hat{p})\|_{H^1(T_r)} \\ &\leq \|id - \hat{l}\|_{\mathcal{L}(H^{d+1}(T_r);H^1(T_r))}\|\hat{v} - \hat{p}\|_{H^{d+1}(T_r)}.\end{aligned}$$

Here, $\mathcal{L}(H^{d+1}(T_r); H^1(T_r))$ is the space of linear and continuous transformations, mapping from $H^{d+1}(T_r)$ to $H^1(T_r)$.
From this the claim follows. □

**Lemma 3.4.3.** (Deny-Lions) [44, Lem. 4.5] *Let $d \geq 0$. Then, there exists a constant $C > 0$, depending on $d$ and $T_r$, such that*

$$\inf_{\hat{p} \in \mathcal{P}_d(T_r)} \|\hat{v} - \hat{p}\|_{H^{d+1}(T_r)} \leq C|\hat{v}|_{H^{d+1}(T_r)} \quad \forall \hat{v} \in H^{d+1}(T_r).$$

*Proof.* For a detailed proof of this Lemma, see [47, Prop. 3.4.4]. □

Combining Lemma 3.4.2 and Lemma 3.4.3, together with Proposition 3.4.1 gives a local interpolation result:

**Theorem 3.4.4.** [44, Thm. 4.4] *Let $d \geq 1$. Then there exists a constant $C > 0$, depending on $d$ and $T_r$, such that for all $T \in \mathcal{T}$ it holds,*

$$|v - \Pi_T^d v|_{H^1(T)} \leq C \frac{h_T^{d+1}}{\rho_T} |v|_{H^{d+1}(T)} \quad \forall v \in H^{d+1}(T).$$

*Proof.* Let $v \in H^{d+1}$. For $d \geq 1$, the relation $H^{d+1}(T) \subset C^0(T)$ is fulfilled (see [43]), therefore the interpolation operator realizes the assumptions of Lemma 3.4.2. Consequently, it holds for all $T \in \mathcal{T}$,

$$\begin{aligned}
|v - \Pi_T^d v|_{H^1(T)} &\leq C\|B_T^{-1}\||detB_T|^{\frac{1}{2}}|\hat{v} - \Pi_{T_r}^d \hat{v}|_{H^1(T_r)} \\
&\leq C\frac{1}{\rho_T}|detB_T|^{\frac{1}{2}}|\hat{v} - \Pi_{T_r}^d \hat{v}|_{H^1(T_r)} \\
&\leq C\frac{1}{\rho_T}|detB_T|^{\frac{1}{2}}|\hat{v}|_{H^1(T_r)} \\
C\frac{1}{\rho_T}\|B_T\|^{d+1}|v|_{H^1(T)} \\
&\leq C\frac{h_T^{d+1}}{\rho_T}|v|_{H^{d+1}(T)}.
\end{aligned}$$

□

Globalizing this result yields:

**Theorem 3.4.5.** [44, Thm. 4.5] *Let $d \geq 1$ and $\{\mathcal{T}_h\}$ be a family of regular meshes of $\Omega$ with respective mesh size $h > 0$. Then there exists a constant $C > 0$, independent of $h$, such that*

$$|v - \Pi_h^d v|_{H^1(\Omega)} \leq Ch^d |v|_{H^{d+1}(\Omega)} \quad \forall v \in H^{d+1}(\Omega).$$

*Proof.* Let $v \in H^{d+1}(\Omega)$. Then,

$$\begin{aligned}
|v - \Pi_h^d v|^2_{H^1(\Omega)} &= \sum_{T \in \mathcal{T}_h} |v - \Pi_T^d v|^2_{H^1(T)} \\
&\leq C \sum_{T \in \mathcal{T}_h} \left( \frac{h_T^{d+1}}{\rho_T} \right)^2 |v|^2_{H^{d+1}(T)} \\
&\leq C \sum_{T \in \mathcal{T}_h} \delta^2 h_T^{2d} |v|^2_{H^{d+1}(T)} \\
&\leq C h^{2d} |v|^2_{H^{d+1}(\Omega)}.
\end{aligned}$$

Here, $\delta$ denotes the shape-regularity constant of the mesh. From this, the claim follows. □

From Theorem 3.2.3 (Céa Lemma) and Theorem 3.4.5 the final statement follows:

**Corollary 3.4.6.** [44, Thm. 4.6] *Let $u \in V$ be the solution of the variational problem (3.7) and $u_h$ its finite element approximation using a subspace consisting of element-wise $\mathcal{P}_d$ functions for $d \geq 1$. If, in addition, $u \in H^{d+1}(\Omega)$, there exists a constant $C > 0$ such that*

$$\|u - u_h\|_{H^1(\Omega)} \leq \frac{M}{\alpha} C h^d |u|_{H^{d+1}(\Omega)}$$

*with $M$ and $\alpha$ denoting the continuity and coercivity constants of the bilinear form associated to the problem.*

*Remark.* Note, that in Corollary 3.4.6 it has to be assumed that $u \in H^{d+1}(\Omega)$. It is possible to reduce this assumption to $u \in H^1(\Omega)$ if one defines other interpolation operators by averaging the nodal values over a local region of the mesh. One example is the operator of Scott and Zhang [49].

Another consequence of Proposition 3.4.1 is a statement about the condition number of the stiffness matrix $A$.

**Theorem 3.4.7.** [47] *Let $A$ be the stiffness matrix generated by the FEM and $h > 0$ the global mesh size of a quasi-uniform triangulation $\mathcal{T}$ of $\Omega$. Then, there exists a constant $C > 0$, independent of $h$, such that*

$$\kappa_2(A) \leq C h^{-2}.$$

*Proof.* The proof of this result uses inverse estimates for piecewise-polynomial functions of a quasi-uniform mesh and is rather technical, so it is omitted here. More details can be found in [47, Prop. 6.3.2.]. □

## 3.5 Generalization to Vector Fields

Up to this point, all statements and theorems have been formulated for scalar valued functions $u \in H^1(\Omega;\mathbb{R})$ or $u \in H_0^1(\Omega;\mathbb{R})$. The aim of this section is to adapt these concepts for vector fields $\mathbf{u} : \mathbb{R}^3 \to \mathbb{R}^3$. The Sobolev theory introduced in Chapter 3.1 carries over naturally by assuming that each component of $\mathbf{u}(\mathbf{x}) = (u_1(\mathbf{x}), u_2(\mathbf{x}), u_3(\mathbf{x}))$ satisfies $u_i \in H^1(\Omega)$ or $u_i \in H_0^1(\Omega)$. As a consequence, we can write $\mathbf{u} \in (H^1(\Omega))^3$ or $\mathbf{u} \in (H_0^1(\Omega))^3$.
Of greater importance is the discretization of the subspaces $V$. For this, let $\mathcal{T}_h$ be a mesh of $\Omega$ with global mesh size $h > 0$. Then, we define the space of vector-valued $\mathcal{T}_h$-piecewise affine and globally continuous polynomials of degree $d$ as [50]

$$\mathbf{V}_h^d \coloneqq \{\mathbf{v}_h \in C(\Omega;\mathbb{R}^3) \mid \mathbf{v}_h|_T \in (\mathcal{P}_d(T))^3 \quad \forall T \in \mathcal{T}_h\}. \tag{3.15}$$

Analogously,

$$\mathring{\mathbf{V}}_h^d \coloneqq \{\mathbf{v}_h \in \mathbf{V}_h^d \mid v_{h,i}|_{\partial\Omega} = 0 \quad \forall i = 1,2,3\}. \tag{3.16}$$

Let again $\mathcal{N} = \{\mathbf{z}_1, \dots, \mathbf{z}_{N_d}\}$ be the nodes of the mesh and $\mathcal{B} = \{\varphi_i\}_{i=1}^{N_d}$ the corresponding scalar-valued nodal basis. We define the vector-valued nodal basis as [51]

$$\mathcal{B}_v \coloneqq \{\boldsymbol{\phi}_i^\alpha : \Omega \to \mathbb{R}^3 \mid \boldsymbol{\phi}_i^\alpha = \varphi_i \mathbf{e}_\alpha, \quad i = 1,\dots,N_d;\ \alpha = 1,2,3\}.$$

Here, $\mathbf{e}_\alpha$ denote the canonical unit vectors in $\mathbb{R}^3$. For simplicity, we often write $\mathcal{B}_v = \{\boldsymbol{\phi}_i\}_{i=1}^{D}$ with $D = 3N_d$ and a coordinate-wise ordering. A vector field can then be represented in $\mathbf{V}_h^d$ as

$$\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{D} c_i \boldsymbol{\phi}_i(\mathbf{x})$$

with scalar coefficients $c_i$. Alternatively, a representation through the scalar-valued basis is also possible:

$$\mathbf{u} = \left( \sum_{i=1}^{N_d} c_{i,1}\varphi_i, \sum_{i=1}^{N_d} c_{i,2}\varphi_i, \sum_{i=1}^{N_d} c_{i,3}\varphi_i \right)^T.$$

Using this ansatz for solving PDEs with the FEM we usually obtain an unknown coefficient vector

$$\mathbf{c} = (c_{1,1}, \dots, c_{N_d,1}, c_{1,2}, \dots, c_{N_d,2}, c_{1,3}, \dots, c_{N_d,3})^T \in \mathbb{R}^D$$

and a $D \times D$ stiffness matrix, which has a block diagonal form with three $N_d \times N_d$ sparse and symmetric blocks on the diagonal. Figure 3.3 illustrates the sparsity pattern of such a matrix.

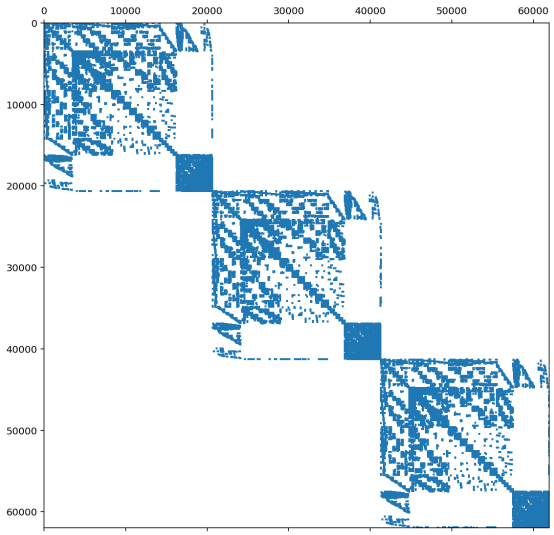


Figure 3.3: Sparsity pattern of a $61941 \times 61941$ stiffness matrix of a vector valued Poisson problem. The symmetric, coordinate-wise block structure is clearly visible.

## 3.6 Remark on the Applicability of classical FEM Theory

The results presented in this chapter provide the standard theoretical framework for finite element discretizations of linear elliptic boundary value problems. In particular, the Galerkin formulation, together with the Lax–Milgram lemma and Céa's Lemma, ensures well-posedness and applicability of the approximation properties presented in Section 3.4 in a Hilbert space setting.

However, the micromagnetic energy minimization problem considered in this work does not fit directly into this framework. The total energy functional is nonlinear, vector-valued, and subject to a pointwise non-convex constraint. Moreover, the coupling between the magnetization and the vector potential introduces nonlinear terms, which leads to a variational problem that cannot be formulated as a coercive bilinear form on a linear space. In particular, the admissible set is not convex, and standard arguments based on Lax–Milgram theory are not applicable to the full problem.

From a mathematical point of view, this results in a regularity gap compared to classical elliptic problems. Existence of minimizers, the analysis of uniqueness, regularity and convergence of the finite element approximations are significantly more involved. Rigorous convergence results for such problems rely on advanced techniques and have been studied, for example, in the context of harmonic maps [52], [53].

Therefore, the theory presented in this chapter should be understood as a foundational tool for the construction and approximation properties of the discrete spaces, rather than as a complete convergence theory for the full micromagnetic minimization problem.

# 4 Finite Element Discretization

As seen in Chapter 3, the FEM reduces finding solutions of PDEs to solving linear systems by forming appropriate function spaces and expanding with their corresponding basis. The aim of this section is to use this to discretize the total Gibbs energy in the setting of the NIST $\mu$MAG standard problem #3, i.e. the energy terms involved in (2.25).
Therefore, we employ a joint discretization scheme, in which both the magnetization and the vector potential are expressed by their finite element ansatz simultaneously. Conversely, in the second part of this chapter alternating minimization methods are tested.
The following computations are largely based on the derivations in [11].

## 4.1 Joint Minimization

Let $\Omega \subset \mathbb{R}^3$ be the magnetic domain. A significant challenge for the numerical computation arises from the integration over the whole space $\mathbb{R}^3$ in (2.25), since the FEM can only handle finite domains. To overcome this obstacle, we approximate $\mathbb{R}^3$ by a finite cube $K$ in which $\Omega$ is embedded [36]. This region is assumed to have zero magnetization, and homogeneous Dirichlet boundary conditions are imposed on the outer faces of $K$. We denote the inner boundary of $K$ by $\Gamma_I$ and the outer facets by $\Gamma_D$. If $K$ is sufficiently large, it is a reasonable representation of the whole space $\mathbb{R}^3$. However, this method also increases the computational effort, since the vector potential has to be computed on the surrounding region as well, which raises the number of degrees of freedom.
We introduce a triangulation $\mathcal{T}_h$ on both the magnetic domain $\Omega$ and on $K$ with global mesh size $h > 0$. The mesh does not need to be uniform; in fact, it is coarsened progressively with increasing distance to $\Omega$. A detailed description of the mesh is provided in Section 6.1.4.
For the finite element subspace we use the space of piecewise affine, vector-valued $\mathcal{P}_1$-polynomials defined in (3.15), including the Dirichlet boundary conditions:

$$\mathring{\mathbf{V}}_h^1 = \{\mathbf{v}_h \in C(\bar{\Omega} \cup K; \mathbb{R}^3) \mid \mathbf{v}_h|_T \in (\mathcal{P}_1(T)),\ \forall T \in \mathcal{T}_h; \quad v_{h,i}|_{\Gamma_D} = 0,\ \forall i = 1,2,3\}. \tag{4.1}$$

Let $\mathcal{N}_h := \{\mathbf{z}_1, \dots, \mathbf{z}_{N_d}\}$ be the nodes of the mesh and $\mathcal{B}_v = \{\boldsymbol{\phi}_i\}_{i=1}^D$ the corresponding vector-valued nodal basis. Here again $D = 3N_d$.

The magnetization and the vector potential can now be represented as a linear combination of these basis elements by

$$\mathbf{m}(\mathbf{x}) = \sum_{i=1}^{D} \alpha_i \boldsymbol{\phi}_i(\mathbf{x})$$

and

$$\mathbf{A}(\mathbf{x}) = \sum_{i=1}^{D} \beta_i \boldsymbol{\phi}_i(\mathbf{x}),$$

with $\alpha_i, \beta_i \in \mathbb{R}$, $\forall i$. It will be useful to express the components of the vector fields through the scalar-valued basis $\mathcal{B} = \{\varphi_i\}_{i=1}^{N_d}$ as well, so we also write, as in Section 3.5,

$$\mathbf{m}(\mathbf{x}) = \left( \sum_{i=1}^{N_d} \alpha_{i,1}\varphi_i(\mathbf{x}), \sum_{i=1}^{N_d} \alpha_{i,2}\varphi_i(\mathbf{x}), \sum_{i=1}^{N_d} \alpha_{i,3}\varphi_i(\mathbf{x}) \right)^T \tag{4.2}$$

and

$$\mathbf{A}(\mathbf{x}) = \left( \sum_{i=1}^{N_d} \beta_{i,1}\varphi_i(\mathbf{x}), \sum_{i=1}^{N_d} \beta_{i,2}\varphi_i(\mathbf{x}), \sum_{i=1}^{N_d} \beta_{i,3}\varphi_i(\mathbf{x}) \right)^T. \tag{4.3}$$

Plugging this into equation (2.25), we can formulate algebraic systems in terms of the coefficient vectors

$$\mathbf{c_m} \coloneqq (\alpha_{1,1}, \dots, \alpha_{N_d,1}, \alpha_{1,2}, \dots, \alpha_{N_d,2}, \alpha_{1,3}, \dots, \alpha_{N_d,3})^T \in \mathbb{R}^D$$

and

$$\mathbf{c_A} \coloneqq (\beta_{1,1}, \dots, \beta_{N_d,1}, \beta_{1,2}, \dots, \beta_{N_d,2}, \beta_{1,3}, \dots, \beta_{N_d,3})^T \in \mathbb{R}^D.$$

### 4.1.1 Discretization of the Exchange Energy

We start with the discretization of the exchange energy,

$$e_{ex}(\mathbf{m}) = \int_\Omega \tilde{A}_{ex} \|\nabla \mathbf{m}\|_F^2 \, d\mathbf{x} = \int_\Omega \tilde{A}_{ex} \sum_{i=1}^{3} |\nabla m_i|^2 \, d\mathbf{x}.$$

Here, we write $\mathbf{m} = (m_1, m_2, m_3)^T$. Following the approach from [11], we start with assembling the stiffness matrix for a single tetrahedron of the mesh and then build a global one with the use of connectivity matrices. Fix an element $T_k \in \mathcal{T}_h$ and use the ansatz (4.2) to get

$$m_i^{(k)}(\mathbf{x}) = \sum_{s=1}^{4} \alpha_{s,i}\varphi_s(\mathbf{x}), \tag{4.4}$$

where $(\varphi_s)_{s=1}^4$ are the shape functions associated with the four local nodes of $T_k$. Note that there are four degrees of freedom per element due to the use of $\mathcal{P}_1$-finite element spaces. Furthermore, the squared gradients can be expressed as

$$|\nabla m_i^{(k)}|^2 = \left(\sum_{s=1}^4 \alpha_{s,i}\nabla\varphi_s\right)\cdot\left(\sum_{t=1}^4 \alpha_{t,i}\nabla\varphi_t\right) = \sum_{s,t=1}^4 \alpha_{s,i}\alpha_{t,i}\nabla\varphi_s\cdot\nabla\varphi_t,$$

for $i = 1, 2, 3$. Therefore, the exchange energy can locally be described by

$$e_{ex}^{(k)} = \frac{1}{2}\sum_{i,j=1}^3\sum_{s,t=1}^4 \alpha_{s,i}(F_{ex}^{(k)})_{si,tj}\alpha_{t,j},$$

with the local $12\times 12$ stiffness matrix $F_{ex}^{(k)}$ with entries:

$$(F_{ex}^{(k)})_{si,tj} = 2\tilde{A}_{ex}\left(\int_{T_k}\nabla\varphi_s\cdot\nabla\varphi_t\,d\mathbf{x}\right)\delta_{ij}.$$

Here $\delta_{ij}$ denotes the Kronecker Delta. Thus, $F_{ex}^{(k)}$ is a block diagonal matrix with three $4\times 4$ diagonal blocks:

$$F_{ex}^{(k)} = 2\tilde{A}_{ex}\begin{bmatrix} S^{(k)} & 0 & 0\\ 0 & S^{(k)} & 0\\ 0 & 0 & S^{(k)}\end{bmatrix}$$

with $S^{(k)}\in\mathbb{R}^{4\times 4}$ and entries $(S^{(k)})_{st} = \int_{T_k}\nabla\varphi_s\cdot\nabla\varphi_t\,d\mathbf{x}, \quad s,t = 1,\dots,4.$
To globally extend this stiffness matrix, we multiply with connectivity matrices $C^{(k)}\in\mathbb{R}^{N_d\times 4}$ with entries

$$C_{ls}^{(k)} = \begin{cases} 1 & \text{if the global node } l \text{ corresponds to a local node } s \text{ on the element } T_k,\\ 0 & \text{else}\end{cases}$$

in the following way

$$S_{mn} = \sum_{k=1}^E\sum_{s,t=1}^4 C_{ms}^{(k)}S_{st}^{(k)}C_{nt}^{(k)}.$$

Consequently, the entries of $S$ reflect the topology of the mesh, since $S_{mn}$ is nonzero only when the nodes $m$ and $n$ share an edge of a mesh element.
The resulting global stiffness matrix is

$$F_{ex} = 2\tilde{A}_{ex}\begin{bmatrix} S & 0 & 0\\ 0 & S & 0\\ 0 & 0 & S\end{bmatrix}\in\mathbb{R}^{D\times D}$$

and the discretized energy becomes

$$\tilde{e}_{ex}(\mathbf{c_m}) = \frac{1}{2}\mathbf{c_m}^T F_{ex}\mathbf{c_m}.$$

### 4.1.2 Discretization of the Anisotropy Energy

Next, we discretize the magnetocrystalline anisotropy energy

$$e_a(\mathbf{m}) = \int_\Omega Q(1 - (\mathbf{m} \cdot \mathbf{a})^2)\, d\mathbf{x}.$$

In the minimization process the first, constant term can be neglected, thus, we only consider

$$\min_{\mathbf{m}} \left[ -Q \int_\Omega (\mathbf{m} \cdot \mathbf{a})^2\, d\mathbf{x} \right].$$

We proceed analogously to the discretization of the exchange energy. Fixing a single mesh element $T_k$ and plugging in the local ansatz for the magnetization leads to

$$e_a^{(k)} = \frac{1}{2} \sum_{i,j=1}^{3} \sum_{s,t=1}^{4} \alpha_{s,i} (F_a^{(k)})_{si,tj} \alpha_{t,j}$$

with the $12 \times 12$ stiffness matrix $F_a^{(k)}$ consisting of

$$(F_a^{(k)})_{si,tj} = -2Q a_i a_j \int_{T_k} \varphi_s \varphi_t\, d\mathbf{x}.$$

Since the easy axis $\mathbf{a} = (a_1, a_2, a_3)^T$ is parallel to the z-axis, we choose $\mathbf{a} = (0, 0, 1)^T$ and the block diagonal form of $F_a^{(k)}$ reduces to

$$F_a^{(k)} = -2Q \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & M^{(k)} \end{bmatrix},$$

with $M^{(k)} \in \mathbb{R}^{4\times 4}$ and entries $(M^{(k)})_{st} = \int_{T_k} \varphi_s \varphi_t\, d\mathbf{x} \quad s, t = 1, \ldots, 4$.
Again, we globalize the results by multiplying with connectivity matrices, i.e.

$$M_{mn} = \sum_{k=1}^{E} \sum_{s,t=1}^{4} C_{ms}^{(k)} M_{st}^{(k)} C_{nt}^{(k)}. \tag{4.5}$$

We arrive at a global stiffness matrix

$$F_a = -2Q \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & M \end{bmatrix} \in \mathbb{R}^{D\times D}$$

and the corresponding reduced discrete energy functional

$$\tilde{e}_a(\mathbf{c_m}) = \frac{1}{2} \mathbf{c_m}^T F_a \mathbf{c_m}$$

which differs from the full discrete anisotropy energy only by an additive constant.

### 4.1.3 Discretization of the Stray Field Energy

Next, we deal with the upper bound for the stray field energy

$$e_{\mathbf{A}}(\mathbf{m}) = \int_{\Omega} \|\mathbf{m}\|^2 d\mathbf{x} + \int_{K} \|\nabla \mathbf{A}\|^2 \, d\mathbf{x} - 2 \int_{\Omega} \mathbf{m} \cdot (\nabla \times \mathbf{A}) \, d\mathbf{x}, \tag{4.6}$$

where we already reduced the integral over the whole space to the model cube $K$. The reformulation of the first two terms, which are isolated in $\mathbf{m}$ and $\mathbf{A}$ respectively, is similar to the sections before. For the quadratic term in $\mathbf{m}$ we use again the local ansatz (4.4) on an element $T_k$ and get

$$\|\mathbf{m}^{(k)}\|^2 = (m_1^{(k)})^2 + (m_2^{(k)})^2 + (m_3^{(k)})^2 = \sum_{i=1}^{3} \sum_{s,t=1}^{4} \alpha_{s,i} \alpha_{t,i} \varphi_s \varphi_t.$$

It follows that

$$\int_{T_k} \|\mathbf{m}\|^2 \, d\mathbf{x} = \sum_{i=1}^{3} \sum_{s,t=1}^{4} \alpha_{s,i} \alpha_{t,i} \int_{T_k} \varphi_s \varphi_t \, d\mathbf{x} = \sum_{i,j=1}^{3} \sum_{s,t=1}^{4} \alpha_{s,i} (F_d^{(k)})_{si,tj} \alpha_{t,j},$$

with the local stiffness matrix $F_d^{(k)} \in \mathbb{R}^{12 \times 12}$ having entries

$$(F_d^{(k)})_{si,tj} = \left( \int_{T_k} \varphi_s \varphi_t \, d\mathbf{x} \right) \delta_{ij}.$$

Again $F_d^{(k)}$ is a block diagonal matrix of the form

$$F_d^{(k)} = \begin{bmatrix} M^{(k)} & 0 & 0 \\ 0 & M^{(k)} & 0 \\ 0 & 0 & M^{(k)} \end{bmatrix}, \tag{4.7}$$

with $M^{(k)}$ defined as before through $(M^{(k)})_{st} = \int_{T_k} \varphi_s \varphi_t \, d\mathbf{x}, \quad s,t = 1, \ldots, 4.$
On the other hand, using a local ansatz

$$A_i^{(k)}(\mathbf{x}) = \sum_{s=1}^{4} \beta_{s,i} \varphi_s(\mathbf{x}) \tag{4.8}$$

on an element $T_k \in \mathcal{T}_h$ yields

$$\int_{T_k} \nabla \mathbf{A} : \nabla \mathbf{A} \, d\mathbf{x} = \sum_{i=1}^{3} \sum_{s,t=1}^{4} \beta_{s,i} \beta_{t,i} \int_{T_k} \nabla \varphi_s \cdot \nabla \varphi_t \, d\mathbf{x} = \sum_{i,j=1}^{3} \sum_{s,t=1}^{4} \beta_{s,i} (F_A^{(k)})_{si,tj} \beta_{t,j}.$$

As before, $F_A^{(k)}$ is a $12 \times 12$ block diagonal matrix of the form

$$F_A^{(k)} = \begin{bmatrix} B^{(k)} & 0 & 0 \\ 0 & B^{(k)} & 0 \\ 0 & 0 & B^{(k)} \end{bmatrix},$$

with $B \in \mathbb{R}^{4\times 4}$ and entries

$$B_{st}^{(k)} = \int_{T_k} \nabla\varphi_s \cdot \nabla\varphi_t \, d\mathbf{x}.$$

