NUMERICAL METHODS FOR DYNAMIC MICROMAGNETICS

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Engineering and Physical Sciences

2015

By
David Shepherd
School of Computer Science

Contents

Abstract **16**

Declaration **17**

Copyright **18**

Acknowledgements **19**

1 Introduction **20**

1.1 Contents of thesis 22

2 Continuous Micromagnetics **24**

2.1 Definitions 24

2.2 The energy of a magnetic body 25

2.3 Dynamic Micromagnetics and the Landau-Lifshitz-Gilbert equation 29

2.4 Non-dimensionalisation 33

2.5 Geometric properties of the continuous Landau-Lifshitz-Gilbert equation 37

3 Numerical Methods for Dynamic Micromagnetic Modelling **40**

3.1 Introduction 40

3.2 Spatial Discretisation 43

3.3 Magnetostatic Field Calculations 45

3.4 Time Integration 49
3.5 Linearisation 68
3.6 Solution of sparse linear systems 71
3.7 Compatibility of methods 77
3.8 Conclusions 78

4 A finite element method for the Landau-Lifshitz-Gilbert equation 79

4.1 Introduction to the finite element method 79
4.2 The FEM applied to the LLG equation 88
4.3 Time Discretisation 94
4.4 The Newton-Raphson Jacobian 96
4.5 Geometric integration with the FEM 103

5 Hybrid finite/boundary element method 112

5.1 The continuous FEM/BEM formulation 113
5.2 Discretisation 119
5.3 Evaluation of the discrete boundary operator 121
5.4 Conclusions 125

6 Efficient solution of the linear systems 126

6.1 Solution of decoupled systems 127
6.2 Solution of the coupled system 128
6.3 Numerical experiments 136
6.4 Outlook 142
6.5 Conclusions and future work 145

7 An adaptive implicit midpoint rule scheme **148**

7.1 The implicit midpoint rule with fixed step size 148

7.2 Adaptive implicit midpoint rule 149

7.3 Numerical experiments with simple ODEs 156

7.4 Numerical experiments with the ODE LLG 165

7.5 Conclusions 174

8 Validation, convergence and conservation experiments **177**

8.1 Example with a wave-like solution 178

8.2 An example with non-uniform applied field 187

8.3 The μmag standard problem #4 191

8.4 Conclusions 204

9 Discretisation induced stiffness in the Landau-Lifshitz-Gilbert equation **206**

9.1 Problem definition 207

9.2 Implementation details 208

9.3 Numerical results 210

9.4 Conclusions 213

10 Conclusions and future work **214**

10.1 Conclusions 214

10.2 Future work 215

A Analytical solutions of the Landau-Lifshitz-Gilbert equation **217**

A.1 Ellipsoidial nano-particle 217

A.2 Wave-like solution in an infinite domain 218

B Properties of the effective field operator **222**

B.1 Linear Symmetrical operator 222

B.2 Effective field and energy 226

C Truncation error derivation for the implicit midpoint rule 227

D Alternative magnetisation re-normalisation methods 229

D.1 Implementation details 229

D.2 Results: tolerance-based renormalisation 232

D.3 Results: the self-correcting LLG 233

D.4 Conclusions and future work 238

Word Count: 48,693

List of Figures

2.1 The domain labels used: Ω is the magnetic material, Γ is the boundary and Ω^c is the (infinite) external region. 25

2.2 The effects of the precession and damping terms in the Landau-Lifshitz and Landau-Lifshitz-Gilbert equations on the magnetisation direction, $\mathbf{M}$, of a single magnetic moment with a constant effective field. 31

3.1 Commutation relationship between the IMR discretisation and the LLG energy derivation. In contrast to other time integration methods the result is independent of the order of operations. 67

4.1 The local and global piecewise polynomial basis functions and numbering schemes for a one-dimensional finite element method. 85

5.1 A 2D representation of the geometry showing the labels used in this chapter. The point $\mathbf{x}'$ is a singular point of the boundary Γ, the angle $\alpha(\mathbf{x}')$ is as shown. 114