Next, we look at the 'mixed' term

$$-2\int_{\Omega} \mathbf{m} \cdot (\nabla \times \mathbf{A}) \, d\mathbf{x}.$$

Using both local formulations (4.4) and (4.8) and the definition of the curl of a vector field $\mathbf{F} \coloneqq (F_1, F_2, F_3)^T$,

$$\nabla \times \mathbf{F} = (\partial_y F_3 - \partial_z F_2, \partial_z F_1 - \partial_x F_3, \partial_x F_2 - \partial_y F_1)^T,$$

we get

$$\begin{aligned}
\int_{T_k} \mathbf{m} \cdot (\nabla \times \mathbf{A}) \, d\mathbf{x} &= \sum_{s,t=1}^{4} \int_{T_k} \alpha_{s,1}\varphi_s(\beta_{t,3}\partial_y\varphi_j - \beta_{t,2}\partial_z\varphi_j) \, d\mathbf{x} \\
&+ \sum_{s,t=1}^{4} \int_{T_k} \alpha_{s,2}\varphi_s(\beta_{t,1}\partial_z\varphi_j - \beta_{t,3}\partial_x\varphi_j) \, d\mathbf{x} \\
&+ \sum_{s,t=1}^{4} \int_{T_k} \alpha_{s,3}\varphi_s(\beta_{t,2}\partial_x\varphi_j - \beta_{t,1}\partial_y\varphi_j) \, d\mathbf{x} \\
&= \sum_{i,j=1}^{3} \sum_{s,t=1}^{4} \alpha_{s,i}(K^{(k)})_{si,tj}\beta_{t,j}.
\end{aligned}$$

Here,

$$K^{(k)} = \begin{bmatrix} 0 & -Z^{(k)} & Y^{(k)} \\ Z^{(k)} & 0 & -X^{(k)} \\ -Y^{(k)} & X^{(k)} & 0 \end{bmatrix} \in \mathbb{R}^{12\times 12},$$

with sub-matrices $X^{(k)}, Y^{(k)}, Z^{(k)} \in R^{4\times 4}$ with entries

$$\begin{aligned}
X_{ij}^{(k)} &= \int_{T_k} \varphi_i \partial_x \varphi_j \, d\mathbf{x}, \\
Y_{ij}^{(k)} &= \int_{T_k} \varphi_i \partial_y \varphi_j \, d\mathbf{x}, \\
Z_{ij}^{(k)} &= \int_{T_k} \varphi_i \partial_z \varphi_j \, d\mathbf{x},
\end{aligned}$$

for $i, j = 1, \ldots, 4$. After globalization of all the involved matrices like in the sections before, we end up with a discretized upper bound of the stray field energy

$$\tilde{e}_{\mathbf{A}}(\mathbf{c_m}, \mathbf{c_A}) = \mathbf{c_m}^T F_d \mathbf{c_m} + \mathbf{c_A}^T F_A \mathbf{c_A} - 2\mathbf{c_m}^T K \mathbf{c_A}.$$

### 4.1.4 The discretized Total Gibbs Free Energy

Combining the derived energy terms leads to a fully discretized formulation

$$\tilde{e}_{NIST}(\mathbf{c_m}, \mathbf{c_A}) = \frac{1}{2}\mathbf{c_m}^T H \mathbf{c_m} + \frac{1}{2}\mathbf{c_A}^T F_A \mathbf{c_A} - \mathbf{c_m}^T 2K \mathbf{c_A}. \tag{4.9}$$

Here, the matrix $H = F_d + F_a + F_{ex} \in \mathbb{R}^{D\times D}$ represents the summed entries for the isolated magnetization terms. This can be further simplified by defining $\mathbf{c} \coloneqq (\mathbf{c_m}, \mathbf{c_A}) \in \mathbb{R}^{2D}$ and

$$\tilde{H} \coloneqq \begin{bmatrix} H & 0 \\ 0 & 0 \end{bmatrix}, \quad \tilde{F} \coloneqq \begin{bmatrix} 0 & 0 \\ 0 & F_A \end{bmatrix}, \quad \tilde{K} \coloneqq \frac{1}{2}\begin{bmatrix} 0 & K \\ K^T & 0 \end{bmatrix} \in \mathbb{R}^{2D\times 2D}.$$

Let now $\mathcal{I} \coloneqq \{1, \dots, N_d\}$ be the set of indices corresponding to the degrees of freedom of the mesh and assume $N \leq N_d$ nodes are part of the magnetic region $\Omega$. Without loss of generality, denote by $\mathcal{I}_\Omega \coloneqq \{1, \dots, N\}$ the set of the node indices inside of $\Omega$.
The full discretized joint minimization problem reads now as

$$\min G(\mathbf{c}) \coloneqq \frac{1}{2}\mathbf{c}^T(\tilde{H} + \tilde{F} - 2\tilde{K})\mathbf{c} \tag{4.10}$$

subject to

$$\mathbf{c} = (\mathbf{c_m}, \mathbf{c_A}) \in \mathbb{R}^{2D}, \quad \begin{cases} \|\mathbf{c}_{\mathbf{m},i}\| = \sqrt{\alpha_{i,1}^2 + \alpha_{i,2}^2 + \alpha_{i,3}^2} = 1 & \text{for } i \in \mathcal{I}_\Omega \\ \mathbf{c}_{\mathbf{m},j} = 0 & \text{else.} \end{cases} \tag{4.11}$$

## 4.2 Alternating Minimization

In contrast to the simultaneous minimization, we now consider a novel alternating minimization approach in order to simplify the occurring energy terms. To calculate the minimum of (2.25) we can start with a fixed magnetization and minimize the upper bound of the stray field energy $e_{\mathbf{A}}$ (4.6) with respect to the vector potential alone. Then the optimization problem for the total energy (2.25) is solved for a new magnetization using the previously calculated vector potential as a parameter. This process is repeated until a total minimum for both the magnetization and the vector potential is reached. Algorithm 1 depicts this alternating scheme. In this setting, the discretization and the resulting algebraic systems change as well.

### 4.2.1 Discretization for fixed Magnetization

Let $\Omega \subset \mathbb{R}^3$ be the magnetic domain and let $\mathbf{m} : \mathbb{R}^3 \to \mathbb{R}^3$ be a fixed magnetization such that $\mathbf{m} = 0$ in $\mathbb{R}^3 \backslash \Omega$. In order to minimize the Gibbs free energy functional (2.25), it is now only necessary to consider the problem

$$\min_{\mathbf{A}} \frac{1}{2}\left( \int_K \|\nabla \mathbf{A}\|^2 \, d\mathbf{x} - 2\int_\Omega \mathbf{m} \cdot (\nabla \times \mathbf{A}) \, d\mathbf{x} \right).$$

**Algorithm 1** Alternating Minimization Scheme

Task: minimize total energy $e_{NIST}(\mathbf{m}, \mathbf{A})$ depicted in (2.25), subject to $\|\mathbf{m}\| = 1$ in the magnetic domain $\Omega$.

1: Choose initial magnetization $\mathbf{m}_0$.
2: **for** $k = 0, 1, 2, \dots$ **do**:
3: $\quad \mathbf{A}_k \leftarrow \operatorname{argmin}_{\mathbf{A}} \frac{1}{2} e_{\mathbf{A}}(\mathbf{m}_k, \mathbf{A})$
4: $\quad \mathbf{m}_{k+1} \leftarrow \operatorname{argmin}_{\mathbf{m}} e_{NIST}(\mathbf{m}, \mathbf{A}_k), \quad s.t. \|\mathbf{m}\| = 1 \text{ in } \Omega$
5: $\quad k \leftarrow k + 1$
6: **end for**

Again, the full space integral is replaced by a finite cube $K$, modeling $\mathbb{R}^3$. Analogous to the simultaneous minimization, the vector potential is written in terms of scalar basis functions of the finite element space as in (4.3) and the discretization of the quadratic term leads to

$$\int_K \|\nabla \mathbf{A}\|^2 \, d\mathbf{x} = \mathbf{c}_{\mathbf{A}}^T F_A \mathbf{c}_{\mathbf{A}},$$

with $F_A$ as in Section 4.1.3. The remaining term

$$\int_\Omega \mathbf{m} \cdot (\nabla \times \mathbf{A}) \, d\mathbf{x}$$

is now linear in $\mathbf{A}$ and therefore, plugging in the ansatz yields

$$\begin{aligned} \mathbf{m} \cdot (\nabla \times \mathbf{A}) = &\sum_{j=1}^{N_d} m_1(\beta_{j,3}\partial_y \varphi_j - \beta_{j,2}\partial_z \varphi_j) \\ &+ \sum_{j=1}^{N_d} m_2(\beta_{j,1}\partial_z \varphi_j - \beta_{j,3}\partial_x \varphi_j) \\ &+ \sum_{j=1}^{N_d} m_3(\beta_{j,2}\partial_x \varphi_j - \beta_{j,1}\partial_y \varphi_j). \end{aligned}$$

This leads to

$$\int_\Omega \mathbf{m} \cdot (\nabla \times \mathbf{A}) \, d\mathbf{x} = \mathbf{f}^T \mathbf{c}_{\mathbf{A}},$$

with

$$\mathbf{f} = (\mathbf{f}^{(1)}, \mathbf{f}^{(2)}, \mathbf{f}^{(3)})^T \in \mathbb{R}^D,$$

with entries

$$f_i^{(1)} = \int_\Omega m_2 \partial_z \varphi_i - m_3 \partial_y \varphi_i \, d\mathbf{x},$$
$$f_i^{(2)} = \int_\Omega m_3 \partial_x \varphi_i - m_1 \partial_z \varphi_i \, d\mathbf{x},$$
$$f_i^{(3)} = \int_\Omega m_1 \partial_y \varphi_i - m_2 \partial_x \varphi_i \, d\mathbf{x}$$

for $i = 1, \ldots, N_d$.
The final quadratic problem reads

$$\min_{\mathbf{c_A} \in \mathbb{R}^D} \frac{1}{2} \mathbf{c_A}^T F_A \mathbf{c_A} - \mathbf{f}^T \mathbf{c_A}. \tag{4.12}$$

### 4.2.2 Discretization for fixed Vector Potential

For a fixed vector potential the discretization of the stray field energy reduces to reformulating

$$\int_\Omega \|\mathbf{m}\|^2 d\mathbf{x} - 2 \int_\Omega \mathbf{m} \cdot (\nabla \times \mathbf{A}) \, d\mathbf{x}.$$

Again, the quadratic term is derived as in the sections before, so we only consider the linear part.
Thus, we obtain

$$\begin{aligned} \int_\Omega \mathbf{m} \cdot (\nabla \times \mathbf{A}) \, d\mathbf{x} &= \int_\Omega m_1 (\nabla \times \mathbf{A})_1 + m_2 (\nabla \times \mathbf{A})_2 + m_3 (\nabla \times \mathbf{A})_3 \, d\mathbf{x} \\ &= \int_\Omega \sum_{i=1}^{3} \left( \sum_{s=1}^{N_d} \alpha_{s,i} \varphi_s \right) (\nabla \times \mathbf{A})_i \, d\mathbf{x}. \end{aligned}$$

This expression can be written as an inner product

$$\mathbf{g}^T \mathbf{c_m}$$

with a load vector

$$\mathbf{g} = (\mathbf{g}^{(1)}, \mathbf{g}^{(2)}, \mathbf{g}^{(3)})^T \in \mathbb{R}^D,$$

with entries

$$g_j^{(i)} = \int_\Omega \varphi_j (\nabla \times \mathbf{A}_i) \, d\mathbf{x}.$$

For the exchange and anisotropy energy, the calculations remain the same as in Sections 4.1.1 and 4.1.2. To conclude, the final constraint problem reads:

$$\min_{\mathbf{c_m} \in \mathbb{R}^D} \frac{1}{2} \mathbf{c_m}^T H \mathbf{c_m} - \mathbf{g}^T \mathbf{c_m} \tag{4.13}$$

subject to

$$\begin{cases} \|\mathbf{c_m}_{,i}\| = \sqrt{\alpha_{i,1}^2 + \alpha_{i,2}^2 + \alpha_{i,3}^2} = 1 & \text{for } i \in \mathcal{I}_\Omega \\ \mathbf{c_m}_{,j} = 0 & \text{else.} \end{cases} \tag{4.14}$$

Here again,

$$H = F_d + F_a + F_{ex}$$

is the combined stiffness matrix and $\mathcal{I}_\Omega$ denotes the set of node indices within the magnetic domain.

# 5 Numerical Methods for Energy Minimization

The aim of this chapter is to provide the mathematical background for the optimization techniques used in the numerical computations. The focus is on establishing variants of the quadratic penalty method, the augmented Lagrangian method and conjugate gradient method for solving large linear systems with a nonlinear unit norm constraint efficiently.

## 5.1 Penalty Methods

In the course of this thesis, different methods of solving the constraint problem (4.10)-(4.11) will be observed. As a starting point serves the classical scheme of penalty methods. The mathematical theory of this section can be found in [54], whereas [39] deals with the micromagnetic adaptions.
The main motivation to use penalty type methods is to transform a constraint optimization problem into an unconstrained one by introducing additional terms to the objective function in order to penalize violations of the constraints. More specifically, consider the problem

$$\min_{\mathbf{c}\in X} G(\mathbf{c}) \quad s.t. \quad h_i(\mathbf{c}) = 0 \quad \forall i \in I. \tag{5.1}$$

Here, $I = \{1, \dots, m\}$ and $\mathbf{h}(\mathbf{c}) \coloneqq (h_1(\mathbf{c}), \dots, h_m(\mathbf{c}))^T$ represents the nonlinear constraints. In the context of problem (4.10) we have $X = \mathbb{R}^{2D}$,

$$G(\mathbf{c}) = G(\mathbf{c_m}, \mathbf{c_A}) = \frac{1}{2}\mathbf{c}^T(\tilde{H} + \tilde{F} - 2\tilde{K})\mathbf{c}$$

and

$$h_i(\mathbf{c}) = \begin{cases} \frac{1}{2}(\|\mathbf{c_{m},}_i\|^2 - 1) & \forall i \in \mathcal{I}_\Omega, \\ \frac{1}{2}\|\mathbf{c_{m},}_i\|^2 & \text{else}, \end{cases} \tag{5.2}$$

with $\mathcal{I}_\Omega$ being the node indices inside the magnet, so the unit norm constraints only act on the coefficients of the magnetization.
Since we are solely dealing with equality constraints, it will be sufficient to consider problem (5.1), but in general it is also possible to treat inequality conditions with a

penalty scheme. See [54] for further reading on this topic.
The *(quadratic) penalty function* associated with problem (5.1) is defined by

$$Q(\mathbf{c};\mu) \coloneqq G(\mathbf{c}) + \frac{\mu}{2}\sum_{i\in I} h_i(\mathbf{c})^2.$$

Here, $\mu > 0$ is the *penalty parameter* which determines the violation of the constraints. In the micromagnetic framework, we get

$$Q(\mathbf{c};\mu) \coloneqq \frac{1}{2}\mathbf{c}^T(\tilde{H}+\tilde{F}-2\tilde{K})\mathbf{c} + \frac{\mu}{2}\sum_{i\in\mathcal{I}_\Omega}\left(\frac{1}{2}(\|\mathbf{c}_{\mathbf{m},i}\|^2-1)\right)^2 + \frac{\mu}{2}\sum_{i\in I\setminus\mathcal{I}_\Omega}\left(\frac{1}{2}\|\mathbf{c}_{\mathbf{m},i}\|^2\right)^2.$$

The benefit of introducing this formulation is that by driving $\mu$ to infinity and successively minimizing the corresponding penalty function, the iterates of the resulting scheme approach the solution of problem (5.1) from the infeasible region of the solution space. This means, instead of solving a constraint problem, a sequence of unconstrained problems can be solved.
One has to be careful in choosing the penalty parameter $\mu$. In general, one starts at a relatively small $\mu_0$ and generates a sequence of increasing parameters, solving a subproblem for every one of them. As the sequence tends to infinity, the constraints get penalized more heavily, forcing the optimal value to fulfill them. However, a too large penalty parameter results in an ill-conditioned Hessian of the objective function (see the discussion below). Therefore, there are several ways to choose a new penalty parameter $\mu_{k+1}$ adaptively by taking into account the complexity of the minimization subproblem. If the optimization step was expensive, the new penalty parameter should be chosen not much larger than the current one, if not, a more ambitious step can be made in order to improve the global convergence speed. In the view of problem (4.10), we measure the difference in constraint violations compared with the previous step. If the improvement was not good enough, $\mu$ is multiplied by a scaling constant $M = 5$, otherwise the parameter is preserved for the next step. A general algorithmic framework is depicted in Algorithm 2. Convergence of the method is shown in the following theorem.

**Theorem 5.1.1.** [54, Thm. 17.2], [39, Thm. 1]: *Assume a framework as in Algorithm 2 with $\{\tau_k\}_{k\in\mathbb{N}} \to 0$ and $\{\mu_k\}_{k\in\mathbb{N}} \to \infty$. Then, for the limit $\mathbf{c}^*$ of the generated sequence $\{\mathbf{c}_k\}_{k\in\mathbb{N}}$ it holds:*

*i) If $\mathbf{c}^*$ is infeasible ($\mathbf{h}(\mathbf{c}^*) \neq 0$), $\mathbf{c}^*$ is a stationary point of the function $\mathbf{c} \mapsto \|\mathbf{h}(\mathbf{c})\|^2$.*

*ii) If $\mathbf{c}^*$ is feasible and $\nabla h_i(\mathbf{c}^*)$ are linearly independent for all $i \in I$, $\mathbf{c}^*$ is a KKT point of problem (4.10).*

*Remark.* [55] The limit point $\mathbf{c}^*$ being a *Karush-Kuhn-Tucker (KKT)* point of problem (4.10) means that there exists a $\boldsymbol{\lambda}^* \in \mathbb{R}^{N_d}$ such that the *Lagrangian function*

$$\mathcal{L}(\mathbf{c};\boldsymbol{\lambda}) \coloneqq G(\mathbf{c}) - \boldsymbol{\lambda}^T\mathbf{h}(\mathbf{c}),$$

**Algorithm 2** Quadr. Penalty method, adapted from [54, Framework 17.1], [39, Alg. 4]

Choose initial $\mathbf{c}_0 \in \mathbb{R}^{2D}$, $\mu_0 > 0, \gamma \in [0,1)$, a tolerance $\varepsilon > 0$ and a sequence $\{\tau_k\}_{k\in\mathbb{N}} \to 0$, set $k = 0$.
**while** $\|\nabla_{\mathbf{c}} Q(\mathbf{c}_k; \mu_k)\| > \varepsilon$ **do**:
  $\mathbf{x}^* \leftarrow \operatorname{argmin}_{\mathbf{c}} Q(\mathbf{c}; \mu_k)$, starting at $\mathbf{c}_k$ and minimizing until $\|\nabla_{\mathbf{c}} Q(\mathbf{x}^*; \mu_k)\| \leq \tau_k$
  **if** $\|\mathbf{h}(\mathbf{x}^*)\| \leq \gamma \|\mathbf{h}(\mathbf{c}_k)\|$ :
    $\mu_{k+1} \leftarrow \mu_k$
  **else** :
    $\mu_{k+1} \leftarrow 5\mu_k$
  $\mathbf{c}_{k+1} \leftarrow \mathbf{x}^*$
  $k \leftarrow k + 1$
**end while**

with $\boldsymbol{\lambda} \in \mathbb{R}^D$ being the *vector of Lagrange multipliers,* fulfills

$$\nabla_{\mathbf{c}} \mathcal{L}(\mathbf{c}^*, \boldsymbol{\lambda}^*) = \mathbf{0}, \tag{5.3}$$
$$\nabla_{\boldsymbol{\lambda}} \mathcal{L}(\mathbf{c}^*, \boldsymbol{\lambda}^*) = \mathbf{0}. \tag{5.4}$$

*Proof.* Let $k > 0$. It holds:

$$\nabla_{\mathbf{c}} Q(\mathbf{c}_k; \mu_k) = \nabla G(\mathbf{c}_k) + \mu_k \sum_{i\in I} h_i(\mathbf{c}_k) \nabla h_i(\mathbf{c}_k), \tag{5.5}$$

and therefore,

$$\left\| \nabla G(\mathbf{c}_k) + \mu_k \sum_{i\in I} h_i(\mathbf{c}_k) \nabla h_i(\mathbf{c}_k) \right\| \leq \tau_k.$$

This is equivalent to

$$\left\| \sum_{i\in I} h_i(\mathbf{c}_k) \nabla h_i(\mathbf{c}_k) \right\| \leq \frac{1}{\mu_k} (\tau_k + \|\nabla G(\mathbf{c}_k)\|).$$

Taking the limit on both sides of this inequality leads, up to a subsequence, to

$$\sum_{i\in I} h_i(\mathbf{c}^*) \nabla h_i(\mathbf{c}^*) = 0,$$

since $\mu_k \to \infty$ as $k \to \infty$.
If $h_i(\mathbf{c}^*) \neq 0$ for at least one $i \in I$, this is equivalent to $\mathbf{c}^*$ being a stationary point of the function $\|\mathbf{h}(\cdot)\|^2$.
Otherwise, if $\mathbf{h}(\mathbf{c}^*) = \mathbf{0}$, then condition (5.4) is fulfilled by definition. Furthermore, if $\nabla h_i(\mathbf{c}^*)$ are linearly independent for all $i \in I$, the matrix

$$D(\mathbf{c}^*) := [\nabla h_i(\mathbf{c}^*)]_{i\in I}$$

has full rank.
Define

$$\lambda_i^{(k)} := -\mu_k h_i(\mathbf{c}_k).$$

From (5.5) we get

$$\nabla_{\mathbf{c}} Q(\mathbf{c}_k; \mu_k) = \nabla G(\mathbf{c}_k) - D(\mathbf{c}_k)^T \boldsymbol{\lambda}^{(k)}. \tag{5.6}$$

Since $\|\nabla_{\mathbf{c}} Q(\mathbf{c}_k; \mu_k)\|$ is bounded by $\tau_k$, we find a finite limit $\lim_{k\to\infty} \boldsymbol{\lambda}^{(k)} = \boldsymbol{\lambda}^* < \infty$. Equation (5.6) can be restated as

$$\boldsymbol{\lambda}^{(k)} = [D(\mathbf{c}_k)D(\mathbf{c}_k)^T]^{-1} D(\mathbf{c}_k)(\nabla G(\mathbf{c}_k) - \nabla_{\mathbf{c}} Q(\mathbf{c}_k; \mu_k)),$$

and in the limit this becomes

$$\boldsymbol{\lambda}^* = [D(\mathbf{c}^*)D(\mathbf{c}^*)^T]^{-1} D(\mathbf{c}^*)\nabla G(\mathbf{c}^*).$$

Therefore,

$$\nabla G(\mathbf{c}^*) - D(\mathbf{c}^*)^T \boldsymbol{\lambda}^* = 0$$

and condition (5.3) is satisfied as well. □

Theorem 5.1.1 hints that additionally to the global stopping criterion, one has to break the iterative loop earlier, if a stationary point of $\|\mathbf{h}(\cdot)\|^2$ is approached. To this end, the matrix $D(\mathbf{c}) := [\nabla h_1(\mathbf{c}), \ldots, \nabla h_{N_d}(\mathbf{c})] \in \mathbb{R}^{D\times N_d}$ from the proof of Theorem 5.1.1 has to be observed in the context of micromagnetism. This matrix has the form

$$D(\mathbf{c}) = \begin{bmatrix} \mathbf{c}_{\mathbf{m},1} & & \\ & \ddots & \\ & & \mathbf{c}_{\mathbf{m},N_d} \end{bmatrix}$$

and the condition of linear inequality of the gradients holds if and only if $D(\mathbf{c}^*)$ has full rank. A feasible $\mathbf{c}$ always fulfills this condition; thus, feasibility of $\mathbf{c}^*$ is sufficient to get a KKT-point.
On the other hand, infeasibility of $\mathbf{c}^*$ means that $\mathbf{h}(\mathbf{c}^*) \in ker(D(\mathbf{c}^*))$ which is equivalent to

$$h_i(\mathbf{c}^*)\mathbf{c}_{\mathbf{m},i}^* = \mathbf{0} \quad \forall i \in \mathcal{I}_\Omega.$$

Therefore, Algorithm 2 will be safeguarded by restarting the iterations if a point $\mathbf{c}_k$ fulfills

$$\max_i h_i(\mathbf{c}_k)(\mathbf{c}_k)_{\mathbf{m},i} < tol,$$

for a given tolerance $tol > 0$ [39].
Nevertheless, the quadratic penalty method can lead to solving problems where the involved matrices have large condition numbers. Note that in every step of Algorithm 2, the previously calculated solution is used as a starting point for the current minimization. This and a small initial choice of $\mu_0$ resolves some of the ill-conditioning issues, but for

larger penalty parameters, problems may occur. Taking a closer look at the Hessian of the penalty function,

$$\nabla^2_{\mathbf{cc}} Q(\mathbf{c};\mu) = \nabla^2 G(\mathbf{c}) + \sum_{i\in I} \mu h_i(\mathbf{c}) \nabla^2 h_i(\mathbf{c}) + \mu D(\mathbf{c})^T D(\mathbf{c}),$$

reveals that for all $\mathbf{c}$ near a feasible minimizer it holds

$$\nabla^2_{\mathbf{cc}} Q(\mathbf{c};\mu) \approx \nabla^2 \mathcal{L}(\mathbf{c};\boldsymbol{\lambda}^*) + \mu D(\mathbf{c})^T D(\mathbf{c}).$$

In this case, the Hessian consists of two parts - one with eigenvalues independent of $\mu$ and another with eigenvalues of order $\mu$. Thus, pushing $\mu$ to infinity leads to an increase of the condition number of $\nabla^2_{\mathbf{cc}} Q$ by the same factor [54].
Despite this drawback, the method is still useful for our calculations. However, this motivates the construction of programs which tend to increase the penalty parameter more moderately, for example the augmented Lagrangian method.

## 5.2 The Augmented Lagrangian Method

In order to deal with the problem of gradual ill-conditioning, the quadratic penalty method suffers from, we desire more options to achieve convergence to a minimizer other than increasing the penalty parameter. One option is considering a Lagrangian-type formulation of the problem.
Therefore, we take a closer look at Theorem 5.1.1 and observe the following:

**Lemma 5.2.1.** [54, Thm. 17.2] *Assume a framework as in Algorithm 2 with* $\{\tau_k\}_{k\in\mathbb{N}} \to 0$ *and* $\{\mu_k\}_{k\in\mathbb{N}} \to \infty$. *If a limit point* $\mathbf{c}^*$ *of the generated sequence* $\{\mathbf{c}_k\}$ *is a KKT-point of problem (4.10), then for any sequence* $\{\mathbf{c}_k\}_{k\in\mathbb{N}}$ *such that* $\mathbf{c}_k \to \mathbf{c}^*$ *it holds, up to a subsequence, that*

$$\lim_{k\to\infty} -\mu_k h_i(\mathbf{c}_k) = \lambda_i^* \quad \forall i \in I,$$

*with* $\boldsymbol{\lambda}^* = (\lambda_1, \dots, \lambda_{N_d})^T$ *being the respective vector of Lagrange multipliers.*

*Proof.* This statement follows from the proof of Theorem 5.1.1. □

This means that $\forall i \in I$,

$$h_i(\mathbf{c}_k) \approx -\frac{\lambda_i^*}{\mu_k},$$

so the current iterate does not fulfill the constraints precisely, even though in the limit it holds $h_i(\mathbf{c}_k) \to 0$ as $k \to \infty$. To avoid this behavior, consider now a combination of the Lagrangian and the penalty function:

$$\mathcal{L}_A(\mathbf{c};\mu,\boldsymbol{\lambda}) := Q(\mathbf{c};\mu) - \sum_{i\in I} \lambda_i h_i(\mathbf{c}).$$

In the algorithmic framework the task is now to update $\mu_k$ and $\boldsymbol{\lambda}^k$ and find

$$\mathbf{c}_k = \underset{\mathbf{c}}{\operatorname{argmin}} \mathcal{L}_A(\mathbf{c}, \mu_k, \boldsymbol{\lambda}^k).$$

For this minimum, the optimality condition

$$0 \approx \nabla_{\mathbf{c}} \mathcal{L}_A(\mathbf{c}_k; \mu_k, \boldsymbol{\lambda}^k) = \nabla_{\mathbf{c}} G(\mathbf{c}) - \sum_{i \in I} \left(\lambda_i^k - \mu_k h_i(\mathbf{c}_k)\right) \nabla h_i(\mathbf{c}_k)$$

holds. Comparing this to the first KKT-condition (5.3) of problem (4.10), we get that

$$\lambda_i^* \approx \lambda_i^k - \mu_k h_i(\mathbf{c}_k)$$

for a vector of Lagrange multipliers $\boldsymbol{\lambda}^*$, or equivalently

$$h_i(\mathbf{c}_k) \approx -\frac{1}{\mu_k}(\lambda_i^* - \lambda_i^k).$$

Thus, in order to achieve feasibility, we can take $\lambda_i^k$ closer to $\lambda_i^*$ in every step by upgrading $\lambda_i^{k+1} = \lambda_i^k - \mu_k h_i(\mathbf{c}_k)$. Algorithm 3 depicts this scheme. The update pattern for $\mu_k$ and $\boldsymbol{\lambda}^k$ follows the approach from [56]. Note that $\mu_k$ is still updated, but it is not necessary to do this by a large factor and at every step. Most importantly, for convergence it is not needed to push $\mu_k$ to infinity as the following theorem suggests.

**Theorem 5.2.2.** [54, Thm. 17.5] *Let $\mathbf{c}^* \in \mathbb{R}^{2D}$ be a solution of (5.1) such that $\nabla h_i(\mathbf{c}^*)$ are linearly independent for all $i \in I$ and let $\boldsymbol{\lambda}^* \in \mathbb{R}^{N_d}$ be the corresponding vector of Lagrange multipliers. Let $(\mathbf{c}^*, \boldsymbol{\lambda}^*)$ fulfill the second-order optimality condition (see Remark below). Then, there exists a $\tilde{\mu} > 0$ such that*

$$\mathbf{c}^* = \underset{\mathbf{c}}{\operatorname{argmin}} \mathcal{L}_A(\mathbf{c}; \mu, \boldsymbol{\lambda}^*) \quad \forall \mu \geq \tilde{\mu}.$$

*Remark.* [54, Thm. 12.6] A tuple $(\mathbf{c}^*, \boldsymbol{\lambda}^*) \in \mathbb{R}^{2D} \times \mathbb{R}^{N_d}$ fulfilling the second-order optimality condition means that

$$\boldsymbol{\xi}^T \nabla^2_{\mathbf{cc}} \mathcal{L}(\mathbf{c}^*; \boldsymbol{\lambda}^*) \boldsymbol{\xi} > 0,$$

for all $\boldsymbol{\xi} \in \{\mathbf{x} \in \mathbb{R}^{2D} \mid \nabla h_i(\mathbf{c}^*)^T \mathbf{x} = 0 \quad \forall i \in I;\, \mathbf{x} \neq \mathbf{0}\}$.

*Proof.* For reasons of simplicity the proof of this theorem is not stated here, but can be found in [54]. □

**Algorithm 3** Augmented Lagrangian method [54, Framework 17.3], [56, Alg. 1]

Choose initial $\mathbf{c}_0 \in \mathbb{R}^{2D}, \boldsymbol{\lambda}^0 \in \mathbb{R}^{N_a}$, $\mu_0 > 0$, global tolerances $\varepsilon, \eta > 0$ and local tolerance parameters $\rho_0, \tau_0 > 0$, set $k = 0$.
**while** $\|\nabla_{\mathbf{c}} \mathcal{L}_A(\mathbf{c}_k; \mu_k, \boldsymbol{\lambda}^k)\| > \varepsilon$ **and** $\|\mathbf{h}(\mathbf{c}_k)\| > \eta$ **do**:
  $\mathbf{x}^* \leftarrow \operatorname{argmin}_{\mathbf{c}} \mathcal{L}_A(\mathbf{c}; \mu_k, \boldsymbol{\lambda}^k)$ by starting at $\mathbf{c}_k$ and minimizing until

$$\|\nabla_{\mathbf{c}} \mathcal{L}_A(\mathbf{x}^*; \mu_k, \boldsymbol{\lambda}^k)\| \leq \tau_k$$

  **if** $\|\mathbf{h}(\mathbf{x}^*)\| < \rho_k$ :
    $\boldsymbol{\lambda}^{k+1} \leftarrow \boldsymbol{\lambda}^k - \mu_k \mathbf{h}(\mathbf{x}^*)$
    $\mu_{k+1} \leftarrow \mu_k$
    $\tau_{k+1} \leftarrow \tau_k / \mu_{k+1}$
    $\rho_{k+1} \leftarrow \rho_k / \sqrt{\mu_{k+1}}$
  **else** :
    $\boldsymbol{\lambda}^{k+1} \leftarrow \boldsymbol{\lambda}^k$
    $\mu_{k+1} \leftarrow 5\mu_k$
    $\tau_{k+1} \leftarrow 1/\mu_{k+1}$
    $\rho_{k+1} \leftarrow 1/\sqrt{\mu_{k+1}}$
  $\mathbf{c}_{k+1} \leftarrow \mathbf{x}^*$
  $k \leftarrow k + 1$
**end while**

## 5.3 The Conjugate Gradient Method

As we have seen in Chapter 4, the finite element discretization results in minimization problems of the following form:

$$\min_{\mathbf{x}} \psi(\mathbf{x}) := \frac{1}{2} \mathbf{x}^T A \mathbf{x} - \mathbf{b}^T \mathbf{x} \tag{5.7}$$

with suitable (nonlinear) unit norm constraints. In the micromagnetic setting $\mathbf{x} \in \mathbb{R}^n$ represents the coefficient vector, $A \in \mathbb{R}^{n \times n}$ the stiffness matrix and $\mathbf{b}^T \in \mathbb{R}^n$ possible linear terms if we fix the magnetization or the vector potential. The joint minimization setting can be interpreted as a special case of this formulation with $\mathbf{b} = \mathbf{0}$. This optimization task is equivalent to solving a linear system, since its gradient satisfies

$$\nabla \psi(\mathbf{x}) = A\mathbf{x} - \mathbf{b}.$$

A common method to solve such tasks is the (nonlinear) *conjugate gradient method* (CG-method), which will be derived in the following sections. We start with the linear CG-method, but in order to include the nonlinear unit-norm constraint, we also implement a projected CG-scheme.

### 5.3.1 The Linear Conjugate Gradient Method

Before addressing the nonlinear case, the linear CG-method will be reviewed. The content of this section largely follows [54], [55] and [57].
For linear methods it is assumed that the matrix $A$ in (5.7) is not only symmetric but also positive definite (SPD). In this case, problem (5.7) becomes convex. The idea of using conjugate directions was first proposed by Hestenes and Stiefel in the 1950s [58] and is well-suited for large and/or sparse matrices since it comes with little storing effort and a fast convergence rate.
We start by defining what conjugate directions are.

**Definition 5.3.1.** [54]: A set of nonzero vectors $\{\mathbf{p}_0, \dots, \mathbf{p}_{k-1}\}$ is called conjugate with respect to the matrix $A$ if

$$\mathbf{p}_i^T A \mathbf{p}_j = 0 \quad \forall i, j \in \{0, \dots, k-1\},\ i \neq j. \tag{5.8}$$

*Remark.* From being conjugate with respect to $A$ it follows that $\{\mathbf{p}_0, \dots, \mathbf{p}_{k-1}\}$ is also a set of linearly independent vectors. Indeed, if we assume $\mathbf{0} = \sum_{i=0}^{k-1} \lambda_i \mathbf{p}_i$, it holds after multiplication with $\mathbf{p}_l^T A$, $0 \leq l \leq k-1$ that

$$0 = \mathbf{p}_l^T A \sum_{i=0}^{k-1} \lambda_i \mathbf{p}_i = \lambda_l \mathbf{p}_l^T A \mathbf{p}_l.$$

Since $A$ is SPD, we have $\lambda_l = 0$.

In the following we will call

$$\mathbf{r}(\mathbf{x}) \coloneqq \nabla \psi(\mathbf{x}) = A\mathbf{x} - \mathbf{b}$$

the residual at a point $\mathbf{x}$. Starting from an initial $\mathbf{x}_0$, the CG-method generates a sequence of iterates by

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \tag{5.9}$$

until a minimizer $\mathbf{x}^*$ is reached, fulfilling $r(\mathbf{x}^*) \approx 0$. Here, $\alpha_k$ denotes a step size which has to be determined during the optimization routine. If we choose the step size as

$$\alpha_k = \operatorname*{argmin}_{\alpha} \psi(\mathbf{x}_k + \alpha \mathbf{p}_k) \tag{5.10}$$

the following convergence result holds:

**Theorem 5.3.1.** [55, Thm. 6.3.2]: *Let $\psi : \mathbb{R}^n \to \mathbb{R}$ be defined as in (5.7) with $A \in \mathbb{R}^{n \times n}$ SPD and $\mathbf{b} \in \mathbb{R}^n$. Furthermore, let $\{\mathbf{p}_0, \dots, \mathbf{p}_{k-1}\}$ be a set of nonzero conjugate vectors with respect to $A$. Any sequence $\{\mathbf{x}_k\}_{k \in \mathbb{N}}$ generated by $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$ with $\alpha_k$ as in (5.10) converges to the solution $\mathbf{x}^*$ of*

$$\min_{\mathbf{x}} \psi(\mathbf{x})$$

*in at most n steps.*
*Furthermore, for* $\mathbf{r}_k := \mathbf{r}(\mathbf{x}_k)$ *the explicit form of the step size is*

$$\alpha_k = -\frac{\mathbf{r}_k^T \mathbf{p}_k}{\mathbf{p}_k^T A \mathbf{p}_k}.$$

*Proof.* We start by explicitly stating the step size. For $k = 0, \dots, n-1$ let

$$\Phi(\alpha) := \psi(\mathbf{x}_k + \alpha \mathbf{p}_k) = \frac{1}{2}\alpha^2 \mathbf{p}_k^T A \mathbf{p}_k + \alpha(A\mathbf{x}_k - \mathbf{b})^T \mathbf{p}_k + \frac{1}{2}\mathbf{x}_k^T A \mathbf{x}_k - \mathbf{b}^T \mathbf{x}_k$$

and

$$\Phi'(\alpha) = \alpha \mathbf{p}_k^T A \mathbf{p}_k + \mathbf{r}_k^T \mathbf{p}_k.$$

In order to satisfy the first order optimality condition $\Phi'(\alpha) = 0$, we get

$$\alpha = -\frac{\mathbf{r}_k^T \mathbf{p}_k}{\mathbf{p}_k^T A \mathbf{p}_k}.$$

For the proof of the convergence statement we observe that the vectors $\{\mathbf{p}_0, \dots, \mathbf{p}_{n-1}\}$ are linearly independent and therefore span the whole space $\mathbb{R}^n$. We can write the difference between the starting point of the iteration $\mathbf{x}_0$ and $\mathbf{x}^*$ as

$$\mathbf{x}^* - \mathbf{x}_0 = \sum_{i=0}^{n-1} \lambda_i \mathbf{p}_i$$

for some scalar coefficients $\lambda_i \in \mathbb{R}$. For all $k = 0, \dots, n-1$ we multiply this difference by $\mathbf{p}_k^T A$ and get by the conjugacy property

$$\lambda_k = \frac{\mathbf{p}_k^T A(\mathbf{x}^* - \mathbf{x}_0)}{\mathbf{p}_k^T A \mathbf{p}_k}.$$

We show that $\lambda_k = \alpha_k$ for all $k = 0, \dots, n-1$. Then the result follows.
Let $\mathbf{x}_k$ be generated by (5.9). Then

$$\mathbf{x}_k = \mathbf{x}_0 + \alpha_0 \mathbf{p}_0 + \cdots + \alpha_{k-1} \mathbf{p}_{k-1}.$$

We again multiply by $\mathbf{p}_k^T A$ and get

$$\mathbf{p}_k^T A(\mathbf{x}_k - \mathbf{x}_0) = 0$$

and therefore

$$\mathbf{p}_k^T A(\mathbf{x}^* - \mathbf{x}_0) = \mathbf{p}_k^T A(\mathbf{x}^* - \mathbf{x}_k) = \mathbf{p}_k^T(\mathbf{b} - A\mathbf{x}_k) = -\mathbf{p}_k^T \mathbf{r}_k.$$

This proves the claim. □

To generate the set of conjugate directions one can in general use the set of eigenvectors of $A$ or a modified Gram-Schmidt procedure to calculate them. However, there is a much more efficient method. For this, we need some preliminary results for the gradients which form the residual.

**Proposition 5.3.2.** [54, Thm. 5.2]: *Let the setting be the same as in Theorem 5.3.1. Then the sequence of residuals* $\{\mathbf{r}_k\}_{k\in\mathbb{N}}$ *fulfills*

*i)* $\mathbf{r}_{k+1} = \mathbf{r}_k + \alpha_k A\mathbf{p}_k$,

*ii)* $\mathbf{r}_k^T \mathbf{p}_i = 0 \quad \forall i = 0, \dots, k-1$.

*Proof.*

i) We have
$$\mathbf{r}_{k+1} = A\mathbf{x}_{k+1} - \mathbf{b} = A(\mathbf{x}_k + \alpha_k \mathbf{p}_k) - \mathbf{b} = \mathbf{r}_k + \alpha_k A\mathbf{p}_k.$$

ii) We prove the second statement by induction over $k$. Start with $k = 1$ and observe by $(i)$ and the definition of $\alpha_k$ that
$$\mathbf{r}_1^T \mathbf{p}_0 = \mathbf{r}_0^T \mathbf{p}_0 + \alpha_0 (A\mathbf{p}_0)^T \mathbf{p}_0 = 0.$$
Assume now $\mathbf{r}_{k-1}^T \mathbf{p}_i = 0$, $\forall i = 0, \dots, k-2$. Again, conclude
$$\mathbf{r}_k^T \mathbf{p}_{k-1} = \mathbf{r}_{k-1}^T \mathbf{p}_{k-1} + \alpha_{k-1} (A\mathbf{p}_{k-1})^T \mathbf{p}_{k-1} = 0.$$
For the other vectors $\mathbf{p}_i$, $i = 0, \dots, k-2$ we have
$$\mathbf{r}_k^T \mathbf{p}_i = \mathbf{r}_{k-1}^T \mathbf{p}_i + \alpha_{k-1} (A\mathbf{p}_{k-1})^T \mathbf{p}_i = 0,$$
since the first summand is zero by the induction hypothesis and the second by the conjugacy property.

□

This leads to a method to recursively generate $\mathbf{p}_0, \dots, \mathbf{p}_{n-1}$ while running the iterations. Here, the key aspect is that the current vector can be computed only with the previous iterate. The conjugacy to the other iterates will follow automatically.
Let $\mathbf{x}_0 \in \mathbb{R}^n$ be given. Choose $\mathbf{p}_0 := -\nabla\psi(\mathbf{x}_0) = -\mathbf{r}_0$. Assume that for $l \in \{0, \dots, n-2\}$ we have already constructed vectors $\mathbf{p}_0, \dots, \mathbf{p}_l \in \mathbb{R}^n \backslash \{0\}$ with
$$\mathbf{p}_i^T A\mathbf{p}_j = 0 \quad \forall i \neq j.$$
Because of Theorem 5.3.1 and Proposition 5.3.2 we have for $0 \leq k \leq l$
$$\alpha_k = -\frac{\mathbf{r}_k^T \mathbf{p}_k}{\mathbf{p}_k^T A\mathbf{p}_k} \quad \text{and} \quad \mathbf{r}_{k+1}^T \mathbf{p}_j = 0 \quad \text{for } j = 0, \dots, k.$$

Assume that $\mathbf{r}_{l+1} \neq 0$. In other words, $\mathbf{x}_{l+1}$ is not a solution of (5.7). Take the ansatz

$$\mathbf{p}_{l+1} \coloneqq -\mathbf{r}_{l+1} + \sum_{i=0}^{l} \beta_i^l \mathbf{p}_i. \tag{5.11}$$

We want the new iterate to fulfill the conjugacy with respect to $A$, so

$$\begin{aligned} &\mathbf{p}_{l+1}^T A \mathbf{p}_j = 0 \Leftrightarrow \\ &\mathbf{r}_{l+1}^T A \mathbf{p}_j = \beta_j^l \mathbf{p}_j^T A \mathbf{p}_j \Leftrightarrow \\ &\beta_j^l = \frac{\mathbf{r}_{l+1}^T A \mathbf{p}_j}{\mathbf{p}_j^T A \mathbf{p}_j} \quad j = 0, \dots, l. \end{aligned}$$

We can simplify this approach by using

$$\mathbf{r}_{l+1}^T \mathbf{r}_j = \mathbf{r}_{l+1}^T \left( -\mathbf{p}_j + \sum_{i=0}^{j-1} \beta_i^{j-1} \mathbf{p}_i \right) = 0 \quad j = 1, \dots, l$$

and

$$\mathbf{r}_{l+1}^T \mathbf{r}_0 = \mathbf{r}_{l+1}^T (-\mathbf{p}_0) = 0.$$

Therefore,

$$\mathbf{r}_{l+1}^T \mathbf{r}_j = 0 \quad j = 0, \dots, l.$$

This implies

$$\mathbf{r}_{j+1} - \mathbf{r}_j = A\mathbf{x}_{j+1} - A\mathbf{x}_j = \alpha_j A \mathbf{p}_j \Rightarrow A\mathbf{p}_j = \frac{1}{\alpha_j}(\mathbf{r}_{j+1} - \mathbf{r}_j).$$

Here we can assume $\alpha_j > 0$, since $-\mathbf{r}_j^T \mathbf{p}_j = \mathbf{r}_k^T \mathbf{r}_k > 0$ because of the choice of the ansatz (5.11).
Finally,

$$\mathbf{r}_{l+1}^T A \mathbf{p}_j = \frac{1}{\alpha_j}(\mathbf{r}_{l+1}^T \mathbf{r}_{j+1} - \mathbf{r}_{l+1}^T \mathbf{r}_j) = 0 \quad j = 0, \dots, l-1.$$

This means, $\beta_j^l = 0$ for all $j = 0, \dots, l-1$ and therefore

$$\mathbf{p}_{l+1} = -\mathbf{r}_{l+1} + \beta_{l+1} \mathbf{p}_l$$

with

$$\beta_l \coloneqq \frac{\mathbf{r}_{l+1}^T A \mathbf{p}_l}{\mathbf{p}_l^T A \mathbf{p}_l} = \frac{\mathbf{r}_{l+1}^T \mathbf{r}_{l+1}}{\mathbf{r}_l^T \mathbf{r}_l}.$$

The last equality follows again from Proposition 5.3.2. This update coefficient was originally proposed by Fletcher and Reeves in [59]. Using it in the iterative process, we obtain the linear CG-method which is depicted in Algorithm 4. Note that as a

**Algorithm 4** Linear CG-method [54, Alg. 5.2]

1: Choose initial $\mathbf{x}_0$; set $\mathbf{r}_0 = \nabla\psi(\mathbf{x}_0)$ and $\mathbf{p}_0 = -\mathbf{r}_0$, choose tolerance parameter $\varepsilon > 0$.
2: **for** $k = 0, 1, 2, \ldots$ **do**:
3: **if** $\|\mathbf{r}_k\|/\|\mathbf{r}_0\| < \varepsilon$ :
4: stop and return $\mathbf{x}_k$
5: $\alpha_k \leftarrow \frac{\mathbf{r}_k^T \mathbf{r}_k}{\mathbf{p}_k^T A \mathbf{p}_k}$
6: $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \alpha_k \mathbf{p}_k$
7: $\mathbf{r}_{k+1} \leftarrow \mathbf{r}_k + \alpha_k A \mathbf{p}_k$
8: $\beta_{k+1} \leftarrow \frac{\mathbf{r}_{k+1}^T \mathbf{r}_{k+1}}{\mathbf{r}_k^T \mathbf{r}_k}$
9: $\mathbf{p}_{k+1} \leftarrow -\mathbf{r}_{k+1} + \beta_{k+1} \mathbf{p}_k$
10: **end for**

convergence criterion we used the relative norm of the residual, i.e.

$$\frac{\|\mathbf{r}_k\|}{\|\mathbf{r}_0\|} < \varepsilon, \tag{5.12}$$

for a chosen parameter $\varepsilon > 0$. In the following we will briefly state a convergence property, showing that the structure of the matrix $A$ can be used to reach faster termination of the algorithm.
By defining the *energy norm* of A by

$$\|\mathbf{x}\|_A^2 := \mathbf{x}^T A \mathbf{x},$$

the condition number of $A$ can be linked to the convergence rate:

**Theorem 5.3.3.** [60, Theorem 10.2.6]: *Let $\varepsilon_k := \mathbf{x}^* - \mathbf{x}_k$ be the error in the k-th iteration of Algorithm 4. Then it holds*

$$\|\varepsilon_k\|_A \leq 2\left(\frac{\sqrt{\kappa_2(A)} - 1}{\sqrt{\kappa_2(A)} + 1}\right)^k \|\varepsilon_0\|_A.$$

*Proof.* A proof of this result can be found in [61]. □

Heuristically, this means that the CG-method converges fast, if $\kappa_2(A) \approx 1$. This motivates the use of preconditioners to improve the condition number of $A$.
The idea behind preconditioning is to apply the CG-method to a system

$$\tilde{A}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}$$

with $\tilde{A} = C^{-T}AC^{-1}, \tilde{\mathbf{x}} = C\mathbf{x}$ and $\tilde{\mathbf{b}} = C^{-T}\mathbf{b}$ where $C \in \mathbb{R}^{n\times n}$ is a non-singular SPD matrix chosen in a way that $\tilde{A}$ has a smaller condition number than $A$.
In practice one does not perform the full computation of $\tilde{A}$, but instead uses $P := C^TC$ as the so-called *preconditioner* which results in the preconditioned CG-method stated in Algorithm 5.
Different choices of preconditioners are tested in the applications in Chapter 6.

**Algorithm 5** Preconditioned linear CG-method [54, Alg. 5.3]

1: Choose a tolerance parameter $\varepsilon > 0$, an initial $\mathbf{x}_0$ and preconditioner $P$; set $\mathbf{r}_0 = \nabla\psi(\mathbf{x}_0)$;
solve $P\mathbf{y}_0 = \mathbf{r}_0$;
set the initial search direction $\mathbf{p}_0 = -\mathbf{y}_0$.
2: **for** $k = 0, 1, 2, \ldots$ **do**:
3: **if** (5.12) **holds:**
4: stop and return $\mathbf{x}_k$
5: $\alpha_k \leftarrow \frac{\mathbf{r}_k^T\mathbf{y}_k}{\mathbf{p}_k^TA\mathbf{p}_k}$
6: $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \alpha_k\mathbf{p}_k$
7: $\mathbf{r}_{k+1} \leftarrow \mathbf{r}_k + \alpha_kA\mathbf{p}_k$
8: solve $P\mathbf{y}_{k+1} = \mathbf{r}_{k+1}$
9: $\beta_{k+1} \leftarrow \frac{\mathbf{r}_{k+1}^T\mathbf{y}_{k+1}}{\mathbf{r}_k^T\mathbf{y}_k}$
10: $\mathbf{p}_{k+1} \leftarrow -\mathbf{y}_{k+1} + \beta_{k+1}\mathbf{p}_k$
11: **end for**

### 5.3.2 The Nonlinear Preconditioned Conjugate Gradient Method

We now aim to generalize the CG-algorithm to nonlinear problems of the form

$$\min_{\mathbf{x}} f(\mathbf{x}),$$

for $f : \mathbb{R}^n \to \mathbb{R}$ being a general nonlinear function. Originally, Fletcher and Reeves [59] proposed this extension by replacing the residual $\mathbf{r}_k$ with the gradient of $f$ at the point $\mathbf{x}_k$. The resulting (preconditioned) nonlinear CG-method is depicted in Algorithm 6. At this point, an important note has to be made about the construction of the step size $\alpha_k$. Until now an exact method to find the minimizer in line 3 of Algorithm 6 was used. However, in the nonlinear case such an approach often results in computationally costly optimization problems, so in practice one uses inexact line search methods instead. This introduces additional challenges. To illustrate this, consider the simplified update rule of the conjugate directions without preconditioning:

$$\mathbf{p}_{k+1} = -\mathbf{g}_{k+1} + \beta_{k+1}\mathbf{p}_{k+1}$$

**Algorithm 6** Preconditioned nonlinear CG-Method [12, Alg. 3]

Choose a tolerance parameter $\varepsilon > 0$, an initial $\mathbf{x}_0$ and preconditioner $P$; compute $\mathbf{g}_0 = \nabla f(\mathbf{x}_0)$;
solve $P\mathbf{y}_0 = \mathbf{g}_0$;
set the initial search direction $\mathbf{p}_0 = -\mathbf{y}_0$.
**for** $k = 0, 1, 2, \ldots$ **do**:
  **if** $\|\mathbf{g}_k\|/\|\mathbf{g}_0\| < \varepsilon$ :
    stop and return $\mathbf{x}_k$
  $\alpha_k \leftarrow \operatorname{argmin}_\alpha f(\mathbf{x}_{k-1} + \alpha \mathbf{p}_{k-1})$
  $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \alpha_k \mathbf{p}_k$
  $\mathbf{g}_{k+1} \leftarrow \nabla f(\mathbf{x}_{k+1})$
  solve $P\mathbf{y}_{k+1} = \mathbf{g}_{k+1}$
  compute $\beta_{k+1}$
  $\mathbf{p}_{k+1} \leftarrow -\mathbf{y}_{k+1} + \beta_{k+1}\mathbf{p}_k$
**end for**

and take the inner product with $\mathbf{g}_{k+1}$ to get

$$\mathbf{g}_{k+1}^T \mathbf{p}_{k+1} = -\|\mathbf{g}_{k+1}\|^2 + \beta_{k+1} \mathbf{g}_{k+1}^T \mathbf{p}_{k+1}.$$

If the line search for $\alpha_k$ is exact, we have $\mathbf{g}_{k+1}^T \mathbf{p}_{k+1} = 0$ and the newly calculated $\mathbf{p}_{k+1}$ is an actual descent direction. On the other hand, if the line search is not exact, it can happen that $\mathbf{g}_{k+1}^T \mathbf{p}_{k+1} > 0$ and $\mathbf{p}_{k+1}$ would be an ascent direction, thus breaking the convergence. Therefore, one has to impose additional conditions on $\alpha_k$. These will be addressed in section 5.3.3.
Moreover, in the nonlinear case, several methods to compute the parameter $\beta_k$ exist. The original proposal by Fletcher and Reeves uses a similar construction as in the linear case:

$$\beta_{k+1}^{FR} = \frac{\mathbf{g}_{k+1}^T \mathbf{y}_{k+1}}{\mathbf{g}_k^T \mathbf{y}_k}.$$

An improved version was presented by Polak and Ribiere [62] and Polyak [63], given by:

$$\beta_{k+1}^{PRP} = \frac{(\mathbf{g}_{k+1} - \mathbf{g}_k)^T \mathbf{y}_{k+1}}{\mathbf{g}_k^T \mathbf{y}_k}.$$

In the context of micromagnetics the construction by Hestenes and Stiefel [58] turns out to be the most efficient:

$$\beta_{k+1}^{HS} = \frac{(\mathbf{g}_{k+1} - \mathbf{g}_k)^T \mathbf{y}_{k+1}}{(\mathbf{g}_{k+1} - \mathbf{g}_k)^T \mathbf{p}_k}. \tag{5.13}$$

For this choice the updated search directions fulfill an additional conjugacy condition, namely

$$(\mathbf{g}_{k+1} - \mathbf{g}_k)^T \mathbf{p}_{k+1} = 0.$$

### 5.3.3 Line Search Methods

In this section a general concept for inexact line search methods to find the step length $\alpha_k$ is presented. The following results are based on [54].
As already discussed in Section 5.3.2 for inexact line search methods it is not necessarily guaranteed that the updated search direction $\mathbf{p}_k$ is a descent direction. To prevent this issue, we want to establish additional conditions on $\alpha_k$. Specifically, we require the new step size to fulfill two properties:

- The function value of $f$ should decrease significantly at the new point $\mathbf{x}_k$.
- The slope of $\alpha \mapsto f(\mathbf{x}_k + \alpha \mathbf{p}_k)$ should be reduced sufficiently.