6.1 The test problem used for the linear solvers in the state at time $t = 0$. The arrows represent the magnetisation direction at each node of the finite element mesh. Colour represents the magnitude of m_x. The semi-transparent pale green grid shows the finite element mesh. 137

6.2 Mean (over a single Newton solve) of the GMRES iterations to converge against problem size for the decoupled LLG preconditioned by ILU(1). The legend indicates the time step size. Data points for all values of α and $\mathcal{K}_1$; both types of quadrature; and both formats of the BEM block are shown but are not distinguished. . 139

6.3 Mean (over a single Newton solve) of the time in seconds to set up an ILU(1) preconditioner against problem size for the decoupled LLG block. The legend indicates the time step size. Data points for all values of α and $\mathcal{K}_1$; both types of quadrature; and both formats of the BEM block are shown but are not distinguished. . 140

6.4 Mean (over a single Newton solve) of the GMRES iterations to converge against problem size for the monolithic system preconditioned by $\mathcal{P}_1$ (inverted by LU decomposition). The legend indicates the time step size. Data points for all values of α and $\mathcal{K}_1$; both types of quadrature; and both formats of the BEM block are shown but are not distinguished. Some data points for the largest N are missing due the LU factors requiring more than 16GB of memory. 141

6.5 Mean (over a single Newton solve) of the time in seconds to set up $\mathcal{P}_1$ (inverted by LU decomposition) against problem size for the monolithic system. The legend indicates the time step size. Data points for all values of α and $\mathcal{K}_1$; both types of quadrature; and both formats of the BEM block are shown but are not distinguished. Some data points for the largest N are missing due the LU factors requiring more than 16GB of memory. 142

6.6 Mean (over a single Newton solve) of the GMRES iterations to converge against problem size for the monolithic system using the partially-inexact preconditioners $\bar{\mathcal{P}}_2$, $\bar{\mathcal{P}}_3$ with $\Delta_n = 0.1$. The legend indicates the preconditioner and quadrature scheme used. Data points for all values of α, $\mathcal{K}_1$, and for both formats of the BEM block are shown but are not distinguished. 143

6.7 Mean (over a single Newton solve) of the time in seconds to set up the partially-inexact preconditioners $\bar{\mathcal{P}}_2$, $\bar{\mathcal{P}}_3$ against problem size for the monolithic system with $\Delta_n = 0.1$. The legend indicates the preconditioner and quadrature scheme used. Data points for all values of α, $\mathcal{K}_1$, and for both formats of the BEM block are shown but are not distinguished. 144

6.8 Mean (over a single Newton solve) of the GMRES iterations to converge against problem size for the monolithic system using the inexact preconditioners $\widetilde{\mathcal{P}_2}$, $\widetilde{\mathcal{P}_3}$ with $\Delta_n = 0.1$. The legend indicates the preconditioner and quadrature scheme used. Data points for all values of α, $\mathcal{K}_1$, and for both formats of the BEM block are shown but are not distinguished. Some data points are missing for the largest N due to a lack of convergence. 145

6.9 Mean (over a single Newton solve) of the time in seconds to set up the inexact preconditioners $\widetilde{\mathcal{P}_2}$, $\widetilde{\mathcal{P}_3}$ against problem size for the monolithic system with $\Delta_n = 0.1$. The legend indicates the preconditioner and quadrature scheme used. Data points for all values of α, $\mathcal{K}_1$, and for both formats of the BEM block are shown but are not distinguished. Some data points are missing for the largest N due to a lack of convergence. 146

7.1 Absolute error, time step size and computed solutions for the example ODE with exact solution $y(t) = t^2 + 0.5$. 159

7.2 Absolute error, time step size and computed solutions for the oscillatory, damped example ODE with exact solution $y(t) = e^{-\beta t}\sin(\omega t)$. 160

7.3 Behaviour of the global error norm with varying tolerance for the oscillatory, damped example ODE with exact solution $y(t) = e^{-\beta t}\sin(\omega t)$. 161