These properties are formalized by the *Wolfe-conditions:*

**Definition 5.3.2.** [54]: Let $c_1 \in (0, 1/2)$ and $c_2 \in (c_1, 1/2)$. A step length $\alpha$ is said to satisfy the Wolfe-conditions if

i) $f(\mathbf{x}_k + \alpha \mathbf{p}_k) \leq f(\mathbf{x}_k) + c_1 \alpha \mathbf{g}_k^T \mathbf{p}_k$.

ii) $|\mathbf{g}_{k+1}^T \mathbf{p}_{k+1}| \geq c_2 |\mathbf{g}_k^T \mathbf{p}_k|$.

Condition *i)* is known as the *Armijo-condition* [64], while condition *ii)* is referred to as the *curvature condition.*

In order to generate a step size that approximately satisfies these conditions, usually backtracking algorithms are employed. A general variant is illustrated in Algorithm 7. Note that the algorithm only enforces the Armijo-condition as a stopping criterion.

**Algorithm 7** Backtracking line search [54, Alg. 3.1]

1: Choose $\alpha_0 > 0, \tau \in (0, 1), c_1 \in (0, 1/2)$.
Set $\alpha \leftarrow \alpha_0$.

2: **while** $f(\mathbf{x}_k + \alpha \mathbf{p}_k) > f(\mathbf{x}_k) + c_1 \alpha \mathbf{g}_k^T \mathbf{p}_k$ :

$$\alpha \leftarrow \tau \alpha$$

**end while.**

Nevertheless, this is an admissible strategy because the backtracking procedure ensures the sufficient decrease of the objective function [54].

### 5.3.4 The Projected Nonlinear CG-method

In order to solve the micromagnetic optimization problem (4.10) - (4.11) we adapt the nonlinear CG-method. The theory in this chapter follows the work of [12].
In the following, we project the coefficient vector $\mathbf{c}$ to the unit sphere in order to satisfy the optimality conditions. Firstly, we restate the definition of the Lagrangian associated with the problem:

$$\mathcal{L}(\mathbf{c};\boldsymbol{\lambda}) \coloneqq G(\mathbf{c}) - \boldsymbol{\lambda}^T \mathbf{h}(\mathbf{c}),$$

where $\boldsymbol{\lambda} \in \mathbb{R}^{N_d}$ denotes the vector of Lagrange multipliers and $\mathbf{h} : \mathbb{R}^{2D} \to \mathbb{R}^{N_d}$ describes the unit norm constraints as stated in (5.2). The Karush-Kuhn-Tucker (KKT) optimality conditions (5.3) - (5.4) require

$$\nabla_{\mathbf{c}} \mathcal{L}(\mathbf{c};\boldsymbol{\lambda}) = \mathbf{0},$$

and since

$$(\nabla_{\mathbf{c}} \mathcal{L}(\mathbf{c};\boldsymbol{\lambda}))_i = (\nabla G(\mathbf{c}))_i - \lambda_i \mathbf{c}_{\mathbf{m},i} \quad \forall i \in \mathcal{I}_\Omega, \tag{5.14}$$

it follows that

$$\lambda_i = \mathbf{c}_{\mathbf{m},i}^T (\nabla G(\mathbf{c}))_i \quad \forall i \in \mathcal{I}_\Omega.$$

Here, $\mathcal{I}_\Omega$ again denotes the node indices inside the magnetic domain. Using this in (5.14) leads to

$$\nabla_{\mathbf{c}} \mathcal{L}(\mathbf{c};\boldsymbol{\lambda}) = \Pi_{\mathbf{c}^\perp}(\nabla G(\mathbf{c}))$$

with the orthogonal projection defined at each node by

$$(\Pi_{\mathbf{c}^\perp} \mathbf{v})_i \coloneqq \begin{cases} \mathbf{v}_i - (\mathbf{v}_i^T \mathbf{c}_{\mathbf{m},i}) \mathbf{c}_{\mathbf{m},i} & \forall i \in \mathcal{I}_\Omega, \\ \mathbf{v}_i & \text{else.} \end{cases}$$

Therefore, the KKT conditions reduce to

$$\Pi_{\mathbf{c}^\perp}(\nabla G(\mathbf{c})) = 0. \tag{5.15}$$

However, instead of solving (5.15), we compute

$$\min_{\mathbf{c} \in \mathbb{R}^{2D}} \tilde{G}(\mathbf{c}) \coloneqq G(\mathcal{N}\mathbf{c})$$

with

$$(\mathcal{N}\mathbf{c})_i \coloneqq \begin{cases} \frac{\mathbf{c}_{\mathbf{m},i}}{\|\mathbf{c}_{\mathbf{m},i}\|} & \forall i \in \mathcal{I}_\Omega, \\ \mathbf{c}_{\mathbf{A},i} & \text{else.} \end{cases}$$

This is a justified approach, since

$$(\nabla \tilde{G}(\mathbf{c}))_i = \frac{1}{\|\mathbf{c}_{\mathbf{m},i}\|} \left( (\nabla G(\mathcal{N}\mathbf{c}))_i - \frac{\mathbf{c}_{\mathbf{m},i}^T (\nabla G(\mathcal{N}\mathbf{c}))_i}{\|\mathbf{c}_{\mathbf{m},i}\|^2} \mathbf{c}_{\mathbf{m},i} \right) \quad \forall i \in \mathcal{I}_\Omega$$

and thus minimizing $\tilde{G}$ is equivalent to solving (5.15). Using an adapted version of the nonlinear preconditioned CG-method from Algorithm 6, we obtain a program to solve

**Algorithm 8** Normalized preconditioned nonlinear CG-Method [12, Alg. 4]

Choose a tolerance parameter $\tau_F$, an initial $\mathbf{c}_0 \in \mathbb{R}^{2D}$ with $\|(\mathbf{c}_0)_{\mathbf{m},i}\|^2 = 1$ and preconditioner $P_0$; compute $\mathbf{g}_0 = \Pi_{\mathbf{c}_0^\perp}(\nabla G(\mathbf{c}_0))$;
solve $P_0\mathbf{y}_0 = \mathbf{g}_0$;
set the initial search direction $\mathbf{p}_0 = -\mathbf{y}_0$.
**for** $k = 0, 1, 2, \ldots$ **do**:
  **if** (5.16) **holds:**
    stop and return $\mathbf{c}_k$
  $\alpha_k \leftarrow \operatorname{argmin}_\alpha G(\mathcal{N}(\mathbf{c}_{k-1} + \alpha\mathbf{p}_{k-1}))$
  $\mathbf{c}_{k+1} \leftarrow \mathcal{N}(\mathbf{c}_k + \alpha_k\mathbf{p}_k)$
  $\mathbf{g}_{k+1} \leftarrow \Pi_{\mathbf{c}_{k+1}^\perp}(\nabla G(\mathbf{c}_{k+1}))$
  compute $P_{k+1}$
  solve $P_{k+1}\mathbf{y}_{k+1} = \mathbf{g}_{k+1}$
  compute $\beta_{k+1}$
  $\mathbf{p}_{k+1} \leftarrow -\mathbf{y}_{k+1} + \beta_{k+1}\mathbf{p}_k$
**end for**

the constraint problem. This is summarized in Algorithm 8. In order to determine the step size $\alpha_k$ we use the backtracking method depicted in Algorithm 7.
For the parameter $\beta_k$ we choose the variant of Hestenes-Stiefel (5.13) but we restart the computations under the following conditions on the gradient

$$\beta_{k+1} = \begin{cases} \frac{(\mathbf{g}_{k+1}-\mathbf{g}_k)^T\mathbf{y}_{k+1}}{(\mathbf{g}_{k+1}-\mathbf{g}_k)^T\mathbf{p}_k} & \text{if } \mathbf{y}_{k+1}^T\mathbf{g}_{k+1} > \mathbf{y}_{k+1}^T\mathbf{g}_k, \\ 0 & \text{else.} \end{cases}$$

This ensures that the update satisfies the Wolfe conditions and $\mathbf{p}_{k+1}$ is indeed a descent direction [65].
To implement a stopping criterion we use the one stated in [66]. Given a tolerance parameter $\tau_F = 10^{-a}$ with $a > 0$, the algorithm terminates if all of the following conditions are satisfied:

$$\begin{aligned} &i)\ G(\mathbf{c}_k) - G(\mathbf{c}_{k+1}) < \tau_F(1 + |G(\mathbf{c}_{k+1})|), \\ &ii)\ \|\mathbf{c}_k - \mathbf{c}_{k+1}\|_\infty < \sqrt{\tau_F}(1 + \|\mathbf{c}_{k+1}\|_\infty), \\ &iii)\ \|\mathbf{g}_{k+1}\|_\infty < \sqrt[3]{\tau_F}(1 + |G(\mathbf{c}_{k+1}|). \end{aligned} \tag{5.16}$$

# 6 Results and Discussion

In this chapter the discretization schemes and algorithms derived in the previous sections are tested and validated in various ways. Firstly, we perform a number of benchmark tests, where problems are solved for which an analytical solution is already known. Finally, a variation of the NIST $\mu$MAG standard problem #3 serves as a test for the joint minimization approach as well as a comparison to the classical alternating minimization schemes. All computations are done in Python and especially for the mesh generation and the finite element discretization the open-source library Netgen/NGsolve is used [67], [68]. Details of the mechanics of Netgen/NGsolve are provided in Appendix B.

## 6.1 Benchmark Tests

In order to check the accuracy of the minimization of the stray field energy, a series of benchmark tests has been implemented. There exists specific geometries where the solutions of magnetostatic equations can be expressed analytically. They serve as testing problems to compare the numerically calculated magnetizations and energies to the analytic solutions. The focus of this chapter lies on the numerical results and error measurements. However, the detailed computations of the involved scalar and vector potentials can be found in Appendix A.
All tests are performed in three dimensions and as the magnetic domain serves a sphere with center at the origin and radius $R > 0$,

$$\Omega = \{(x, y, z) \in \mathbb{R}^3 \mid x^2 + y^2 + z^2 < R^2\}.$$

For numerical treatment, the sphere is enclosed by a cube to model the full space integrals, as discussed in Section 4.1. As the accuracy of the results is also sensitive to the size of the outer region, the choice of the diameter of the enclosing cube requires careful consideration. In [69] it was proposed, that the distance between the outer boundary and the particle should be at least five times the size of the magnetic domain, so for all following computations we chose the outer cube to be six times as large as $\Omega$. Tests with larger outer domains only lead to an increase in the computational complexity but hardly influenced the accuracy of the solutions.
The meshing on both the inner and outer domain is performed by the internal mesher of Netgen (an extract of the code can be found in Appendix B). The mesh is refined on the magnetic domain and gradually gets coarser towards the outer boundary of the surrounding cube. This structure can be observed in Figure 6.1.

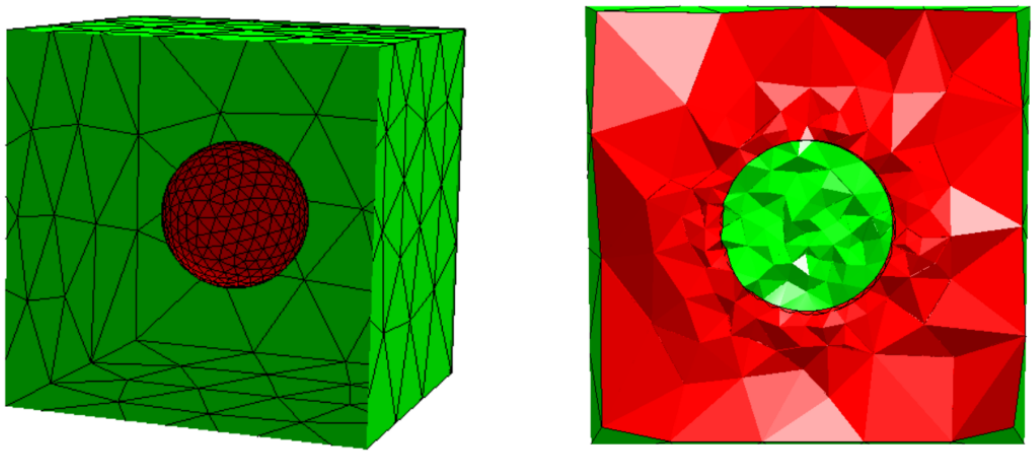

Figure 6.1: A spherical magnet embedded in an exterior cubic domain. Both domains are meshed with tetrahedrons, with a finer triangulation on the sphere. The second figure shows the gradual coarsening of the mesh size towards the outer boundary. For visualization purposes a larger mesh size has been chosen.

### 6.1.1 Fixed Magnetization in a Sphere

#### Example 1

We begin by calculating the vector potential from a fixed magnetization. As shown in [30] (for explicit computations see Appendix A), for a scaled magnetization

$$\mathbf{m}(\mathbf{x}) = \begin{cases} \frac{\mathbf{x}}{\|\mathbf{x}\|} & \text{in } \Omega \\ 0 & \text{else,} \end{cases} \tag{6.1}$$

the scalar potential for the sphere can be expressed as

$$\begin{aligned} \phi^{int}(\mathbf{x}) &= \|\mathbf{x}\| - R, \\ \phi^{ext}(\mathbf{x}) &= 0. \end{aligned}$$

Here, $\phi = (\phi^{int}, \phi^{ext}) \in H^1(\Omega) \times H^1_{loc}(\bar{\Omega}^c)$ is split into a part inside and outside of the magnetic domain respectively. The vector potential can also be computed analytically. From the calculations in Appendix A we get

$$\mathbf{A}^{int}(\mathbf{x}) = \mathbf{A}^{ext}(\mathbf{x}) = \mathbf{0}, \tag{6.2}$$

and thus,

$$\mathbf{b}^{int}(\mathbf{x}) = \mathbf{b}^{ext}(\mathbf{x}) = \mathbf{0}. \tag{6.3}$$

To measure the distance between this analytical solution and the numerically computed vector potential $\tilde{\mathbf{A}}$ and magnetic induction $\tilde{\mathbf{b}}$, we observe the $L^2(\mathcal{D};\mathbb{R}^3)$-norm, the $H^1(\mathcal{D};\mathbb{R}^3)$-semi-norm and the $H^1(\mathcal{D};\mathbb{R}^3)$-norm of the difference between algebraic and numerical field. Here, $\mathcal{D} \in \{\Omega, \bar{\Omega}^c\}$, so the errors are monitored separately inside and outside of the magnetic domain.
The corresponding value for the stray field energy can also be computed analytically. From

$$e_d = \int_\Omega \|\mathbf{m}\|^2 - \mathbf{m} \cdot \mathbf{b}\, d\mathbf{x}$$

it follows that

$$e_d = V_\Omega,$$

where $V_\Omega$ denotes the volume of the sphere.
The problem is solved for a radius of $R = 0.2$. Therefore, the analytical value of the stray field energy is $4\pi/375 \approx 0.0335\,[\frac{1}{2}\mu_0 M_s^2]$.
Firstly, the minimization problem was established according to the calculations from Section 4.2.1. The corresponding quadratic form in (4.12) was solved by the preconditioned CG-method from Algorithm 5. As an initial guess the coefficient vector of a perturbation of the analytical solution was used, i.e. the constant vector field

$$\mathbf{A}_0^{int}(\mathbf{x}) = \mathbf{A}_0^{ext}(\mathbf{x}) = (\varepsilon, \varepsilon, \varepsilon)^T,$$

where $\varepsilon$ is drawn uniformly from the interval $[-10^{-3}, 10^{-3}] \subset \mathbb{R}$.
In the course of this test problem, we consider the performance of two preconditioners for the CG-method. A simple diagonal (Jacobi) preconditioner of the stiffness matrix $F_A$ of the form

$$P = diag(F_A),$$

is compared with a multigrid preconditioner with a Gauss-Seidl smoother [70]. Table 6.1 shows a comparison of runtime and number of CG-iterations needed for different mesh sizes for these two preconditioners. It can be observed that even though the

| $\#elements$ | $t_J^p$ (s) | $t_m^p$ (s) | $t_J^{CG}$ (s) | $t_m^{CG}$ (s) | $it_J$ | $it_m$ |
|---|---|---|---|---|---|---|
| 2326 | 0.03 | 0.06 | 0.01 | 0.03 | 50 | 2 |
| 11964 | 0.11 | 0.21 | 0.02 | 0.03 | 84 | 2 |
| 65598 | 0.51 | 2.61 | 0.07 | 0.07 | 145 | 2 |
| 451714 | 4.87 | 154.41 | 1.57 | 1.62 | 250 | 2 |

Table 6.1: Performance of the CG method with Jacobi and multigrid preconditioners (Gauss–Seidel smoothing) for varying numbers of elements. Runtimes (in seconds) are split into setup time ($t_J^p, t_m^p$) and CG solve time ($t_J^{CG}, t_m^{CG}$). The corresponding iteration counts are denoted by $it_J$ and $it_m$.

multigrid approach needs less iterations even for larger mesh sizes, the Jacobi method is superior in terms of runtime as the problems get larger. This is due to the fast setup of

the preconditioned system for diagonal methods. Additionally, the CG-steps are more expensive in the multigrid setup, resulting into similar CG solving times as the diagonal method, even though less iterations are needed. Therefore, the Jacobi preconditioner is used in all following tests.
Table 6.2 shows the norm errors of the computed vector potential $\tilde{\mathbf{A}}$ and its corresponding magnetic induction $\tilde{\mathbf{b}}$ with their respective analytical solutions (6.2) and (6.3) inside the magnetic domain for different numbers of mesh elements, whereas Table 6.3 depicts the behavior in the enclosed cube. It can be observed that the method is already very accurate even for relatively large mesh sizes.

| $\#elements_\Omega$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{L^2(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{H^1(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{H^1(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{b}} - \mathbf{b}\|_{L^2(\Omega;\mathbb{R}^3)}$ |
|---|---|---|---|---|
| 644 | $7.8e-06$ | $8.9e-05$ | $9.0e-05$ | $1.1e-04$ |
| 1994 | $3.1e-06$ | $1.5e-04$ | $1.5e-04$ | $9.8e-05$ |
| 29863 | $4.3e-07$ | $4.3e-05$ | $4.3e-05$ | $2.9e-05$ |
| 244537 | $4.7e-08$ | $1.1e-05$ | $1.1e-05$ | $6.7e-06$ |

Table 6.2: Norm difference of analytical and numerical vector potential and magnetic induction for a fixed scaled magnetization $\mathbf{m}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|$ inside a sphere with radius $R = 0.2$.

| $\#elements_{\bar{\Omega}^c}$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{L^2(\bar{\Omega}^c;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{H^1(\bar{\Omega}^c;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{H^1(\bar{\Omega}^c;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{b}} - \mathbf{b}\|_{L^2(\bar{\Omega}^c;\mathbb{R}^3)}$ |
|---|---|---|---|---|
| 1682 | $1.2e-05$ | $1.4e-04$ | $1.4e-04$ | $9.0e-05$ |
| 9970 | $1.2e-06$ | $1.0e-05$ | $1.0e-05$ | $7.3e-06$ |
| 35735 | $1.1e-07$ | $8.9e-07$ | $8.9e-07$ | $6.2e-07$ |
| 207177 | $8.3e-09$ | $6.3e-08$ | $6.3e-08$ | $4.3e-08$ |

Table 6.3: Norm difference of analytical and numerical vector potential and magnetic induction for a fixed scaled magnetization $\mathbf{m}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|$ in the cube surrounding the sphere.

The computed energy is $\tilde{e_d} \approx 0.0335\,[\frac{1}{2}\mu_0 M_s^2]$ for all observed mesh sizes and matches the analytical value even for the coarsest mesh, consisting of 2326 total elements, with an absolute error of approximately $10^{-8}$.

### Example 2

As a second example we consider a uniform magnetization

$$\mathbf{m}(\mathbf{x}) = \begin{cases} (0,0,1)^T & \text{in } \Omega, \\ 0 & \text{else.} \end{cases} \tag{6.4}$$

Again, $\Omega$ is a sphere with radius $R = 0.2$. In this case, the exact scalar potential is given by [30]

$$\phi^{int}(\mathbf{x}) = \frac{x_3}{3},$$
$$\phi^{ext}(\mathbf{x}) = R^3 \frac{x_3}{3\|\mathbf{x}\|^3},$$

whereas the analytic vector potential is described by

$$\mathbf{A}^{int}(\mathbf{x}) = \frac{1}{3}(-x_2, x_1, 0)^T,$$
$$\mathbf{A}^{ext}(\mathbf{x}) = \frac{R^3}{3\|\mathbf{x}\|^3}(-x_2, x_1, 0)^T.$$

From this the magnetic induction can be deduced as

$$\mathbf{b}^{int}(\mathbf{x}) = \nabla \times \mathbf{A}^{int}(\mathbf{x}) = (0, 0, 2/3)^T,$$
$$\mathbf{b}^{ext}(\mathbf{x}) = \nabla \times \mathbf{A}^{ext}(\mathbf{x}) = \frac{R^3}{\|\mathbf{x}\|^5}\big(x_1x_3, x_2x_3, (-x_1^2 - x_2^2 + 2x_3^2)/3\big)^T.$$

Again, see Appendix A for the computation of the scalar and vector potential.
Using this for calculating the stray field energy, we get an exact energy value of $1/3V_\Omega \approx 0.0117[\frac{1}{2}\mu_0 M_s^2]$. Again, problem (4.12) is solved by the preconditioned CG-method and as a starting vector the coefficients of a constant perturbation of the analytical solution are used, i.e.

$$\mathbf{A}_0^{int}(\mathbf{x}) = \mathbf{A}^{int}(\mathbf{x} + \varepsilon\mathbf{1})$$
$$\mathbf{A}_0^{ext}(\mathbf{x}) = \mathbf{A}^{ext}(\mathbf{x} + \varepsilon\mathbf{1}),$$

with $\varepsilon \in [-10^{-3}, 10^{-3}]$ drawn uniformly. Table 6.4 and Table 6.5 depict the norm differences inside and outside the sphere.

| $\#elements_{\bar{\Omega}^c}$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{L^2(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{H^1(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{H^1(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{b}} - \mathbf{b}\|_{L^2(\Omega;\mathbb{R}^3)}$ |
|---|---|---|---|---|
| 644 | $1.4e-03$ | $1.6e-02$ | $1.7e-02$ | $4.9e-02$ |
| 1994 | $5.2e-04$ | $6.8e-03$ | $6.8e-03$ | $3.2e-02$ |
| 29863 | $2.5e-04$ | $3.2e-03$ | $3.2e-03$ | $2.2e-02$ |
| 244537 | $1.4e-04$ | $1.7e-03$ | $1.7e-03$ | $1.6e-02$ |

Table 6.4: Norm difference of analytical and numerical vector potential and magnetic induction for a fixed uniform magnetization $\mathbf{m}(\mathbf{x}) = (0, 0, 1)^T$ inside a sphere with radius $R = 0.2$.

In Table 6.6 the evolution of the calculated stray field energy is shown for an increasing number of total mesh elements. Figure 6.2 shows the numerical vector potential in comparison with its analytical expression. In Figure 6.3 the absolute error is portrayed.

| $\#elements_{\bar{\Omega}^c}$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{L^2(\bar{\Omega}^c;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{H^1(\bar{\Omega}^c;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{A}} - \mathbf{A}\|_{H^1(\bar{\Omega}^c;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{b}} - \mathbf{b}\|_{L^2(\bar{\Omega}^c;\mathbb{R}^3)}$ |
|---|---|---|---|---|
| 1682 | $5.3e-03$ | $5.6e-02$ | $5.6e-02$ | $5.2e-02$ |
| 9970 | $3.6e-03$ | $3.8e-02$ | $3.8e-02$ | $3.3e-02$ |
| 35735 | $2.5e-03$ | $2.7e-02$ | $2.7e-02$ | $2.2e-02$ |
| 207177 | $1.8e-03$ | $2.0e-02$ | $2.0e-02$ | $1.6e-02$ |

Table 6.5: Norm difference of analytical and numerical vector potential and magnetic induction for a fixed uniform magnetization $\mathbf{m}(\mathbf{x}) = (0,0,1)^T$ in the cube surrounding the sphere.

| $\#elements$ | $\tilde{e}_d[\frac{1}{2}\mu_0 M_s^2]$ |
|---|---|
| 2326 | 0.0162 |
| 11964 | 0.0133 |
| 65598 | 0.0122 |
| 451714 | 0.0117 |

Table 6.6: Calculated stray field energy for a fixed uniform magnetization $\mathbf{m}(\mathbf{x}) = (0,0,1)^T$ in a sphere with radius $R = 0.2$. The reference energy is $e_d = 0.0117[\frac{1}{2}\mu_0 M_s^2]$.

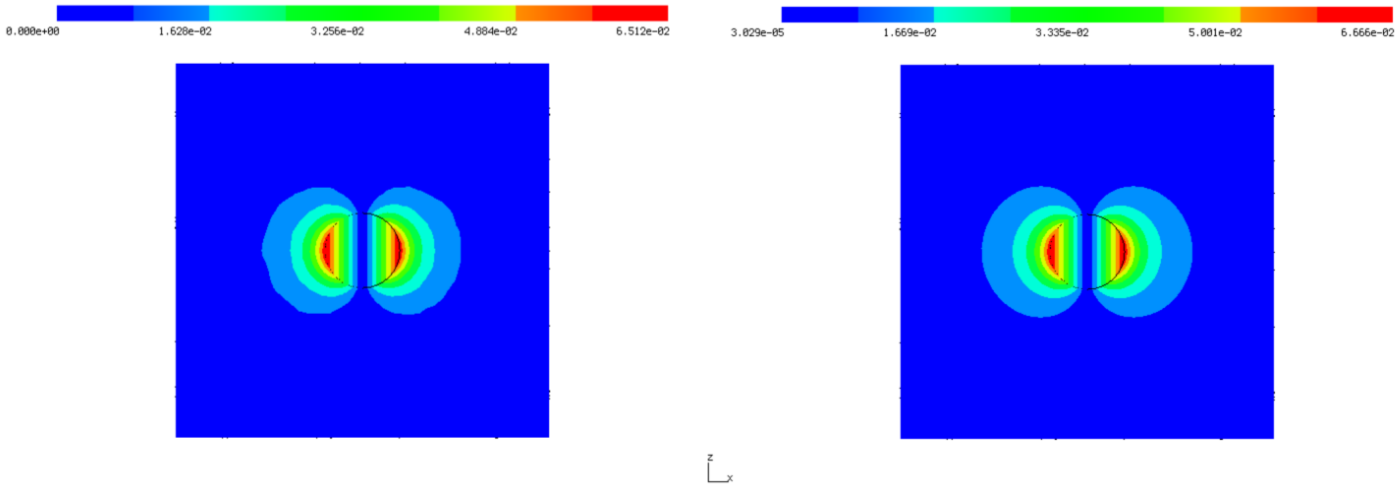


Figure 6.2: Comparison of the numerically computed vector potential $\tilde{\mathbf{A}}$ (left) and the analytic expression $\mathbf{A}$ (right) for a uniform magnetization in a sphere with radius $R = 0.2$ embedded in a cube.

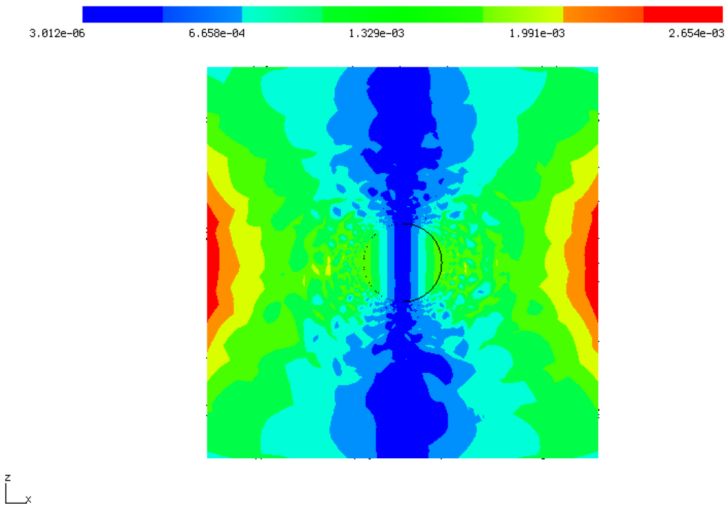


Figure 6.3: Absolute error of the analytical and the numerically computed vector potential for a uniform magnetization in a sphere with radius $R = 0.2$ embedded in a cube.

### 6.1.2 Fixed Vector Potential in a Sphere

Next, we want to calculate the minimal upper bound for the stray field energy and therefore an optimal magnetization for a fixed vector potential. This means, we aim to solve the problem

$$\min_{\mathbf{m}} e_{\mathbf{A}}(\mathbf{m}, \mathbf{A}) \quad s.t. \|\mathbf{m}\| = 1 \quad \text{in } \Omega, \tag{6.5}$$

with $e_{\mathbf{A}}$ being the upper bound for the stray field energy described in (2.19). To this end, we reuse the setting of a sphere with a fixed radius of $R = 0.2$. The construction of the corresponding stiffness matrix follows from the computations in Section 4.2.2. The resulting system is of the form:

$$\frac{1}{2}\mathbf{c_m} F_d \mathbf{c_m} - \mathbf{g}^T \mathbf{c_m},$$

subject to the discretized unit norm constraints (4.14). Here, $\mathbf{c_m}$ is the coefficient vector of $\mathbf{m}$ and $F_d$ the stiffness matrix for the upper bound, i.e. the globalized version of (4.7). Different methods are tested to satisfy the unit norm constraint, namely the quadratic penalty method (Algorithm 2), the augmented Lagrangian method (Algorithm 3) and the projected CG-method (Algorithm 8).

#### Example 1

To begin with, we consider again the case of a scaled magnetization on the sphere (6.1), but this time we fix the associated vector potential (6.2). As a starting configuration we

choose a small perturbation of $\mathbf{m}$, i.e the constant vector field

$$\mathbf{m}_0(\mathbf{x}) = \begin{cases} \frac{(x_1+\varepsilon, x_2+\varepsilon, x_3+\varepsilon)^T}{\|\mathbf{x}+\varepsilon\mathbf{1}\|} & \text{in } \Omega, \\ 0 & \text{else}, \end{cases} \tag{6.6}$$

with $\varepsilon$ drawn uniformly from the interval $[-10^{-3}, 10^{-3}]$.
Similar to the last section, we measure the $L^2$-norm, the $H^1$-semi-norm and the $H^1$-norm of the absolute differences between the numerical and the analytical magnetization separately inside and outside the sphere. The norm differences for the three methods as well as the computed energies and the maximal violation of the norm constraint are depicted in Tables 6.7, 6.8 and 6.9.

| method | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{L^2(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\Omega;\mathbb{R}^3)}$ |
|---|---|---|---|
| qpm | $1.9e-03$ | 2.2113 | 2.2113 |
| aLm | $9.6e-04$ | 2.2108 | 2.2108 |
| pCG | $1.7e-03$ | 2.2113 | 2.2113 |

Table 6.7: Norm difference of analytical and numerical magnetization for the fixed vector potential associated with $\mathbf{m}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|$ inside a sphere with radius $R = 0.2$. The methods were tested on a mesh with 244537 elements inside the sphere.

| method | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{L^2(\bar{\Omega}^c;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\bar{\Omega}^c;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\bar{\Omega}^c;\mathbb{R}^3)}$ |
|---|---|---|---|
| qpm | 0.0433 | 7.2024 | 7.2025 |
| aLm | 0.0433 | 7.2024 | 7.2025 |
| pCG | 0.0433 | 7.2024 | 7.2025 |

Table 6.8: Norm difference of analytical and numerical magnetization for the fixed vector potential associated with $\mathbf{m}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|$ outside the sphere with 207177 mesh elements in the outside region.

| Method | $\tilde{e}_d$ | $err_\infty$ |
|---|---|---|
| qpm | 0.0334 | $8.9e-07$ |
| aLm | 0.0334 | $2.6e-10$ |
| pCG | 0.0334 | $2.2e-16$ |

Table 6.9: Calculated stray field energy values and maximal constraint violations for the methods tested for a fixed vector potential associated with a magnetization $\mathbf{m}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|$. The analytical stray field energy is $e_d = 0.0335\,[\frac{1}{2}\mu_0 M_s^2]$.

One observes that the difference in the norms is large compared to the previous results. Taking a closer look at the $L^2$-norm of the absolute error in Figure 6.4 it can be seen that the inconsistencies lay mainly at the boundary of the sphere. Figure 6.5 depicts the Ngsolve implementation of the Euclidean norm of the initial magnetization $\|\mathbf{m}_0\|$.

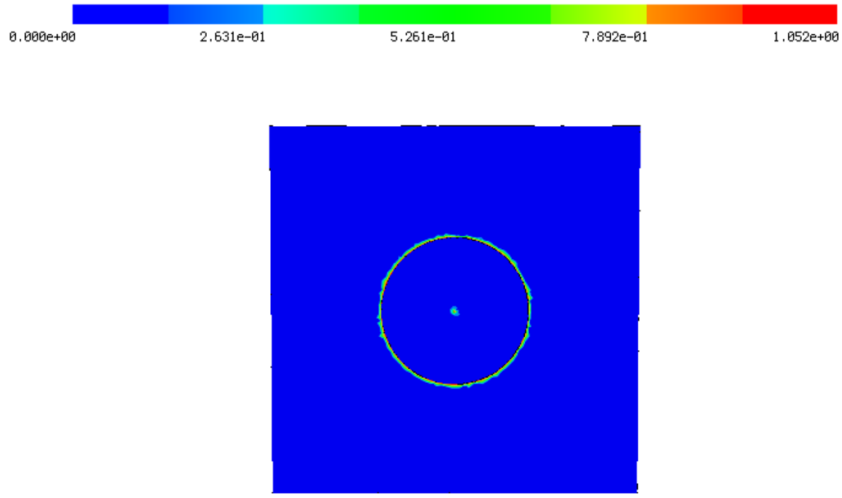


Figure 6.4: Absolute difference of the scaled magnetization (6.1) and the computed solution of a nonlinear CG method.

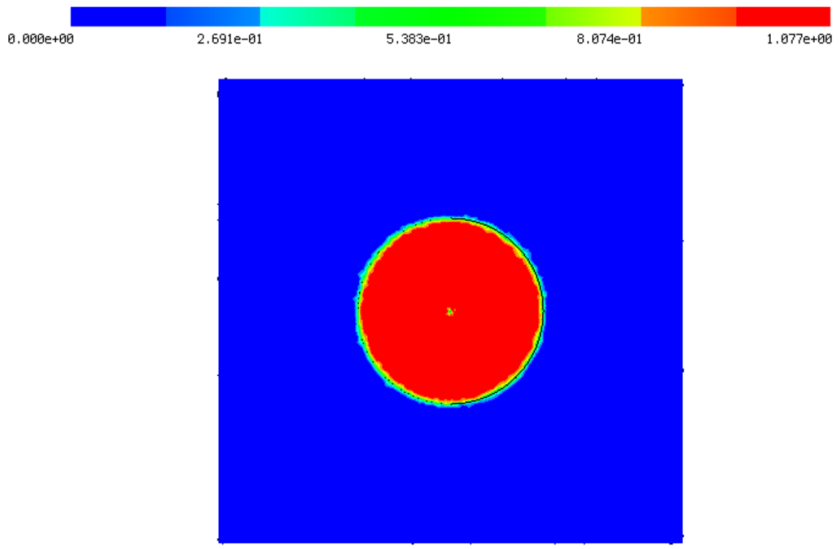


Figure 6.5: Euclidean norm of the initial configuration (6.6). The norm deviates at the boundary of the sphere from the expected values.

It shows, apart from a singularity at the origin, in the vicinity of the boundary of the sphere, the values already deviate from 1 and 0 in the respective domains. This happens because the finite element space $\mathring{\mathbf{V}}_h^1 \subset (H^1(\Omega \cup \bar{\Omega}^c))^3$ in (4.1) is globally defined over both the inner and the outer domain and therefore enforces continuity of the involved functions. As a result, the program tries to smoothen the magnetization and ignores the jump at the boundary of $\Omega$, resulting in the measured errors. One suggested fix of this problem could be the use of two separate but coupled finite element spaces for the inner and outer domain. This approach is discussed in the following Section 6.1.3.

### Example 2

For the sake of completeness, the calculations for a fixed vector potential have also been carried out in the case of a uniform magnetization (6.4) on a global finite element space. Again, the constant vector field of a random perturbation of the solution acts as a starting configuration:

$$\mathbf{m}_0(\mathbf{x}) = \begin{cases} (\varepsilon, \varepsilon, 1+\varepsilon)^T & \text{in } \Omega \\ 0 & \text{else,} \end{cases} \tag{6.7}$$

with $\varepsilon \in [-10^{-3}, 10^{-3}]$. Tables 6.10 and 6.11 depict the resulting norm errors, whereas Table 6.12 shows the calculated energies for the different methods. Again, the inaccuracies originate from ignoring the jump at the boundary of the sphere.

| method | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{L^2(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\Omega;\mathbb{R}^3)}$ |
|---|---|---|---|
| qpm | $2.4e-04$ | $1.3e-07$ | $2.4e-04$ |
| aLm | $1.2e-05$ | $7.6e-09$ | $1.2e-05$ |
| pCG | $2.3e-04$ | $2.6e-15$ | $2.3e-04$ |

Table 6.10: Norm difference of analytical and numerical magnetization for the fixed vector potential associated with $\mathbf{m}(\mathbf{x}) = (0,0,1)^T$ inside a sphere with radius $R = 0.2$. The methods were tested on a mesh with 244537 elements inside the sphere.

| method | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{L^2(\bar{\Omega}^c;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\bar{\Omega}^c;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\bar{\Omega}^c;\mathbb{R}^3)}$ |
|---|---|---|---|
| qpm | 0.0433 | 7.1920 | 7.1922 |
| aLm | 0.0433 | 7.1920 | 7.1922 |
| pCG | 0.0433 | 7.1920 | 7.1922 |

Table 6.11: Norm difference of analytical and numerical magnetization for the fixed vector potential associated with $\mathbf{m}(\mathbf{x}) = (0,0,1)^T$ outside the sphere with 207177 mesh elements in the outside region.

| Method | $\tilde{e}_d$ | $err_\infty$ |
|---|---|---|
| qpm | 0.0116 | $3.2e-09$ |
| aLm | 0.0116 | $2.0e-10$ |
| pCG | 0.0116 | $1.7e-16$ |

Table 6.12: Calculated stray field energy values and maximal constraint violations for the methods tested for a fixed vector potential associated with a uniform magnetization $\mathbf{m}(\mathbf{x}) = (0,0,1)^T$. The analytical stray field energy is $e_d = 0.0117\,[\frac{1}{2}\mu_0 M_s^2]$.

### 6.1.3 Separation of Finite Element Spaces

As suggested in Section 6.1.2 a possible root for errors at the boundary of domains is the use of a globally defined finite element space $\mathring{\mathbf{V}}_h^1 \subset (H^1(\Omega \cup \bar{\Omega}^c))^3$. Therefore, for the following computations, we introduce two separate function spaces

$$\mathring{\mathbf{V}}_h^1(\Omega) \coloneqq \{\mathbf{v}_h \in C(\Omega;\mathbb{R}^3) \mid \mathbf{v}_h|_T \in (\mathcal{P}_1(T)) \quad \forall T \in \mathcal{T}_h\} \subseteq (H^1(\Omega))^3$$

and

$$\begin{aligned}\mathring{\mathbf{V}}_h^1(\bar{\Omega}^c) \coloneqq \{&\mathbf{v}_h \in C(\bar{\Omega}^c;\mathbb{R}^3) \mid \mathbf{v}_h|_T \in (\mathcal{P}_1(T)), \forall T \in \mathcal{T}_h; \quad v_{h,i}|_{\Gamma_D} = 0, \forall i = 1,2,3\} \\ &\subseteq (H^1(\bar{\Omega}^c))^3\end{aligned}$$

for a given triangulation $\mathcal{T}_h$ with mesh size $h > 0$. Then, for the computations of the finite element matrices and load vectors the compound space

$$\mathring{\mathbf{V}}_h^1 \coloneqq \mathring{\mathbf{V}}_h^1(\Omega) \times \mathring{\mathbf{V}}_h^1(\bar{\Omega}^c) \tag{6.8}$$

is used.
Again, the schemes are tested for a scaled and a uniform magnetization in a sphere.

#### Example 1

As in the previous section we start with the scaled magnetization (6.1) in a sphere with radius $R = 0.2$. Table 6.13 shows the norm differences inside the sphere. They have been reduced drastically due to the splitting of the FE-spaces. Outside the sphere the methods are now able to keep $\mathbf{m}(\mathbf{x}) = 0, \quad \forall \mathbf{x} \in \bar{\Omega}^c$. As a consequence, the norm errors outside the magnetic domain are exactly 0. Moreover, in Figure 6.6 it can be noticed that the errors inside the sphere result solely from the singularity at the origin and the inaccuracies at the boundary have been eliminated completely.

| method | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{L^2(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\Omega;\mathbb{R}^3)}$ |
|---|---|---|---|
| qpm | $8.3e-04$ | 0.0835 | 0.0835 |
| aLm | $8.0e-04$ | 0.0871 | 0.0871 |
| pCG | $7.0e-04$ | 0.0780 | 0.0780 |
| ini | $4.8e-03$ | 0.0358 | 0.0358 |

Table 6.13: Norm difference of analytical and numerical magnetization for the fixed vector potential associated with $\mathbf{m}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|$ inside a sphere with radius $R = 0.2$. A compound FE-space consisting of two separated function spaces for inside and outside the magnetic domain was used. Additionally, the initial error of the difference $\mathbf{m}_0 - \mathbf{m}$ is shown for reference.

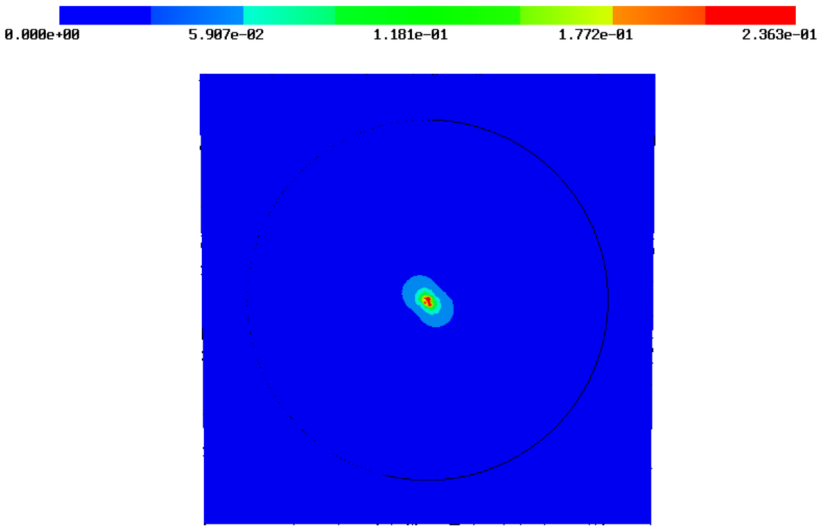


Figure 6.6: Absolute difference of the analytical and numerical magnetization for a fixed vector potential associated to the scaled magnetization $\mathbf{m}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|$ in the case of two separated FE-spaces. The inaccuracies only occur at the singularity at the origin.

**Example 2**

The splitting approach has also been tested for a uniform magnetization (6.4). Again, Table 6.14 depicts the results inside the sphere for the different methods, whereas outside the magnetic domain the errors are exactly zero as before.

| method | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{L^2(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\Omega;\mathbb{R}^3)}$ | $\|\tilde{\mathbf{m}} - \mathbf{m}\|_{H^1(\Omega;\mathbb{R}^3)}$ |
|---|---|---|---|
| qpm | $1.1e-03$ | $2.1e-09$ | $1.1e-03$ |
| aLm | $1.5e-04$ | $1.1e-08$ | $1.5e-04$ |
| pCG | $4.0e-05$ | $6.7e-15$ | $4.0e-05$ |
| ini | $3.2e-03$ | $1.5e-14$ | $3.2e-03$ |

Table 6.14: Norm difference of analytical and numerical magnetization for the fixed vector potential associated with $\mathbf{m}(\mathbf{x}) = (0,0,1)^T$ inside a sphere with radius $R = 0.2$ for an approach with two separated FE-spaces. Additionally, the initial error of the difference $\mathbf{m}_0 - \mathbf{m}$ is shown for reference.

Both examples show that splitting the finite element domains increases the accuracy of the stray field computations immensely. Compared to the initial error the $L^2$-norm of the difference to the calculated magnetization improved by a factor of 10 for the scaled and by a factor of $10^2$ for the uniform magnetization, at least in the case of the CG-method. This is sufficient for our purposes.

### 6.1.4 Discretization Accuracy of Exchange and Anisotropy Energy

After successfully solving analytical benchmark problems for the stray field energy it is now necessary to verify the accuracy of the discretizations of the remaining energy terms. We will not solve isolated minimization problems but use magnetization configurations where the analytical solution of the corresponding energies is known and compare them with the results of the linear systems established in Sections 4.1.1 and 4.1.2. Since we eventually want to work in the setting of the NIST $\mu$MAG Standard Problem #3, the magnetic domain $\Omega$ is now the unit cube enveloped in a larger outer cube $K$ modeling the whole space $\mathbb{R}^3$. We implement a non-uniform mesh on both $\Omega$ and $K$. It is refined on the magnetic domain and gradually gets coarser towards $\Gamma_D$, the outer boundary of $K$. Additionally, the mesh density is increased near the corners of $\Omega$ to overcome the continuity issues at the sharp edges. The resulting domain and mesh structure can be observed in Figure 6.7.
Again, the finite element spaces are separated for the inner and the outer domain. The following tests were established in [36].

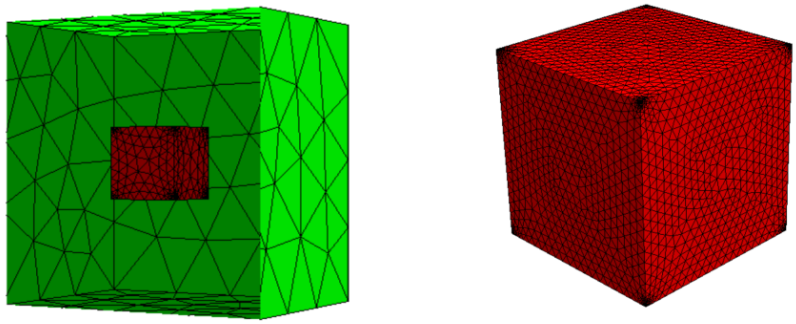

Figure 6.7: The magnetic domain $\Omega$ as a unit cube (red) embedded in the surrounding outer body $K$. For better visibility the tetrahedrons in the interior region of $K$ have been removed. The right panel shows the non-uniform mesh on the facets of $\Omega$.

### Anisotropy energy

To check the accuracy of the linear system for the anisotropy energy a homogeneous magnetization tilted along the easy axis is chosen [36], i.e.

$$\mathbf{m} = (sin\,\theta, 0, cos\,\theta)^T \quad \text{in } \Omega,$$

and $\mathbf{m} = \mathbf{0}$ outside for a fixed angle $\theta \in (0, \pi/2)$. Therefore, the analytical energy is

$$e_a(\mathbf{m}) = Q \int_\Omega 1 - (\mathbf{m} \cdot \mathbf{a})^2 \, d\mathbf{x} = Q \int_\Omega (sin\,\theta)^2 \, d\mathbf{x} = Q(sin\,\theta)^2 Vol(\Omega) = Q(sin\,\theta)^2,$$

since the easy axis is chosen as $\mathbf{a} = (0, 0, 1)^T$.
Deriving the stiffness matrix $F_a$ from Section 4.1.2 and solving

$$\tilde{e}_a(\mathbf{m}) = F_a \mathbf{m},$$

we can compare these energies for a uniformly chosen tilt $\theta \in (0, \pi/2)$. The obtained absolute error is

$$|\tilde{e}_a - e_a| \approx 10^{-15},$$

so around machine precision for all tested tilts and different mesh sizes ranging from a number of $3 \times 10^4$ to $10^6$ elements. This coincides with the observations in [36].

### Exchange Energy

For the exchange energy the analysis has to be more precise and the number of elements of the mesh indeed is an issue. The proposed magnetization for testing the discretization in

[36] is an inhomogeneous 180° wall, so the magnetization changes sign inside the magnet, i.e.

$$\mathbf{m}(\mathbf{x}) = (0, sin(2\pi x_1/L), cos(2\pi x_1/L))^T \quad \text{in } \Omega,$$

and constant zero outside. Here, $L$ denotes the reduced length of the cube. As a consequence, the reduced exchange energy is

$$\begin{aligned} e_{ex}(\mathbf{m}) &= \tilde{A}_{ex} \int_\Omega \|\nabla \mathbf{m}\|_F^2 \, d\mathbf{x} \\ &= \tilde{A}_{ex} \int_\Omega \frac{4\pi^2}{L^2} \|(0, cos(2x_1\pi/L), sin(2x_1\pi/L)^T\|^2 \, d\mathbf{x} \\ &= 4\tilde{A}_{ex} \frac{\pi^2}{L^2} Vol(\Omega). \end{aligned}$$

For a cube with length $L = 1$ we get

$$e_{ex} = 4\pi^2.$$

Again we calculate the energy with the linear system derived in 4.1.1 by

$$\tilde{e}_{ex} = F_{ex}\mathbf{m}.$$

We tested the discretization for different mesh sizes and computed the ratio between the analytical and numerical energy. The results can be seen in Figure 6.8. The outcome gets better, the more elements occur in the mesh, especially inside the magnet. Again, a high precision in the discretization can be achieved, but only for suitable mesh sizes.

## 6.2 Total Energy Minimization

In contrast to the isolated tests in the previous sections, this chapter deals with the minimization of the total energy in the setting of the NIST $\mu$MAG standard problem #3 [35]. As introduced in Section 6.1.4 the magnetic domain $\Omega$ is the unit cube enclosed in an outer cube $K$.
As a starting configuration of the magnetization, we use the flower state $\mathbf{m}_f$ and the vortex state $\mathbf{m}_v$ depicted in Chapter 2.8. For the parameters of the flower state we always chose $a = c = 1$ and $b = 2$. Tests with other configurations yielded similar results as the chosen values.
The aim of this test is to compute the minimum of the total energy (2.25) for both initial states in the single domain limit (SDL), the length of the cube, where flower and vortex state result in the same energy configuration. In [23] a SDL of $L = 8.4729\,[l_{ex}]$ was computed. This value is used for the calculation of the dimensionless exchange parameter $\tilde{A}_{ex}$, as shown in Section 2.8. Again, a split in the finite element spaces for inner and outer domains has been performed.
Note that all energy terms in the reference [23] are in units of $[\frac{1}{2}\mu_0 M_s^2]$, so an additional factor 2 has to be added to the reduced energy formulations in equation (2.25).