7.4 Comparison of the `oomph-lib` and VODE implementations of adaptive BDF2 for the oscillatory, damped example ODE with exact solution $y(t) = e^{-\beta t}\sin(\omega t)$. Absolute error, time step size and computed solutions against time are shown. 162

7.5 Absolute error, step size and computed solutions for the stiff example ODE. 163

7.6 Absolute error, step size and computed solutions for the stiff order reduction example ODE given in (3.30). 163

7.7 Behaviour of the global error norm with varying tolerance vs mean step size for the order reduction example. 164

7.8 Plot of $\mathbf{m}$ and Δ_n over time for the relaxing nano-sphere problem solved by adaptive IMR. 169

7.9 Plot of $\mathbf{m}$ and Δ_n over time for the relaxing nano-sphere problem solved by adaptive BDF2. 170

7.10 Plot of $\mathbf{m}$ and Δ_n over time for the relaxing nano-sphere problem solved by adaptive TR. 170

7.11 Plot of magnetisation length errors, $\mathcal{E}_{|\mathbf{m}|} = ||\mathbf{m}| - 1|$, over time for the relaxing nano-sphere problem solved with each of the three adaptive integrators (without re-normalisation of $\mathbf{m}$). 171

7.12 Plot of maximum magnetisation length error over time, $\max_n ||\mathbf{m}_n| - 1|$, against the mean converged Newton residual norm for the relaxing nano-sphere problem with a wide range of α, ϵ_{N} and ϵ_{Δ} parameters. 172

7.13 Convergence plot of the maximum error in the energy for the relaxing nano-sphere problem with $\alpha = 0$ solved using BDF2 with and without re-normalisation of $\mathbf{m}$. The energy error when using IMR is too small to be calculated. The digit 1 or 0 in the legend indicates re-normalisation or no re-normalisation respectively. . . 173

7.14 Convergence plot of the switching time error, (7.33), against average time step for each method. The digit 1 or 0 in the legend indicates re-normalisation or no re-normalisation respectively. . . 174

7.15 Convergence plots of the error in magnetisation, (7.37), against average time step for each method. The digit 1 or 0 indicates re-normalisation or no re-normalisation respectively. 175

8.1 Snapshot of the approximate solution for the 2D wave problem at $t = 0.1$ time units obtained using adaptive IMR with nodal integration. Arrows indicate the nodal values of $\mathbf{m}$, colour indicates the value of m_x, and the pale green area is the finite element mesh used. The z-component of the magnetisation is constant over the domain with $m_z = 0.958$. 180

8.2 The temporal behaviour of the wave solution at $\mathbf{x} = \mathbf{0}$ and the time step selected by the various adaptive integration schemes without re-normalisation. 180

8.3 Convergence in the error norm $\mathcal{E}_{\mathbf{m}}$ after a single step of time integration for the 2D wave-like problem. 182

8.4 Convergence of $\int_T \mathcal{E}_{\mathbf{m}} \, \mathrm{d}t$, where $T = [0, 5]$, for the 2D wave-like problem. 183

8.5 Convergence of $\int_T \mathcal{E}_{m_z} \, \mathrm{d}t$, where $T = [0, 5]$, for the 2D wave-like problem. 184

8.6 Evolution of $\mathcal{E}_{|\mathbf{m}|}$ with various time integrators and quadratures for the 2D wave-like problem. 184

8.7 Evolution of the error in energy with various time integration schemes and quadratures for the undamped 2D wave-like problem. 185

8.8 Correlation between maximum (over all nodes and all time steps) error of nodal magnetisation lengths and the mean (over time) of the converged Newton residual norm in the 2D wave-like problem solved using adaptive IMR and nodal quadrature. 186

8.9 Correlation between the maximum (over time) of the error in energy and the mean (over time) of the converged Newton residual norm in the undamped 2D wave-like problem solved using adaptive IMR and nodal quadrature. 186