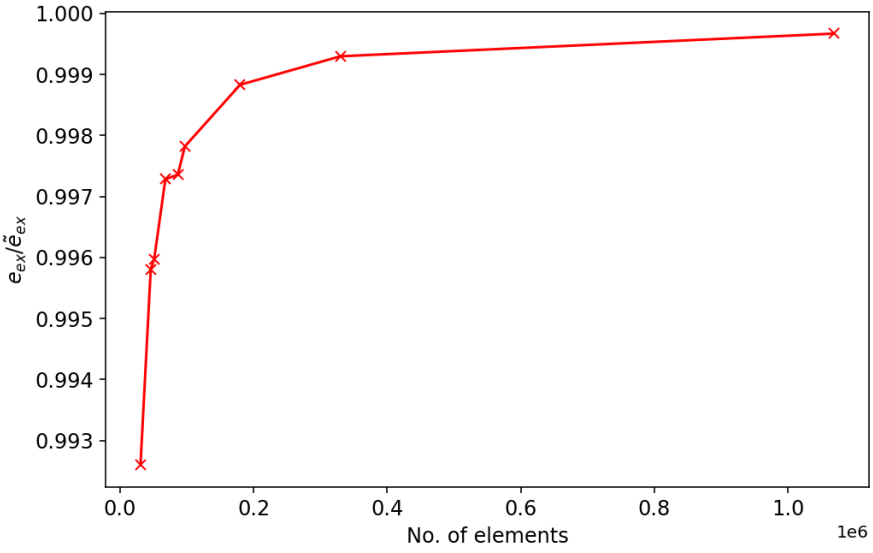


Figure 6.8: Ratio between analytical and numerical value of the exchange energy for a rotating magnetization inside a unit cube as a function of the number of elements of the mesh.

### 6.2.1 Alternating Minimization

Before testing simultaneous minimization, we implement alternating optimization methods as described in Algorithm 1. Consequently, we solve the minimization task for a fixed magnetization (4.12) and the corresponding problem for a fixed vector potential (4.13) - (4.14) alternately.
Again, we are comparing quadratic penalty- (Algorithm 2), augmented Lagrangian- (Algorithm 3) and projected CG-methods (Algorithm 8). Tables 6.15 and 6.16 show the results compared to the reference solution, whereas Figure 6.9 illustrates the evolution of the energy terms over the iterations of Algorithm 1.

### 6.2.2 Joint Minimization

Finally, the total energy is minimized simultaneously with respect to the magnetization and the vector potential. Therefore the system matrix is set up as seen in Section 4.1 and problem (4.10) - (4.11) has to be solved. Again, the domain is split in an inner and outer region and finite element spaces are built accordingly. As in the previous sections, the results of quadratic penalty, augmented Lagrangian and projected CG-methods are compared with the reference solution.
Once more, the algorithms are initialized by the coefficient vectors of the normalized flower and vortex states depicted in (2.23) - (2.24). This was done as follows:
Let $\mathbf{c}_{\mathbf{m},0}$ be this initial coefficient vector. In order to obtain a starting configuration for

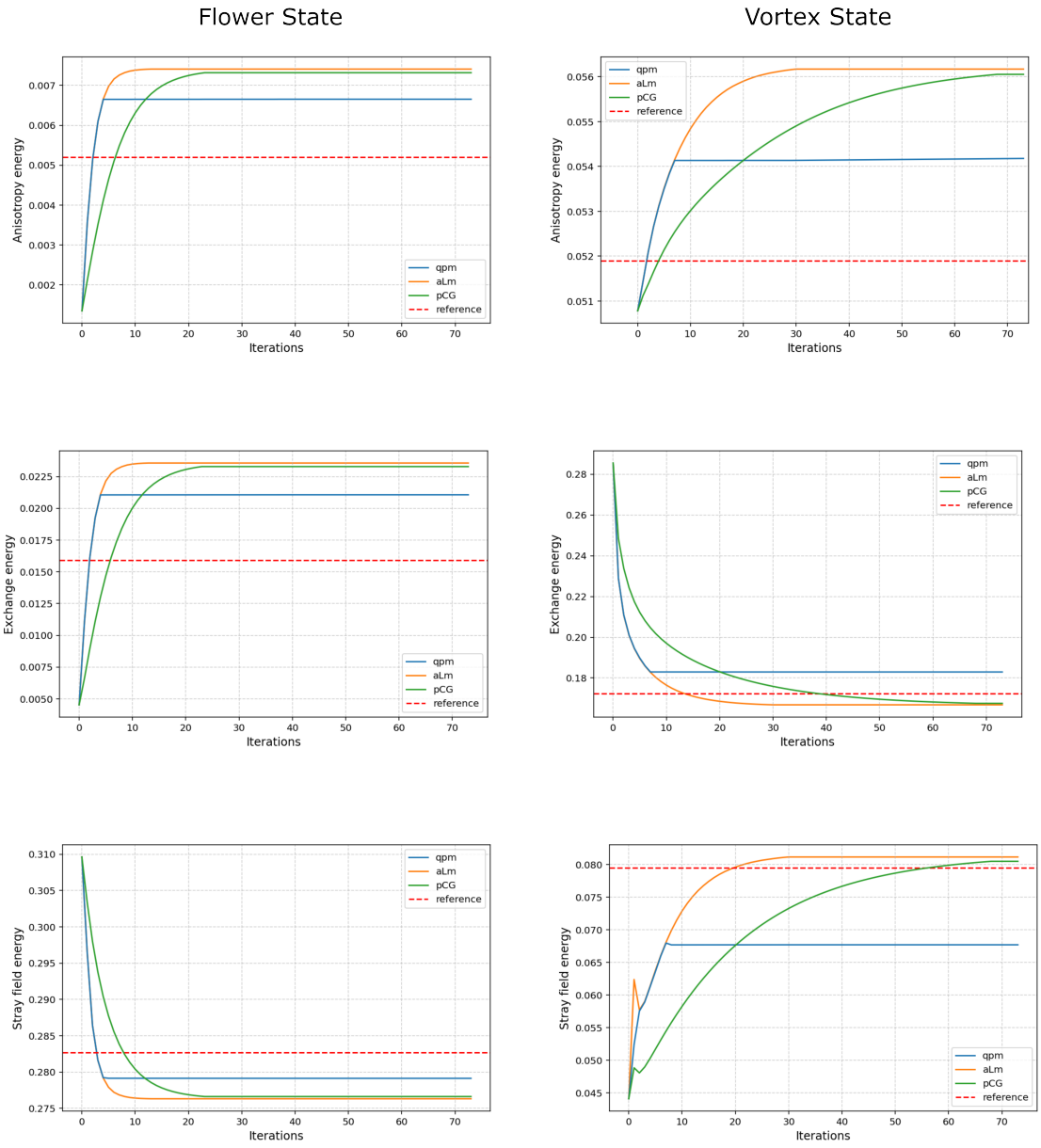


Figure 6.9: Comparison of the evolution of the individual energy terms for different minimization methods. The dotted, red line describes the corresponding reference solution from [23].

| | qpm | aLm | pCG | ref |
|---|---|---|---|---|
| $e_a$ | 0.0067 | 0.0074 | 0.0073 | 0.0058 |
| $e_{ex}$ | 0.0211 | 0.0236 | 0.0233 | 0.0177 |
| $e_d$ | 0.2790 | 0.2763 | 0.2766 | 0.2804 |
| $< m_1 >$ | 0.0017 | 0.0023 | 0.0018 | −0.0008 |
| $< m_2 >$ | −0.0002 | −0.0003 | −0.0003 | −0.0034 |
| $< m_3 >$ | 0.9579 | 0.9529 | 0.9536 | 0.9700 |
| $e$ | 0.3069 | 0.3073 | 0.3072 | 0.3038 |

Table 6.15: Results of the alternating minimization of the total energy with an initial flower state. The individual energy terms in units of $[\frac{1}{2}\mu_0 M_s^2]$ as well as the mean magnetizations are compared to the reference solution from [23].

| | qpm | aLm | pCG | ref |
|---|---|---|---|---|
| $e_a$ | 0.0544 | 0.0562 | 0.0561 | 0.0520 |
| $e_{ex}$ | 0.1804 | 0.1668 | 0.1676 | 0.1738 |
| $e_d$ | 0.0697 | 0.0812 | 0.0805 | 0.0781 |
| $< m_1 >$ | 0.0179 | 0.0155 | 0.0165 | −0.0005 |
| $< m_2 >$ | 0.2761 | 0.3520 | 0.3480 | 0.3464 |
| $< m_3 >$ | −0.0147 | −0.0008 | −0.0045 | −0.0068 |
| $e$ | 0.3044 | 0.3042 | 0.3041 | 0.3038 |

Table 6.16: Results of the alternating minimization of the total energy with an initial vortex state. The individual energy terms in units of $[\frac{1}{2}\mu_0 M_s^2]$ as well as the mean magnetizations are compared to the reference solution from [23].

the vector potential we solve the minimization problem for a fixed magnetization once, i.e.

$$\mathbf{c}_{\mathbf{A},0} = \underset{\mathbf{c}\in\mathbb{R}^D}{\operatorname{argmin}} \frac{1}{2}\mathbf{c}^T F_A \mathbf{c} - \mathbf{f}^T \mathbf{c},$$

with the system matrix $F_A \in \mathbb{R}^{D\times D}$ and the load vector $\mathbf{f} \in \mathbb{R}^D$ established in Section 4.2.1. This is attempted to be solved by the linear CG-method (Algorithm 4). Note that it is not necessary to fully run the algorithm until it terminates at an exact solution, but already truncate the iterative process after a few steps. The full starting vector then reads as

$$\mathbf{c}_0 = (\mathbf{c}_{\mathbf{m},0}, \mathbf{c}_{\mathbf{A},0})^T \in \mathbb{R}^{2D}.$$

For this setup, the initial energy configurations are given in Table 6.17.

Another remark has to be made on the projected CG-method. Until now, it was sufficient to use the diagonal preconditioner

$$P = diag(A)$$

| | $e_a^{(0)}$ | $e_{ex}^{(0)}$ | $e_d^{(0)}$ | $e^{(0)}$ |
|---|---|---|---|---|
| flower | 0.0013 | 0.0045 | 0.3096 | 0.6251 |
| vortex | 0.0508 | 0.2855 | 0.0441 | 0.4245 |

Table 6.17: Initial energy configurations for the starting values used in the joint minimization approach for a magnetization in a flower and vortex state. All energy terms are in units of $[1/2\mu_0 M_s^2]$.

for the respective system matrix $A$. This no longer works in the joint setting, since the system matrix is of the form

$$A = \tilde{H} + \tilde{F} - 2\tilde{K} \in \mathbb{R}^{2D \times 2D}$$

(see (4.10)) and taking only the diagonal neglects too much of the structure of the individual matrix terms. Therefore, we found that an incomplete LU factorization yields a more satisfactory outcome [70].
The results of the computations can be found in Tables 6.18 and 6.19.

| | qpm | aLm | pCG | ref |
|---|---|---|---|---|
| $e_a$ | 0.0018 | 0.0029 | 0.0068 | 0.0058 |
| $e_{ex}$ | 0.0054 | 0.0088 | 0.0209 | 0.0177 |
| $e_d$ | 0.3062 | 0.2983 | 0.2774 | 0.2804 |
| $< m_1 >$ | 0.0006 | 0.0007 | −0.0001 | −0.0008 |
| $< m_2 >$ | −0.0001 | −0.0001 | −0.0010 | −0.0034 |
| $< m_3 >$ | 0.9892 | 0.9829 | 0.9618 | 0.9700 |
| $e$ | 0.3134 | 0.3100 | 0.3051 | 0.3038 |

Table 6.18: Results of the joint minimization of the total energy with an initial flower state. The individual energy terms in units of $[\frac{1}{2}\mu_0 M_s^2]$ as well as the mean magnetizations are compared to the reference solution from [23].

## 6.3 Discussion

In this section we analyze and compare the results of alternating and the simultaneous optimization procedure. We focus on the accuracy of the methods with respect to the reference solution as well as on stability depending on the initial configuration and on an interpretation from a mathematical point of view.

From the comparison of Tables 6.15 - 6.16 and Tables 6.18 - 6.19 it can be observed that both approaches are capable of approximating the reference solution, but show differences in robustness and consistency. In the alternating scheme all methods yield total energies which are close to the reference value for both initial flower and vortex

| | qpm | aLm | pCG | ref |
|---|---|---|---|---|
| $e_a$ | 0.0511 | 0.0516 | 0.0556 | 0.0520 |
| $e_{ex}$ | 0.2545 | 0.2270 | 0.1739 | 0.1738 |
| $e_d$ | 0.0560 | 0.0587 | 0.0810 | 0.0781 |
| $< m_1 >$ | 0.0229 | 0.0224 | 0.0123 | −0.0005 |
| $< m_2 >$ | 0.0429 | 0.0848 | 0.3172 | 0.3464 |
| $< m_3 >$ | −0.0192 | −0.0185 | −0.0159 | −0.0068 |
| $e$ | 0.3616 | 0.3374 | 0.3105 | 0.3038 |

Table 6.19: Results of the joint minimization of the total energy with an initial vortex state. The individual energy terms in units of $[\frac{1}{2}\mu_0 M_s^2]$ as well as the mean magnetizations are compared to the reference solution from [23].

state. This is in accordance with the definition of the single domain limit, where both states should converge to the same energy configuration.
On the other hand, the joint minimization yields comparable results only for an initial flower state and only with the projected CG-method as an iterative solver. For an initial vortex state, the deviations from the reference value are larger, but still the pCG-method delivers the best results compared to the penalty-type schemes.
Overall, treating the total energy as the single object of comparison, the alternating method is more robust and less sensitive to the initialization, whereas the joint approach demands a more careful choice of solver to achieve comparable results.

### Influence of the Optimization Method

In the alternating framework all three methods yield similar results. This indicates that the decoupled subproblems are sufficiently well conditioned and can be handled efficiently.
In contrast, the joint minimization shows strong dependence on the choice of the solver. Quadratic penalty methods perform the worst, indicating that ill conditioning effects for large penalty parameters, as discussed in Section 5.1, occur. Indeed, for simultaneous optimization we obtain parameters $\mu \approx \mathcal{O}(10^9)$, whereas $\mu$ never exceeds $10^5$ in the alternating scheme. In alignment with the theory, the augmented Lagrangian method produces more stable and balanced outcomes. Notably, the projected CG-method shows the best results in both optimization schemes and even in the joint setting it is an appropriate choice provided a suitable preconditioner.

### Computational Aspects

From a computational perspective, the alternating scheme benefits from splitting the problem into smaller and simpler subproblems, whereas the joint program possesses

more complexity due to the form and size of its system matrix

$$A = \tilde{H} + \tilde{F} - 2\tilde{K}.$$

Especially the pattern in $\tilde{K}$ introduces a coupling between the involved variables and imposes a certain density. In Figure 6.10 the difference in complexity is depicted by comparing the number of nonzero elements in the respective system matrices for different numbers of degrees of freedom. However, the simultaneous minimization approach has

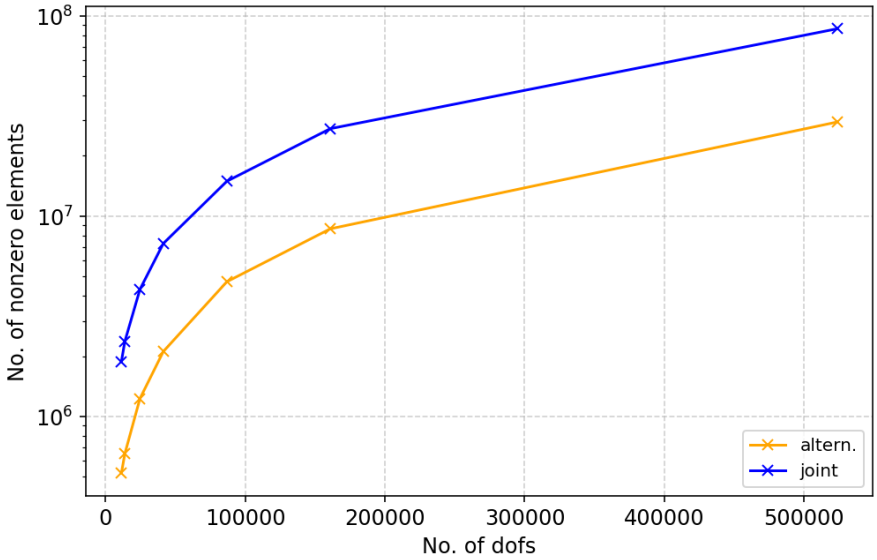


Figure 6.10: Comparison of the nonzero elements in the system matrices for alternating and joint minimization schemes for different numbers of degrees of freedom.

also advantages in terms of computational scaling. Comparing only the projected CG-methods for both schemes yields a noticeable decrease in the number of CG-iterations needed and thus also in the runtime. Table 6.20 shows the results of this comparison for a minimization with an initial flower state and fixed global mesh size $h$ and tolerance parameter $\tau_F$ for the stopping criterion in (5.16). To be comparable, both methods use the same preconditioner, namely the incomplete LU factorization.
It can be observed that setting up the preconditioner for the joint method takes significantly longer than in the alternating scheme due to the larger size of the system matrices. However, the number of CG-iterations can be reduced drastically by doing so.

To summarize, the alternating method describes a robust and reliable optimization scheme, which delivers accurate results independent of the initial configuration and optimization method. The joint approach is more challenging numerically, but also capable of achieving comparable accuracy when combined with a suitable solver. In particular, the projected CG-method significantly outperforms the penalty and the augmented Lagrange method. Additionally, the reduced computational complexity is a promising sign

| | total runtime | time precond. | No. CG-iter. |
|---|---|---|---|
| alt. | 140 s | 0.01 s | 1660 |
| joint | 101 s | 45 s | 546 |

Table 6.20: Comparison of the alternating and the joint minimization principle in terms of computational complexity. Both methods used an initial flower state and the same meshing of the domain with 86934 total nodes. As a stopping parameter $\tau_F = 10^{-6}$ is used in the convergence criterion (5.16). For both methods an incomplete LU factorization functions as a preconditioner.

for future development. Therefore, while the alternating program remains the method of choice, the joint approach represents a viable and potentially more powerful alternative, although further optimization will be needed in future work.

# 7 Conclusion

In this work one of the most prominent static energy equations, the total Gibbs free energy, was minimized with respect to the magnetization obeying a nonlinear unit norm constraint inside the magnetic domain. In particular, the magnetostatic self-energy was approximated by an upper bound proposed by Brown, eliminating the non-local nature of the former. As a result, the upper bound also depends on a vector potential, making the objective a function in two three-dimensional variables. Moreover, the vector potential is defined on the whole space, which posed an additional challenge in the discretization.

This results in the computational domain being the magnet of spherical or cubic shape embedded in an outer cube of relatively large size, modeling the whole space. This domain was partitioned in a mesh of tetrahedrons and for the discretization of the energy terms a finite element ansatz was employed. More specifically, a P1-finite element subspace was constructed to model the magnetization and the vector potential. This transforms the analytic energy equation into a sparse algebraic system.

In a next step, different methods for solving these systems and handling the nonlinear unit norm constraint were introduced, namely the quadratic penalty method, the augmented Lagrangian method and the projected CG-method. The main idea is to simultaneously minimize the energy function in both the vector potential and the magnetization. Additionally, an alternating optimization method was implemented as well.

First benchmark test with solutions which can be computed analytically showed that the discretization lacked sufficient accuracy. It was observed that the inaccuracies appeared mainly at the boundary of the magnetic domain, suggesting that one global finite element space results in interpolation phenomena in this area. Therefore, an important improvement was the splitting of the domain into an inner and an outer finite element space, reducing the computational errors drastically.

Finally, the total energy was minimized and the joint optimization scheme was compared with the alternating methods. To this end, a modification of the NIST $\mu$MAG standard problem #3 was observed by statically computing the energy states at a given single domain limit. The calculations showed that although the alternating approach is still the more robust and viable method, the joint minimization procedure shows promising signs to be a serious contender especially in terms of computational complexity. Therefore, this approach represents a potential direction for additional research and merits further investigation in the future.

# Appendix A

## Analytical computations

In this section the calculations of the scalar and the vector potential are displayed in settings where an analytical solution is computable. Two variants of a magnetization in three dimensions will be observed, whereas the domain of interest will stay the same for both, $\Omega \subset \mathbb{R}^3$ is a sphere with radius $R > 0$, i.e.

$$\Omega = \{(x, y, z) \in \mathbb{R}^3 \mid x^2 + y^2 + z^2 < R^2\}.$$

### Example 1

We start with the example of a scaled magnetization

$$\mathbf{m}(\mathbf{x}) = \begin{cases} \frac{\mathbf{x}}{\|\mathbf{x}\|} & \text{in } \Omega \\ 0 & \text{else.} \end{cases} \tag{7.1}$$

Firstly, we aim to compute the scalar potential $\phi : \mathbb{R}^3 \to \mathbb{R}$ for this given magnetization. Using the magnetostatic equations for the stray field, as well as the splitting Ansatz of García-Cervera and Roma, yields the following transmission problem [30], [71]: Split $\phi = \phi_1 + \phi_2$ with $\phi_1 = (\phi_1^{int}, \phi_1^{ext}) \in H^1(\Omega) \times H^1_{loc}(\bar{\Omega}^c)$ fulfilling

$$\begin{aligned} -\Delta\phi_1^{int} &= -\nabla \cdot \mathbf{m} && \text{in } \Omega, \\ \phi_1^{int} &= 0 && \text{on } \partial\Omega, \end{aligned} \tag{7.2}$$

and $\phi_1^{ext} = 0$ in $\bar{\Omega}^c$. For $\phi_2 = (\phi_2^{int}, \phi_2^{ext}) \in H^1(\Omega) \times H^1_{loc}(\bar{\Omega}^c)$ it holds

$$\begin{aligned} -\Delta\phi_2^{int} &= 0 && \text{in } \Omega, \\ -\Delta\phi_2^{ext} &= 0 && \text{in } \bar{\Omega}^c, \\ [\phi_2] &= 0 && \text{on } \partial\Omega, \\ \left[\frac{\partial\phi_2}{\partial\mathbf{n}}\right] &= -\mathbf{m}\cdot\mathbf{n} + \frac{\partial\phi_1^{int}}{\partial\mathbf{n}} && \text{on } \partial\Omega, \\ \phi_2^{ext} &= \mathcal{O}\left(\frac{1}{\|\mathbf{x}\|}\right) && \text{as } \|\mathbf{x}\| \to \infty. \end{aligned} \tag{7.3}$$

Here, the jump conditions $[\cdot]$ describe the differences in the behaviors at the boundary of $\Omega$ by

$$[\phi_2] \coloneqq (\phi_2^{ext} - \phi_2^{int})\big|_{\partial\Omega}$$
$$\left[\frac{\partial\phi_2}{\partial\mathbf{n}}\right] \coloneqq \left(\frac{\partial\phi_2^{ext}}{\partial\mathbf{n}} - \frac{\partial\phi_2^{int}}{\partial\mathbf{n}}\right)\Big|_{\partial\Omega}.$$

For the specific choice of the magnetization it holds,

$$\nabla\cdot\mathbf{m} = \frac{2}{\|\mathbf{x}\|} \quad \text{in } \Omega.$$

So for $\phi_1$ one needs to solve

$$\begin{aligned} -\Delta\phi_1^{int} &= -\frac{2}{\|\mathbf{x}\|} && \text{in } \Omega, \\ \phi_1^{int} &= 0 && \text{on } \partial\Omega, \\ \phi_1^{ext} &= 0 && \text{in } \bar{\Omega}^c. \end{aligned}$$

Write $r = \|\mathbf{x}\|$ and assume that $\phi_1^{int}$ can be described by a radial function:

$$\phi_1^{int}(\mathbf{x}) = f(r).$$

Then, in three dimensions the Laplacian for radial functions is

$$\Delta f(r) = \frac{2}{r}f'(r) + f''(r).$$

Therefore, we solve

$$-\left(f'' + \frac{2}{r}f'\right) = -\frac{2}{r}.$$

This is equivalent to

$$-(r^2 f')' = -2r,$$

and consequently

$$r^2 f' = r^2 + C$$

for a constant $C \in \mathbb{R}$. However, regularity at $r = 0$ forces $C = 0$, hence

$$f(r) = r + D,$$

for $D \in \mathbb{R}$ and together with the boundary condition $f(R) = 0$ we get

$$\phi_1^{int}(\mathbf{x}) = \|\mathbf{x}\| - R. \tag{7.4}$$

On the other hand, the outward normal vector on a sphere with radius $R$ is described by

$$\mathbf{n}(\mathbf{x}) = \frac{\mathbf{x}}{R}$$

and therefore on $\partial\Omega$ it holds

$$\frac{\partial \phi_1^{int}}{\partial \mathbf{n}} = \nabla \phi_1^{int} \cdot \mathbf{n} = 1.$$

This yields

$$\left[\frac{\partial \phi_2}{\partial \mathbf{n}}\right] = -\mathbf{m} \cdot \mathbf{n} + 1 = -\frac{\|\mathbf{x}\|}{R} + 1 = 0.$$

Together with the other requirements in (7.3) we get

$$\phi_2^{int}(\mathbf{x}) = \phi_2^{ext}(\mathbf{x}) = 0. \tag{7.5}$$

Combining (7.4) and (7.5) leaves us with the final analytic solution

$$\begin{aligned} \phi_{int}(\mathbf{x}) &= \|\mathbf{x}\| - R, \\ \phi_{ext}(\mathbf{x}) &= 0. \end{aligned}$$

A similar computation can be done for the vector potential

$$\mathbf{A} = \mathbf{A}_1 + \mathbf{A}_2$$

with

$$\begin{aligned} \Delta \mathbf{A}_1^{int} &= -\nabla \times \mathbf{m} && \text{in } \Omega, \\ \mathbf{A}_1^{int} &= 0 && \text{on } \partial\Omega, \\ \mathbf{A}_1^{ext} &= 0 && \text{in } \bar{\Omega}^c \end{aligned} \tag{7.6}$$

and

$$\begin{aligned} \Delta \mathbf{A}_2^{int} &= \mathbf{0} && \text{in } \Omega, \\ \Delta \mathbf{A}_2^{ext} &= \mathbf{0} && \text{in } \bar{\Omega}^c, \\ [\mathbf{A}_2] &= \mathbf{0} && \text{on } \partial\Omega, \\ \left[\frac{\partial \mathbf{A}_2}{\partial \mathbf{n}}\right] &= -\mathbf{m} \times \mathbf{n} + \frac{\partial \mathbf{A}_1^{int}}{\partial \mathbf{n}} && \text{on } \partial\Omega, \\ (\mathbf{A}_2^{ext})_j &= \mathcal{O}\left(\frac{1}{\|\mathbf{x}\|}\right) && \text{as } \|\mathbf{x}\| \to \infty \quad \text{for } j = 1, 2, 3. \end{aligned} \tag{7.7}$$

For the specific choice of $\mathbf{m}$ as in (7.1) it holds

$$\nabla \times \mathbf{m} = \mathbf{0}.$$

Therefore,

$$\Delta \mathbf{A}_1^{int} = \mathbf{0} \quad \text{in } \Omega$$

and $\mathbf{A}_1^{int} = \mathbf{A}_1^{ext} = \mathbf{0}$ is a solution of (7.6).
Furthermore, $\partial \mathbf{A}_1^{int} / \partial \mathbf{n} = \mathbf{0}$ and hence

$$\left[\frac{\partial \mathbf{A}_2}{\partial \mathbf{n}}\right] = -\mathbf{m} \times \mathbf{n} = \frac{\mathbf{x}}{\|\mathbf{x}\|} \times \frac{\mathbf{x}}{R} = \mathbf{0}.$$

This yields,

$$\mathbf{A}_2^{int} = \mathbf{A}_2^{ext} = \mathbf{0}$$

and accordingly,

$$\mathbf{A}^{int} = \mathbf{A}^{ext} = \mathbf{0}$$

is an analytically solution of (7.6) and (7.7). As a result, the magnetic induction is

$$\mathbf{b} = \nabla \times \mathbf{A} = \mathbf{0} \quad \text{in } \Omega \cup \bar{\Omega}^c.$$

**Example 2**

Next we consider a uniform magnetization

$$\mathbf{m}(\mathbf{x}) = \begin{cases} (0,0,1)^T & \text{in } \Omega, \\ 0 & \text{else.} \end{cases} \tag{7.8}$$

For the calculation of the scalar potential we use again the splitting Ansatz depicted in (7.2) and (7.3). For a uniform magnetization we have

$$\nabla \cdot \mathbf{m} = 0$$

and therefore the computations reduce to solving

$$-\Delta \phi_1^{int} = 0 \qquad \text{in } \Omega,$$

$$\phi_1^{int} = 0 \quad \text{on } \partial\Omega,$$

the Laplace equation with homogeneous Dirichlet boundary conditions which has the trivial solution

$$\phi_1^{int} = 0.$$

We proceed by solving for $\phi_2$. Firstly, we consider the jump condition on $\partial\Omega$,

$$\left[\frac{\partial \phi_2}{\partial \mathbf{n}}\right] = -\mathbf{m} \cdot \mathbf{n} = -\frac{x_3}{R}. \tag{7.9}$$

Problem (7.3) is linear in $x_3$ which motivates the Ansatz

$$\phi_2^{int}(\mathbf{x}) = B x_3.$$

In the exterior the decay condition has to be fulfilled, so

$$\phi_2^{ext}(\mathbf{x}) = C \frac{x_3}{\|\mathbf{x}\|^3}.$$

Here $B, C \in \mathbb{R}$ are constants which have to be determined. Enforcing the continuity of $\phi_2$ at $\|\mathbf{x}\| = R$ gives

$$B x_3 = C \frac{x_3}{R}$$

and thus $C = BR^3$. Moreover, on $\partial\Omega$ the normal derivatives are

$$\frac{\partial \phi_2^{int}}{\partial \mathbf{n}} = (0,0,B)^T \cdot \frac{\mathbf{x}}{R} = B\frac{x_3}{R},$$

and

$$\frac{\partial \phi_2^{ext}}{\partial \mathbf{n}} = C\left( -\frac{3x_1x_3}{R^5}, -\frac{3x_2x_3}{R^5}, \frac{x_1^2+x_2^2-2x_3^2}{R^5}\right)^T \cdot \frac{\mathbf{x}}{R} = -C\frac{2x_3}{R^4} = -B\frac{2x_3}{R}.$$

Combined with estimate (7.9), it follows that

$$-\frac{x_3}{R} = \left[\frac{\partial \phi_2}{\partial \mathbf{n}}\right] = \frac{\partial \phi_2^{ext}}{\partial \mathbf{n}} - \frac{\partial \phi_2^{int}}{\partial \mathbf{n}} = -\frac{3Bx_3}{R},$$

and consequently,

$$B = \frac{1}{3}, \quad C = \frac{R^3}{3}.$$

The analytic solution for the scalar potential for a uniform magnetization is therefore,

$$\begin{aligned}\phi_{int}(\mathbf{x}) &= \frac{x_3}{3},\\ \phi_{ext}(\mathbf{x}) &= R^3\frac{x_3}{3\|\mathbf{x}\|^3}.\end{aligned}$$

Similar to before, the vector potential is computed using equations (7.6) and (7.7). We start by observing that for a uniform magnetization

$$\nabla \times \mathbf{m} = \mathbf{0}$$

holds. Thus, like in Example 1 $\mathbf{A}_1^{int} = \mathbf{A}_1^{ext} = \mathbf{0}$. For $\mathbf{A}_2$ we again consider the jump condition on the boundary

$$\left[\frac{\partial \mathbf{A}_2}{\partial \mathbf{n}}\right] = -\mathbf{m} \times \mathbf{n} = -\frac{1}{R}(-x_2, x_1, 0)^T.$$

We proceed by forming the Ansatz for $\mathbf{A}_2^{int}$ by

$$\mathbf{A}_2^{int}(\mathbf{x}) = B(-x_2, x_1, 0)^T$$

and for $\mathbf{A}_2^{ext}$ by taking into account the decay conditions from (7.7):

$$\mathbf{A}_2^{ext}(\mathbf{x}) = \frac{C}{\|\mathbf{x}\|^3}(-x_2, x_1, 0)^T,$$

for constants $B, C \in \mathbb{R}$. Ensuring $[\mathbf{A}_2] = \mathbf{0}$ on $\partial\Omega$ leads to

$$B(-x_2, x_1, 0)^T = \frac{C}{R^3}(-x_2, x_1, 0)^T \Rightarrow C = R^3B.$$

Moreover,

$$\frac{\partial \mathbf{A}_2^{int}}{\partial \mathbf{n}} = \frac{B}{R}(-x_2, x_1, 0)^T,$$
$$\frac{\partial \mathbf{A}_2^{ext}}{\partial \mathbf{n}} = \frac{2C}{R^5}(x_2, -x_1, 0)^T = -\frac{2B}{R^2}(-x_2, x_1, 0)^T.$$

This yields,

$$-\frac{1}{R}(-x_2, x_1, 0)^T = \left[\frac{\partial \mathbf{A}_2}{\partial \mathbf{n}}\right] = \frac{\partial \mathbf{A}_2^{ext}}{\partial \mathbf{n}} - \frac{\partial \mathbf{A}_2^{int}}{\partial \mathbf{n}} = -\frac{2B - BR}{R^2}.$$

As a result,

$$B = \frac{1}{3R^2}, \quad C = \frac{R}{3}.$$

After rescaling with $R^2$ we finally get

$$\mathbf{A}^{int}(\mathbf{x}) = \frac{1}{3}(-x_2, x_1, 0)^T,$$
$$\mathbf{A}^{ext}(\mathbf{x}) = \frac{R^3}{3\|\mathbf{x}\|^3}(-x_2, x_1, 0)^T.$$

Hence, the b-field is denoted by

$$\mathbf{b}^{int}(\mathbf{x}) = \nabla \times \mathbf{A}^{int}(\mathbf{x}) = (0, 0, 2/3)^T,$$
$$\mathbf{b}^{ext}(\mathbf{x}) = \nabla \times \mathbf{A}^{ext}(\mathbf{x}) = \frac{R^3}{\|\mathbf{x}\|^5}\big(x_1x_3, x_2x_3, (-x_1^2 - x_2^2 + 2x_3^2)/3\big)^T.$$

# Appendix B

## Netgen/NGSolve

The aim of this chapter is to examine the structure of the open-source library Netgen/NGsolve and to showcase the most important parts of the code used for the calculations in this work.
In general, Netgen and NGsolve are two separate packages, the former is used for mesh generation, the latter is the actual finite element solver. The core computations of Netgen/NGsolve are programmed in C++ but wrapped within a Python interface, which will be observed in the following sections. Further explanations to the contents of this section can be found in the documentation of Netgen/NGsolve [72], as well as in [73] for more details on mesh generation and [74] for NGsolves internal C++ framework.

### Mesh generation

We start by showing a prototype case of a mesh generation process with Netgen. Firstly, a geometry has to be set up. This is done with the implemented library *open cascade technology* (OCCT) [75]. This package provides models of geometric objects together with their properties, i.e. edges, vertices and boundaries. Additionally, interaction between multiple objects, such as intersections or unions, is performed by OCCT as well. The following code generates the unit cube, embedded in another larger cubic domain, like in the setup of the NIST $\mu$MAG standard problem #3:

```
from netgen.occ import *

# inner cube
magnet = Box((-0.5,-0.5,-0.5),(0.5,0.5,0.5))

# outer cube
pp = 2*bricksize
brick = Box((-pp,-pp,-pp),(pp,pp,pp))
air = brick - magnet

geo = Glue([air, magnet])
occgeo = OCCGeometry(geo)
```

Subsequently, Netgen generates a mesh using the so-called *advancing front technique* [73] This means that firstly a 2D boundary mesh is set up on the facets of the domain.

Therefore, the program detects 'special points', intersection points of multiple facets or points with maximal or minimal coordinates, by solving nonlinear optimization problems. Then, edges are defined as the line segments between those special points. For curves, a predictor-corrector method is used, where, starting at an initial point, a small step along the corresponding tangential vector is made, followed by a projection back onto the curve. These edges are then subdivided according to a given maximal mesh size. Finally, a surface mesh is established for every facet, following a set of abstract rules that guarantee well-shaped, equilateral triangles. For more detailed descriptions of these rules, see [73]. To conclude this for 3D geometries, a mesh of tetrahedrons is built inwards from the surface. The following code summarizes the result:

```
magnet.mat('magnet').maxh=h
magnet.faces.name='inner_bound'
brick.faces.name='outer'
air.mat('brick')
mesh = Mesh(occgeo.GenerateMesh(maxh=2, grading=0.3))
```

Here, 'maxh' gives the maximal mesh size and 'grading' determines a mesh refinement towards the outer boundary. Additionally, we introduced names for the different domains and boundaries.

### Finite Element Method with NGSolve

The first step for applying a finite element scheme in NGSolve is to define a finite element space. As we are always working in 3D and with polynomial functions of degree $d = 1$ we implement:

```
fes_mag = VectorH1(mesh, order=1, definedon='magnet')
fes_air = VectorH1(mesh, order=1, definedon='brick', dirichlet='outer')
fes = FESpace([fes_mag, fes_air])
```

Here, the approach of two separated finite element spaces for inner and outer domain as introduced in section 6.1.3 is shown. Note, that we defined Dirichlet boundary conditions on the outer boundary. The expression 'VectorH1' defines a subspace of the vector valued $(H^1(\Omega))^3$ space.
To define functions on those space one uses the concept of *coefficient functions*. These are functions stored as expression trees of coordinates and binary operations. For example the function

$$f(x) = x(2 + x)$$

is stored as [72]

```
coef binary operation '*', real
  coef coordinate x, real
  coef binary operation '+', real
    coef 2, real
    coef coordinate x, real.
```

Coefficient functions can also be defined discontinuous and domain-wise, for example in the case of an initial flower state, we write

```
m=mesh.MaterialCF({'magnet' : (x*z,y*z+1/8*y**3*z**3,1),
                   'brick': (0,0,0)})
```

The following code shows the assembly of the element matrix for the discretized stray field energy. For simplicity, the case of a fixed vector potential is depicted:

```
def stiff_stray2(fes):
    (u_mag,_), (v_mag,_) = fes.TnT()

    a = BilinearForm(fes)
    a += InnerProduct(u_mag, v_mag)*dx(definedon='magnet')
    a.Assemble()
    return a
```

Here, '$u_{mag}$' and '$v_{mag}$' describe trial and test functions, which will be combined in a symbolic bilinear form, only defined on the magnetic domain. In 'a.Assemble()' this symbolic form is evaluated element-wise in the local basis elements. On each element a local matrix is calculated and afterwards assembled into a global matrix by global degree of freedom numbering. Multiple local computations are possible in parallel due to coloring. For calculation of the gradients, automatic differentiation is used as well as an integration rule object for evaluating integrals numerically. More specifically, the integration rule provides an integration point which contains the corresponding coordinates on the reference element (see (3.12)) as well as quadrature weights. Then the quadrature is applied to the mapping onto the physical element, i.e. to evaluate the integral of a function $f : \mathbb{R}^n \to \mathbb{R}$ on a mesh element $T$,

$$\int_T f \, d\mathbf{x},$$

the integration rule computes

$$\sum_{i=1}^{N_q} w_i f(F_T(\hat{\mathbf{x}}_i)) |det J_T(\hat{\mathbf{x}}_i)|.$$

Here,

$$\mathbf{x} = F_T(\hat{\mathbf{x}})$$

is the transformation from the reference element onto the physical element (see (3.14)) and $J_T$ its Jacobian, again computed by automatic differentiation [74].
The matrix 'a' is then stored in an NGsolve internal sparse matrix format, but with the following line of code it can be converted into a SciPy sparse matrix:

```
M = csr_matrix(a.mat.CSR())
```

The resulting matrix is block diagonal with three blocks each corresponding to one coordinate, as depicted in Figure 3.3. Assembling load vectors follows the same scheme:

```
def load_vector(fes, b_mag):
    (v_mag,_) = fes.TestFunction()
    f = LinearForm(fes)
    f += InnerProduct(b_mag, v_mag)*dx(definedon='magnet')
    f.Assemble()
    return f.vec.FV().NumPy()
```

From here, one can work within NumPy's and SciPy's well known framework to establish the iterative techniques presented in this thesis.

### Matrix Assembling

In this section the code of the specific element matrices established in Chapter 4 is displayed. Only the joint minimization procedure is shown, as the matrices for the iterative methods follow directly from the presented code.

```
from ngsolve import *
import numpy as np
from scipy.linalg import *
from scipy.sparse import bmat

#Stray Field Energy (fixed magnetization)
def stiff_stray(fes):
    (u_mag,_), (v_mag,_) = fes.TnT()
    a = BilinearForm(fes)
    a += InnerProduct(u_mag, v_mag) * dx(definedon='magnet')
    with TaskManager():
        a.Assemble()

    return a

#Anisotropy Energy
def stiff_an(fes, Q):

    (u_mag,_), (v_mag, _) = fes.TnT()
    a = BilinearForm(fes)

    a += 2 * Q * InnerProduct(u_mag, v_mag) * dx(definedon='magnet')
    with TaskManager():
        a.Assemble()

    # Get block diagonal form
    # DOF blocks:
    # magnet_x: 0:n_block
    # magnet_y: n_block:2*n_block
    # magnet_z: 2*n_block:3*n_block
    # air_x: 3*n_block:4*n_block
    # air_y: 4*n_block:5*n_block
    # air_z: 5*n_block:6*n_block
    # Keep only x,y components -> equivalent to keeping the z-component
    # with negative sign as in Chapter 4
```

```
36
37      spmat = csr_matrix(a.mat.CSR(),shape=(fes.ndof, fes.ndof))
38      n = spmat.shape[0]
39      n_block = n // 6
40
41      cols_to_keep = (list(range(0, 2*n_block)) +
42                      list(range(3*n_block, 5*n_block)))
43
44      def zero_columns(spmat, cols_to_keep):
45          # Zero out all columns not in cols_to_keep
46
47          _,n_cols = spmat.shape
48          mask = np.zeros(n_cols, dtype=spmat.dtype)
49          mask[cols_to_keep] = 1.0
50          return spmat.multiply(mask)
51
52
53      spmat_filtered = zero_columns(spmat, cols_to_keep)
54      return spmat_filtered
55
56  #Exchange Energy
57  def stiff_ex(fes, A_ex):
58      (u_mag,_), (v_mag,_) = fes.TnT()
59      a = BilinearForm(fes)
60      a += 2 * A_ex * InnerProduct(Grad(u_mag), Grad(v_mag)) * dx(definedon
        ='magnet')
61      with TaskManager():
62          a.Assemble()
63
64      return a
65
66  # Magnetization assembling
67  def mat_m(fes, A_ex, Q):
68      M=stiff_stray(fes)
69      M_sp=csr_matrix(M.mat.CSR(),shape=(fes.ndof, fes.ndof))
70      E= stiff_ex(fes, A_ex)
71      E_sp=csr_matrix(E.mat.CSR(),shape=(fes.ndof, fes.ndof))
72      A_sp= stiff_an(fes, Q)
73      H= M_sp +E_sp+ A_sp
74
75      n=H.shape[0]
76      zero = csr_matrix((n, n))
77      return bmat([[H, zero],[zero, zero]], format='csr')
78
79  # Vector Potential assembling
80  def mat_A(fes, alpha=1000):
81      (u_mag, u_air), (v_mag, v_air) = fes.TnT()
82      a = BilinearForm(fes)
83      a+= InnerProduct(Grad(u_mag), Grad(v_mag))*dx('magnet')
84      a+= InnerProduct(Grad(u_air), Grad(v_air))*dx('brick')
85
86      # Coupling for the boundary terms
87      h = specialcf.mesh_size #minimal mesh size
```

```
    a += alpha/h * InnerProduct(u_mag - u_air, v_mag - v_air) * ds('
    inner_bound')
    with TaskManager():
        a.Assemble()

    F=csr_matrix(a.mat.CSR(),shape=(fes.ndof, fes.ndof))
    n=F.shape[0]
    zero = csr_matrix((n, n))
    return bmat([[zero, zero],[zero, F]], format="csr")

# Mixed term assembling
def mat_mix(fes):
    (u_mag,_),(v_mag,_) = fes.TnT()
    a = BilinearForm(fes, symmetric=False) #matrix is antisymmetric
    a += 0.5*InnerProduct(v_mag, curl_func(u_mag))*dx(definedon='magnet')
    with TaskManager():
        a.Assemble()

    K = csr_matrix(a.mat.CSR(),shape=(fes.ndof, fes.ndof))
    n=K.shape[0]
    zero = csr_matrix((n, n))
    return bmat([[zero, K],[K.T,zero]], format="csr")


#Final matrix
def mat_sum(fes, A_ex, Q):
    H= mat_m(fes, A_ex, Q)
    F= mat_A(fes)
    K=mat_mix(fes)
    return H+F-2*K
```

## Iterative Methods - Code

In this section the code for the algebraic methods used in this work is presented.

### Quadratic Penalty Method

```
import numpy as np
from scipy.linalg import *
from scipy.optimize import minimize

# objective function & gradient
def Func(x, H):
    return 0.5 * x @ (H @ x)

def grad_Func(x, H):
    return H @ x

# penalty term
```

```
def P_split(m, mu, idx_mag, idx_air):
    # m = coefficient vector of magnetization
    # idx_mag, idx_air = indices of nodes inside/outside the magnet

    # magnet block
    m_mag = m[idx_mag]
    m_mag2 = m_mag**2
    N_mag = len(m_mag2) // 3
    mag_nodes = [
        m_mag2[:N_mag],
        m_mag2[N_mag:2*N_mag],
        m_mag2[2*N_mag:]
    ]
    norms_mag = np.sum(mag_nodes, axis=0)
    P_mag = np.sum((0.5*(norms_mag - 1))**2)

    # air block
    m_air = m[idx_air]
    m_air2 = m_air**2
    N_air = len(m_air2) // 3

    air_nodes = [
        m_air2[:N_air],
        m_air2[N_air:2*N_air],
        m_air2[2*N_air:]
    ]
    norms_air = np.sum(air_nodes, axis=0)
    P_air = np.sum((0.5*norms_air)**2)

    return mu/2 * (P_mag + P_air)

# joint coupling
def P_simul(x, mu, shift, idx_mag, idx_air):
    # x = (c_m, c_A)
    # shift = last index of the magnetization vector
    m=x[:shift]
    pen= P_split(m, mu, idx_mag, idx_air)
    return pen

# gradient of penalty term
def grad_P_split(m, mu, idx_mag, idx_air):
    grad = np.zeros_like(m)

    # magnet block
    m_mag = m[idx_mag]
    N_mag = len(m_mag) // 3

    mx = m_mag[:N_mag]
    my = m_mag[N_mag:2*N_mag]
    mz = m_mag[2*N_mag:]

    norms_mag = mx**2 + my**2 + mz**2
    factor_mag = (norms_mag - 1.0)
```

```
66
67      grad_mag = np.concatenate([
68          mx * factor_mag,
69          my * factor_mag,
70          mz * factor_mag
71      ])
72
73      grad[idx_mag] = mu/2 * grad_mag
74
75      # air block
76      m_air = m[idx_air]
77      N_air = len(m_air) // 3
78
79      ax = m_air[:N_air]
80      ay = m_air[N_air:2*N_air]
81      az = m_air[2*N_air:]
82
83      norms_air = ax**2 + ay**2 + az**2
84      factor_air = norms_air
85
86      grad_air = np.concatenate([
87          ax * factor_air,
88          ay * factor_air,
89          az * factor_air
90      ])
91
92      grad[idx_air] = mu/2 * grad_air
93
94      return grad
95
96  # gradient coupling
97  def grad_P_simul(x, mu, shift, idx_mag, idx_air):
98      grad = np.zeros_like(x)
99      m=x[:shift]
100     grad_m = grad_P_split(m, mu, idx_in_m, idx_out_m)
101     grad[:shift] = grad_m
102     return grad
103
104
105 # Penalty function
106 def Q(x_np, H, mu, shift, idx_mag, idx_air):
107     return Func(x_np, H) + P_simul(x_np, mu, shift, idx_mag, idx_air)
108
109 def Q_grad(x_np, H, mu, shift, idx_mag, idx_air):
110     return grad_Func(x_np, H) + grad_P_simul(x_np, mu, shift, idx_mag,
        idx_air)
111
112 #minimzer from SciPy
113 def scipy_minimize(x_np, mu, shift, idx_mag, idx_air, H):
114     def f_wrapped(x):
115         return Q(x,H, mu, shift, idx_mag, idx_air)
116     def fprime_wrapped(x):
117         return Q_grad(x, H, mu, shift, idx_mag, idx_air)
```

```
118
119     res= minimize(f_wrapped, x_np, jac= fprime_wrapped,
120             method='L-BFGS-B')
121     return res.x
122
123 #constraint vector
124 def constraint_mag(m_mag):
125     N = len(m_mag) // 3
126     norms = (m_mag[:N]**2 +
127              m_mag[N:2*N]**2 +
128              m_mag[2*N:]**2)
129     return 0.5 * (norms - 1.0)
130
131 def constraint_air(m_air):
132     N = len(m_air) // 3
133     norms = (m_air[:N]**2 +
134              m_air[N:2*N]**2 +
135              m_air[2*N:]**2)
136     return 0.5 * norms
137
138 #returns max_j |h_j(m)m_j| for stopping criterion
139 def feasibility_split(m, idx_mag, idx_air):
140     # magnet
141     m_mag = m[idx_mag]
142     c_mag = constraint_mag(m_mag)
143     m_mag_int = interleave_block(m_mag)
144     prod_mag = c_mag.reshape(-1,1) * m_mag_int
145     feas_mag = np.max(np.linalg.norm(prod_mag, axis=1))
146
147     # air
148     m_air = m[idx_air]
149     c_air = constraint_air(m_air)
150     m_air_int = interleave_block(m_air)
151     prod_air = c_air.reshape(-1,1) * m_air_int
152     feas_air = np.max(np.linalg.norm(prod_air, axis=1))
153
154     return max(feas_mag, feas_air)
155
156
157 #full penalty method
158 def qpm(mu0, x_np, H, shift, idx_mag, idx_air, tau= 1e-02,
159         max_iter=30, tol_c=1e-06, tol_grad=1e-08, tol_2 = 1e-08,
160         stagnation_tol= 1e-10,
161         gamma= 0.5):
162
163     v_prev=1e-10
164     mu=mu0
165     for k in range(max_iter):
166         print('begin minimization')
167         x_np= scipy_minimize(x_np, mu, shift, idx_mag, idx_air, H)
168         print('minimization ended')
169         m_np = x_np[:shift]
170         m_mag=m_np[idx_mag]
```

```
        m_air=m_np[idx_air]
        ck_mag= constraint_mag(m_mag)
        ck_air=constraint_air(m_air)
        ck= np.concatenate((ck_mag,ck_air), axis=None)
        vk= np.linalg.norm(ck)
        print(f'\nPenalty method iteration {k+1}: Constraint violation: {
    vk}\n')

        grad = Q_grad(x_np, H, mu, shift, idx_mag, idx_air)
        norm_grad= np.linalg.norm(grad)
        delta = feasibility_split(m_np, idx_mag, idx_air)

        if norm_grad < tol_grad and vk < tol_c:
            print('Stopping tolerance reached. Return KKT point.')
            return x_np

        if delta < tol_2 and abs(vk - v_prev)< stagnation_tol:
            print('Stopping tolerance reached. Return infeasible point.')
            return x_np

        if vk <= gamma * v_prev:
            print('Infeasibility reduced sufficiently, keep mu.')
            Mk=1
        else:
            print('Update mu')
            Mk=5

        mu= Mk*mu
        print('New mu:', mu)
        if math.isnan(mu) == True:
            mu=mu0
        v_prev=vk
        tau=max(tau / np.sqrt(mu), 1e-12)

    print('Maximal iterations reached. Return x.')
    return x_np
```

## Augmented Lagrangian Method

```
# helper for norm at nodes
def node_norms(m_block):
    N = len(m_block) // 3
    mx = m_block[:N]
    my = m_block[N:2*N]
    mz = m_block[2*N:]
    return mx**2 + my**2 + mz**2

# Lagrangian part
def L_split(m, lam, idx_mag, idx_air):
    # m = [m_mag | m_air]
    # lam = [lam_mag | lam_air]
```

```
    m_mag = m[idx_mag]
    m_air = m[idx_air]
    lam_mag = lam[:len(m_mag)//3]
    lam_air = lam[len(m_mag)//3:]

    #  magnet constraint |m|^2 - 1
    norms_mag = node_norms(m_mag)
    L_mag = np.sum(lam_mag * (norms_mag - 1.0))

    # air constraint |m|^2
    norms_air = node_norms(m_air)
    L_air = np.sum(lam_air * norms_air)

    return 0.5 * (L_mag + L_air)

# gradient of Lagrangian
def grad_L_split(m, lam, idx_mag, idx_air):

    m_mag = m[idx_mag]
    m_air = m[idx_air]

    N_mag = len(m_mag) // 3

    lam_mag = lam[:N_mag]
    lam_air = lam[N_mag:]

    # expand multipliers per component
    lam_mag_all = np.tile(lam_mag, 3)
    lam_air_all = np.tile(lam_air, 3)

    # block gradients
    grad_mag = m_mag * lam_mag_all
    grad_air = m_air * lam_air_all

    # assemble global gradient
    grad = np.zeros_like(m)
    grad[idx_mag] = grad_mag
    grad[idx_air] = grad_air

    return grad


def L_simul(x, lam, shift, idx_mag, idx_air):
    m= x[:shift]
    return L_split(m, lam, idx_mag, idx_air)

def grad_L_simul(x, lam, shift, idx_mag, idx_air):
    grad = np.zeros_like(x)
    m=x[:shift]
    grad_m = grad_L_split(m, lam, idx_mag, idx_air)
    grad[:shift] = grad_m
    return grad

```

```
66
67 # new objective function & gradient
68 def Lagrangian_value(x_np,H, mu, lam, shift, idx_mag, idx_air):
69     return (Func(x_np, H) + P_simul(x_np, mu, shift,idx_mag, idx_air)+
70             L_simul(x_np, lam, shift,idx_mag, idx_air))
71
72 def Lagrangian_grad(x_np,H, mu, lam, shift,idx_mag, idx_air):
73     return (grad_Func(x_np, H) + grad_P_simul(x_np, mu, shift,idx_mag,
    idx_air)+
74             grad_L_simul(x_np, lam, shift,idx_mag, idx_air))
75
76
77 def scipy_minimize(x_np, mu, lam, shift,idx_mag, idx_air, H):
78
79     def f_wrapped(x):
80         return Lagrangian_value(x,H, mu, lam, shift, idx_mag, idx_air)
81     def fprime_wrapped(x):
82         return Lagrangian_grad(x, H, mu, lam, shift, idx_mag, idx_air)
83
84     res= minimize(f_wrapped, x_np, jac= fprime_wrapped, method='L-BFGS-B'
    )
85
86     return res.x
87
88 # full aLm framework
89 def aLm(mu0, lam0, x_np, H, shift, idx_mag, idx_air, max_iter=200,
90         tol_c=1e-06, tol_grad=1e-06, tol_2 = 1e-08,
91         stagnation_tol= 1e-10, gamma= 0.5):
92     v_prev=1e-10
93     mu=mu0
94     lam=lam0
95     eps=1e-05
96     nu=1e-05
97     for k in range(max_iter):
98         print('begin minimization')
99         x_np= scipy_minimize(x_np, mu, lam, shift, idx_mag, idx_air, H)
100        m_np=x_np[:shift]
101        m_mag=m_np[idx_mag]
102        m_air=m_np[idx_air]
103        ck_mag= constraint_mag(m_mag)
104        ck_air=constraint_air(m_air)
105        ck= np.concatenate((ck_mag,ck_air), axis=None)
106        vk= np.linalg.norm(ck)
107        print(f'''\nAugmented Lagr. method iteration {k+1}:
108              Constraint violation: {vk}\n''')
109
110        grad = Lagrangian_grad(x_np, H, mu, lam, shift, idx_mag, idx_air)
111        norm_grad= np.linalg.norm(grad)
112        delta = feasibility_split(m_np, idx_mag, idx_air)
113        print('Norm of grad:',norm_grad)
114
115        if norm_grad < tol_grad and vk < tol_c:
116            print('Stopping tolerance reached. Return KKT point.')
```

```
            return x_np

        if delta < tol_2 and abs(vk - v_prev)< stagnation_tol:
            print('Stopping tolerance reached. Return infeasible point.')
            return x_np

        #Update scheme
        if vk < eps:
            print('Update lambda')
            lam = lam + mu*ck
            nu=nu/mu
            eps= eps/mu**(1/2)
        else:
            print('Update mu')
            mu= mu*5
            nu= 1/mu
            eps= 1/mu**(1/2)


        print(f'New mu: {mu}.')
        if math.isnan(mu) == True:
            mu=mu0
        v_prev=vk

    return x_np
```

## Projected CG-method

```
from scipy.sparse.linalg import  spilu, LinearOperator
from scipy.sparse import eye

# helper functions for changing order of coeff. vector
# NGsolve saves vectors coordinatewise and not nodewise
def interleave_block(m_block):
    N = len(m_block) // 3
    return np.vstack((
        m_block[:N],
        m_block[N:2*N],
        m_block[2*N:]
    )).T

def deinterleave_block(m_inter):
    return np.concatenate((
        m_inter[:,0],
        m_inter[:,1],
        m_inter[:,2]
    ))

# project gradient
def project_gradient(x, grad, shift, idx_mag, idx_air):
    m = x[:shift]
```

```
    grad_proj = grad.copy()

    m_mag = m[idx_mag]
    g_mag = grad[idx_mag]

    m_int = interleave_block(m_mag)
    g_int = interleave_block(g_mag)

    dot = np.sum(g_int * m_int, axis=1, keepdims=True)
    g_proj_int = g_int - dot * m_int

    grad_proj[idx_mag] = deinterleave_block(g_proj_int)

    # Air block of m (no projection)
    grad_proj[idx_air] = grad[idx_air]

    return grad_proj

# normalize iterates (only the magnetization)
def normalize_split(m, idx_mag, idx_air):

    m_new = np.zeros_like(m)

    # magnet block
    m_mag = m[idx_mag]
    m_int = interleave_block(m_mag)

    norms = np.linalg.norm(m_int, axis=1, keepdims=True)
    norms_safe = np.where(norms == 0, 1, norms)

    m_mag_norm = m_int / norms_safe
    m_new[idx_mag] = deinterleave_block(m_mag_norm)


    return m_new

def normalize_full(x, shift, idx_mag, idx_air):
    m = x[:shift]
    A = x[shift:]

    m = normalize_split(m, idx_mag, idx_air)
    return np.concatenate([m, A])

# backtracking line search
def wolfe_line_search(x, d, H, grad, idx_mag, idx_air, shift,
                              alpha0=1.0, tau=0.5,
                              c1=0.1,
                              maxiter=30):
    Fm = Func(x,H)
    slope0 = np.dot(grad, d)

    alpha = alpha0
    for j in range(maxiter):
```

```
77
78          x_trial = normalize_full(x+alpha*d, shift, idx_mag, idx_air)
79          F_trial = Func(x_trial, H)
80
81          if F_trial > Fm + c1 * alpha * slope0:
82              alpha *= tau
83              continue
84
85
86          print(f'Wolfe line search accepted alpha={alpha} after {j}
    backtracks')
87          return alpha
88
89      print('Wolfe line search did not converge, fallback alpha=', alpha)
90      return alpha
91
92
93  # incomplete LU precon
94  def precon(H, delta = 1e-08):
95      # regularize system matrix
96      H_reg = H + delta * eye(H.shape[0], format="csc")
97
98      ilu = spilu(
99          H_reg.tocsc(),
100         drop_tol=1e-2,
101         fill_factor=5,
102         permc_spec='MMD_AT_PLUS_A'
103     )
104
105     H_prec = LinearOperator(H.shape, matvec=ilu.solve)
106     return H_prec
107
108 # full pCG framework
109 def pCG_simul(x0, H, idx_mag, idx_air, H_prec, shift, max_iter=2000,
110                           tol=1e-6):
111     x = normalize_full(x0, shift, idx_mag, idx_air)
112     grad = grad_Func(x, H)
113     grad_proj = project_gradient(x, grad, shift, idx_mag, idx_air)
114     y = H_prec(grad_proj)
115     d = -y
116     alpha=1
117     for j in range(max_iter):
118         alpha = wolfe_line_search(x, d, H, grad, idx_mag,
119                                       idx_air, shift, alpha0=alpha)
120         x_new= normalize_full(x+ alpha*d, shift, idx_mag, idx_air)
121         grad_new = grad_Func(x_new, H)
122         grad_proj_new = project_gradient(x_new, grad_new, shift,
123                                                   idx_mag, idx_air)
124
125         y_new=H_prec(grad_proj_new)
126
127         delta_g=grad_proj_new-grad_proj
128         num=np.dot(delta_g,y_new)
```

```
129         den=np.dot(delta_g,d)
130         beta=num/den if den!=0 else 0.0
131 
132         # restart condition
133         if np.dot(y_new,grad_proj_new) <= np.dot(y_new, grad_proj):
134             beta = 0.0
135 
136         d_new = -y_new + beta * d
137 
138         print(f'Iteration {j+1}: Norm of Gradient:',
139               np.linalg.norm(grad_proj_new))
140 
141         if np.linalg.norm(grad_proj_new) < tol:
142             print(f'Converged at iteration {j}')
143             break
144 
145         x=x_new
146         grad = grad_new
147         grad_proj = grad_proj_new
148         y = y_new
149         d= d_new
150 
151     return x
```

# Bibliography


[1] OpenAI. *ChatGPT*. URL: https://chatgpt.com/de-DE/ (visited on 04/08/2026).

[2] J. Fischbacher, A. Kovacs, M. Gusenbauer, H. Oezelt, L. Exl, S. Bance, and T. Schrefl. "Micromagnetics of rare-earth efficient permanent magnets". In: *Journal of Physics D: Applied Physics* 51.19 (May 2018). arXiv:1903.11922 [cond-mat], p. 193002. URL: http://arxiv.org/abs/1903.11922 (visited on 01/23/2025).

[3] Statista. *Neodymium-iron-boron permanent magnet demand growth rate worldwide*. en. URL: https://www.statista.com/statistics/1134603/neodymium-iron-boron-permanent-magnet-demand-growth-rate-worldwide/ (visited on 03/27/2026).

[4] Statista. *Rare earth permanent magnet demand for industrial applications globally 2025*. en. URL: https://www.statista.com/statistics/693363/industrial-permanent-magnet-rare-earth-demand-worldwide/ (visited on 03/30/2026).

[5] Statista. *Rare earth permanent magnet demand for wind turbines globally 2025*. en. URL: https://www.statista.com/statistics/693305/wind-turbine-permanent-magnet-rare-earth-demand-worldwide/ (visited on 03/30/2026).

[6] Statista. *Rare earth permanent magnet demand for automotive purposes globally 2025*. en. URL: https://www.statista.com/statistics/693357/automotive-permanent-magnet-rare-earth-demand-worldwide/ (visited on 03/30/2026).

[7] D. A. Gkika, M. Chalaris, and G. Z. Kyzas. "Review of methods for obtaining rare earth elements from recycling and their impact on the environment and human health". In: *Processes* 12.6 (2024). Number: 1235. URL: https://www.mdpi.com/2227-9717/12/6/1235.

[8] S. Schaffer, N. J. Mauser, T. Schrefl, D. Suess, and L. Exl. "Machine learning methods for the prediction of micromagnetic magnetization dynamics". In: *IEEE Transactions on Magnetics* 58.2 (Feb. 2022). arXiv:2103.09079 [physics], pp. 1–6. URL: http://arxiv.org/abs/2103.09079 (visited on 01/23/2025).

[9] L. Exl, D. Suess, and T. Schrefl. "Micromagnetism". In: *Handbook of magnetism and magnetic materials*. Ed. by M. Coey and S. Parkin. Cham: Springer International Publishing, 2020, pp. 1–44. URL: https://doi.org/10.1007/978-3-030-63101-7_7-1.

[10] C. Abert, L. Exl, G. Selke, A. Drews, and T. Schrefl. "Numerical Methods for the Stray-Field Calculation: A Comparison of recently developed Algorithms". In: *Journal of Magnetism and Magnetic Materials* 326 (Jan. 2013). arXiv:1204.4302 [physics], pp. 176–185. URL: http://arxiv.org/abs/1204.4302 (visited on 01/22/2025).

[11] T. Schrefl, G. Hrkac, S. Bance, D. Suess, O. Ertl, and J. Fidler. "Numerical methods in micromagentics (finite element method)". In: *Handbook of magnetism and advanced magentic materials*. 2007.

[12] L. Exl, J. Fischbacher, A. Kovacs, H. Oezelt, M. Gusenbauer, and T. Schrefl. "Preconditioned nonlinear conjugate gradient method for micromagnetic energy minimization". In: *Computer Physics Communications* 235 (Feb. 2019). arXiv:1801.03690 [physics], pp. 179–186. URL: http://arxiv.org/abs/1801.03690 (visited on 01/23/2025).

[13] W. F. Brown. *Micromagnetics*. eng. Interscience tracts on physics and astronomy ; 18. New York, NY [u.a.]: Interscience Publ., 1963.

[14] P. Asselin and A. Thiele. "On the field Lagrangians in micromagnetics". In: *IEEE Transactions on Magnetics* 22.6 (Nov. 1986). Conference Name: IEEE Transactions on Magnetics, pp. 1876–1880. URL: https://ieeexplore.ieee.org/document/1064664/?arnumber=1064664 (visited on 01/23/2025).

[15] S. Schaffer, T. Schrefl, H. Oezelt, N. J. Mauser, and L. Exl. *Physics aware machine learning for micromagnetic energy minimization: recent algorithmic developments*. arXiv:2409.12877 [physics]. Sept. 2024. URL: http://arxiv.org/abs/2409.12877 (visited on 01/16/2025).

[16] A. Kovacs, L. Exl, A. Kornell, J. Fischbacher, M. Hovorka, M. Gusenbauer, L. Breth, H. Oezelt, M. Yano, N. Sakuma, A. Kinoshita, T. Shoji, A. Kato, and T. Schrefl. "Conditional physics informed neural networks". In: *Communications in Nonlinear Science and Numerical Simulation* 104 (Jan. 2022). arXiv:2104.02741 [cs], p. 106041. URL: http://arxiv.org/abs/2104.02741 (visited on 01/23/2025).

[17] L. Landau and E. Lifshitz. "Phys. Z. Sowjetunion". In: *Physikalische Zeitschrift der Sowjetunion* 8 (1935), p. 153.

[18] H. Kronmüller. "General Micromagnetic Theory and Applications". In: *Materials Science and Technology*. May 2019, pp. 1–43. URL: https://doi.org/10.1002/9783527603978.mst0460 (visited on 09/25/2025).

[19] W. F. Brown Jr. "Micromagnetics, Domains, and Resonance". In: *Journal of Applied Physics* 30.4 (Apr. 1959), S62–S69. URL: https://doi.org/10.1063/1.2185970 (visited on 09/23/2025).

[20] S. Pollok, S. Kotewitz, N. Lassen, and R. Bjørk. "Transformer Neural Networks for Predicting Magnetization Dynamics". In: *67nd Annual Conference on Magnetism and Magnetic Materials* (2022), pp. 256–257. (Visited on 10/31/2022).

[21] C.-M. Pfeiler, M. Ruggeri, B. Stiftner, L. Exl, M. Hochsteger, G. Hrkac, J. Schöberl, N. J. Mauser, and D. Praetorius. "Computational micromagnetics with Commics". en. In: *Computer Physics Communications* 248 (Mar. 2020), p. 106965. URL: https://linkinghub.elsevier.com/retrieve/pii/S0010465519303145 (visited on 01/27/2025).

[22] W. Heisenberg. "Zur Theorie des Ferromagnetismus". In: *Zeitschrift für Physik* 49.9 (Sept. 1928), pp. 619–636. URL: https://doi.org/10.1007/BF01328601.

[23] S. A. Schaffer. "Physics informed machine learning in the field of micromagnetism". en. In: (2024). Publisher: :none. URL: https://utheses.univie.ac.at/detail/73920 (visited on 03/04/2025).

[24] P. Gruszecki, C. Banerjee, M. Mruczkiewicz, O. Hellwig, A. Barman, and M. Krawczyk. "The influence of the internal domain wall structure on spin wave band structure in periodic magnetic stripe domain patterns". en. In: *Solid State Physics*. Vol. 70. Elsevier, 2019, pp. 79–132. URL: https://linkinghub.elsevier.com/retrieve/pii/S0081194719300037 (visited on 09/23/2025).

[25] C. W. Abert. "Discrete mathematical concepts in micromagnetic computations". phd. Staats-und Universitätsbibliothek Hamburg Carl von Ossietzky, 2013.

[26] B. D. Cullity and C. D. Graham. *Introduction to magnetic materials*. en. Second edition. Piscataway: IEEE Press, 2009.

[27] J. D. Jackson. *Classical electrodynamics*. eng. 3. ed., [Nachdr.] Hoboken, NY: Wiley, 2009.

[28] I. Tsukerman, A. Plaks, and H. N. Bertram. "Multigrid methods for computation of magnetostatic fields in magnetic recording problems". In: *Journal of Applied Physics* 83.11 (June 1998), pp. 6344–6346. URL: https://doi.org/10.1063/1.367730 (visited on 03/26/2026).

[29] D. Fredkin and T. Koehler. "Hybrid method for computing demagnetizing fields". In: *IEEE Transactions on Magnetics* 26.2 (Mar. 1990). Conference Name: IEEE Transactions on Magnetics, pp. 415–417. URL: https://ieeexplore.ieee.org/document/106342/?arnumber=106342 (visited on 01/23/2025).

[30] L. Exl and T. Schrefl. "Non-uniform FFT for the finite element computation of the micromagnetic scalar potential". In: *Journal of Computational Physics* 270 (Aug. 2014). arXiv:1305.3162 [physics], pp. 490–505. URL: http://arxiv.org/abs/1305.3162 (visited on 01/23/2025).

[31] B. Livshitz, A. Boag, H. N. Bertram, and V. Lomakin. "Nonuniform grid algorithm for fast calculation of magnetostatic interactions in micromagnetics". In: *Journal of Applied Physics* 105.7 (Mar. 2009), p. 07D541. URL: https://doi.org/10.1063/1.3076048 (visited on 03/26/2026).

[32] L. Greengard and V. Rokhlin. "A fast algorithm for particle simulations". In: *Journal of Computational Physics* 73.2 (Dec. 1987), pp. 325–348. URL: https://www.sciencedirect.com/science/article/pii/0021999187901409.

[33] L. Exl, W. Auzinger, S. Bance, M. Gusenbauer, F. Reichel, and T. Schrefl. "Fast stray field computation on tensor grids". In: *Journal of Computational Physics* 231.7 (Apr. 2012), pp. 2840–2850. URL: https://www.sciencedirect.com/science/article/pii/S0021999111007510.

[34] M. Razaviyayn, T. Huang, S. Lu, M. Nouiehed, M. Sanjabi, and M. Hong. "Non-convex Min-Max Optimization: Applications, Challenges, and Recent Theoretical Advances". In: *IEEE Signal Processing Magazine* 37.5 (Sept. 2020), pp. 55–66. URL: http://arxiv.org/abs/2006.08141 (visited on 03/26/2026).

[35] A. Hubert and R. McMichael. *muMAG Standard Problem #3*. URL: https://www.ctcms.nist.gov/~rdm/mumag.org.html.

[36] R. Hertel and H. Kronm. "Finite element calculations on the single-domain limit of a ferromagnetic cubeFa solution to mMAG Standard Problem No. 3". en. In: *Journal of Magnetism and Magnetic Materials 238* (2002), pp. 185–199.

[37] M. E. Schabes and H. N. Bertram. "Magnetization processes in ferromagnetic cubes". In: *Journal of Applied Physics* 64.3 (Aug. 1988), pp. 1347–1357. URL: https://doi.org/10.1063/1.341858 (visited on 03/18/2026).

[38] R. P. Cowburn and M. E. Welland. "Micromagnetics of the single-domain state of square ferromagnetic nanostructures". In: *Physical Review B* 58.14 (Oct. 1998), pp. 9217–9226. URL: https://link.aps.org/doi/10.1103/PhysRevB.58.9217.

[39] L. Exl. "Tensor grid methods for micromagnetic simulations". en. PhD thesis. TU Wien, 2014. URL: https://repositum.tuwien.at/handle/20.500.12708/5669 (visited on 05/20/2025).

[40] K.-J. Bathe. *Finite element procedures*. eng. Boston, Mass.: Bathe, 2006.

[41] J. N. Reddy. *Introduction to the Finite Element Method, Third Edition*. eng. 3rd edition. New York, N.Y: McGraw-Hill Education, 2006.

[42] L. C. Evans. *Partial differential equations*. 2nd ed. Graduate studies in mathematics v. 19. Providence, R.I: American Mathematical Society, 2010.

[43] R. A. Adams and J. J. F. Fournier, eds. *Sobolev spaces*. eng. 2nd ed. Pure and applied mathematics v. 140. Amsterdam Boston: Academic Press, 2003.

[44] A. Quarteroni. *Numerical Models for Differential Problems*. en. Milano: Springer Milan, 2014. URL: http://link.springer.com/10.1007/978-88-470-5522-3 (visited on 03/03/2025).

[45] S. Sauter, C. Schwab, and S. Sauter. *Boundary element methods*. Softcover repr. of the hardcover 1st ed. 2010. Springer Series in Computational Mathematics 39. Berlin Heidelberg: Springer, 2010.

[46] D. Braess. *Finite elements: theory, fast solvers, and applications in elasticity theory*. eng. Trans. by L. L. Schumaker. Third edition. Cambridge: Cambridge University Press, 2010.

[47] A. Quarteroni and A. Valli. *Numerical approximation of partial differential equations*. eng. Springer series in computational mathematics 23. Berlin New York Paris [etc.]: Springer, 1994.

[48] M. G. Larson and F. Bengzon. *The finite element method: theory, implementation, and applications*. eng. Texts in computational science and engineering 10. Heidelberg New York: Springer, 2013.

[49] L. R. Scott and S. Zhang. "Finite Element Interpolation of Nonsmooth Functions Satisfying Boundary Conditions". In: *Mathematics of Computation* 54.190 (1990), pp. 483–493. URL: http://www-jstor-org.uaccess.univie.ac.at/stable/2008497 (visited on 10/14/2025).

[50] D. Praetorius, M. Ruggeri, and B. Stiftner. "Convergence of an implicit–explicit midpoint scheme for computational micromagnetics". en. In: *Computers & Mathematics with Applications* 75.5 (Mar. 2018), pp. 1719–1738. URL: https://linkinghub.elsevier.com/retrieve/pii/S089812211730754X (visited on 02/17/2025).

[51] T. Hughes. *The finite element method: Linear static and dynamic finite element analysis*. Prentice-Hall, 1987. URL: https://books.google.at/books?id=pF-IQgAACAAJ.

[52] F. Alouges. "A new algorithm for computing liquid crystal stable configurations: The harmonic mapping case". In: *SIAM Journal on Numerical Analysis* 34.5 (1997), pp. 1708–1726. URL: https://doi.org/10.1137/S0036142994264249.

[53] M. Kruzík and A. Prohl. "Recent developments in the modeling, analysis, and numerics of ferromagnetism". In: *SIAM Review* 48.3 (2006), pp. 439–483. URL: https://doi.org/10.1137/S0036144504446187.

[54] J. Nocedal and S. J. Wright. *Numerical optimization*. en. 2nd ed. Springer series in operations research. New York: Springer, 2006.

[55] F. Jarre and J. Stoer. *Optimierung: Einführung in mathematische Theorie und Methoden*. ger. 2. Aufl. 2019. Masterclass. Berlin, Heidelberg: Springer Berlin Heidelberg, 2019.

[56] K. Deng, R. Wang, Z. Zhu, J. Zhang, and Z. Wen. *The Augmented Lagrangian Methods: Overview and Recent Advances*. arXiv:2510.16827 [math]. Oct. 2025. URL: http://arxiv.org/abs/2510.16827 (visited on 12/16/2025).

[57] J. F. Bonnans, ed. *Numerical optimization: theoretical and practical aspects*. en. 2. ed., [Nachdr.] Universitext. Berlin Heidelberg: Springer, 2010.

[58] M. R. Hestenes and E. Stiefel. "Methods of conjugate gradients for solving linear systems". In: *Journal of research of the National Bureau of Standards* 49 (1952), pp. 409–435. URL: https://api.semanticscholar.org/CorpusID:2207234.

[59] R. Fletcher and C. M. Reeves. "Function minimization by conjugate gradients". In: *The Computer Journal* 7.2 (Jan. 1964), pp. 149–154. URL: https://doi.org/10.1093/comjnl/7.2.149.

[60] G. H. Golub and C. F. Van Loan. *Matrix computations*. eng. 3rd ed. Johns Hopkins series in the mathematical sciences. Baltimore London: Johns Hopkins university press, 1996.

[61] D. G. Luenberger. *Introduction to linear and nonlinear programming*. eng. Reading, Mass.: Addison-Wesley, 1973.

[62] E. Polak and G. Ribiere. "Note sur la convergence de méthodes de directions conjuguées". fr. In: *Revue française d'informatique et de recherche opérationnelle. Série rouge* 3.16 (1969), pp. 35–43. URL: http://www.esaim-m2an.org/10.1051/m2an/196903R100351 (visited on 10/08/2025).

[63] B. Polyak. "The conjugate gradient method in extreme problem". In: *USSR Computational Mathematics and Mathematical Physics* 9 (Dec. 1969), pp. 94–112.

[64] L. Armijo. "Minimization of functions having Lipschitz continuous first partial derivatives". en. In: *Pacific Journal of Mathematics* 16.1 (Jan. 1966), pp. 1–3. URL: http://msp.org/pjm/1966/16-1/p01.xhtml (visited on 10/10/2025).

[65] Z. Salleh and A. Alhawarat. "An efficient modification of the Hestenes-Stiefel nonlinear conjugate gradient method with restart property". In: *Journal of Inequalities and Applications* 2016.1 (Apr. 2016), p. 110. URL: https://doi.org/10.1186/s13660-016-1049-5.

[66] P. E. Gill, W. Murray, and M. H. Wright. *Practical optimization*. eng. Repr. Bingley: Emerald, 2008.

[67] J. Schöberl. *Netgen Mesh Generator*. URL: https://sourceforge.net/projects/netgen-mesher/.

[68] J. Schöberl. *NGSolve Finite Element Library*. URL: https://sourceforge.net/projects/ngsolve/.

[69] Q. Chen and A. Konrad. "A review of finite element open boundary techniques for static and quasi-static electromagnetic field problems". eng. In: *IEEE transactions on magnetics* 33.1 (1997), pp. 663–676.

[70] Y. Saad. *Iterative methods for sparse linear systems*. eng. 2. ed., [Nachdr.] Philadelphia, PA: SIAM, Society for Industrial and Applied Mathematics, 2007.

[71] C. Garcia-Cervera and A. Roma. "Adaptive Mesh Refinement for Micromagnetics Simulations". en. In: *IEEE Transactions on Magnetics* 42.6 (June 2006), pp. 1648–1654. URL: http://ieeexplore.ieee.org/document/1634474/ (visited on 01/23/2025).

[72] *Netgen/NGSolve — Netgen/NGSolve documentation*. URL: https://ngsolve.org/index.html (visited on 03/09/2026).

[73] J. Schöberl. "NETGEN An advancing front 2D/3D-mesh generator based on abstract rules". In: *Computing and Visualization in Science* 1.1 (July 1997), pp. 41–52. URL: https://doi.org/10.1007/s007910050004.

[74] J. Schöberl. "C++ 11 implementation of finite elements in NGSolve". In: *Institute for analysis and scientific computing, Vienna University of Technology* 30 (2014).

[75] *Open CASCADE Technology — Collaborative development portal*. URL: https://dev.opencascade.org/ (visited on 03/09/2026).

# List of Figures

# List of Tables

# List of Algorithms