8.10 Evolution of $\mathcal{E}_{|\mathbf{m}|}$ with various time integration schemes and quadratures for the 2D nonuniform field problem. 189

8.11 Evolution of the error in energy with various time integration schemes and quadratures for the undamped 2D nonuniform field problem. 189

8.12 Evolution of $\mathcal{E}_{|\mathbf{m}|}$ with IMR, both quadratures, and both element shapes for the 2D nonuniform field problem. 190

8.13 Evolution of the error in energy with IMR, both quadratures, and both element shapes for the undamped 2D nonuniform field problem. 191

8.14 Initial S-state as generated by adaptive IMR with nodal quadrature and $n_x = 75$ nodes along the x direction. Colour indicates the z (out of plane) component of magnetisation. 194

8.15 Evolution of the mean magnetisation for the µmag problem #4 with field 1 using monolithic IMR with nodal quadrature. For comparison we also show the results submitted to the µmag website by d'Aquino *et al.* [73]. 195

8.16 Evolution of the mean magnetisation for the µmag problem #4 with field 2 using monolithic IMR with nodal quadrature. For comparison we also show the results submitted to the µmag website by d'Aquino *et al.* [73]. 196

8.17 Evolution of the mean magnetisation for the µmag problem #4 with field 2 using IMR with nodal quadrature and monolithic (implicit) or semi-implicit (decoupled) coupling to the magnetostatics problem. 197

8.18 Evolution of the mean magnetisation for the µmag problem #4 with field 2 using various time integration schemes, Gaussian quadrature, monolithic coupling, and $n_x = 100$. 197

8.19 Evolution of $\mathcal{E}_{|\mathbf{m}|}$ with IMR, nodal quadrature and both coupling strategies for the µmag problem #4 with field 2. 198

8.20 Evolution of the error in energy with various time integration schemes, quadratures, and coupling strategies for the undamped µmag problem #4 with field 2. Results for other combinations of methods are very similar. 199

8.21 Mean y component of the magnetisation for the µmag problem 4 with field 2 solved using IMR, monolithic coupling, both quadrature schemes and varying spatial resolution n_x. The number in the legend indicates $n_x/5$. 200

8.22 GMRES iterations to converge against problem size for the monolithic system using the inexact preconditioner $\widetilde{\mathcal{P}_3}$ with the various time integration schemes, quadratures, and applied fields for the µmag problem #4 (including the calculation of the relaxed initial condition). 202

8.23 Time in seconds to set up the inexact preconditioner $\widetilde{\mathcal{P}_3}$ against problem size for the monolithic system using the inexact preconditioner $\widetilde{\mathcal{P}_3}$ with the various time integration schemes, quadratures, and applied fields for the µmag problem #4 (including the calculation of the relaxed initial condition). 203

8.24 Newton-Raphson iterations to converge against problem size for the monolithic system with the various time integration schemes, quadratures, and applied fields for the µmag problem #4 (including the calculation of the relaxed initial condition). 204

9.1 Maximum stable time step as a function of discretisation size for LLG without magnetostatics. Data points are $\Delta_{\max}$, the largest stable time step found. Error bars represent the range in which the largest stable time step is contained. The horizontal dashed line shows the time step where RK2 and IMR are equally efficient (IMR is more efficient when the maximum stable step of the RK2 method moves below this line). The maximum time step is limited to 0.1 for accuracy reasons. 210

9.2 Stable time step against discretisation size for LLG with FEM/BEM magnetostatics. 211

9.3 Stable time step against discretisation size for LLG with $\mathbf{h}_{\text{ms}} = -\mathbf{m}/3$. 211

A.1 Magnetisation values over time at $x = 0$ for the one dimensional wave solution with $k = 2\pi$, $\alpha = 0.05$ and various c (taken well away from $\frac{\pi}{2}$). 220

A.2 One dimensional wave solution values over time at $x = 0$ with $k = 2\pi$, $\alpha = 0.05$ and c approaching the critical value $\frac{\pi}{2}$. 221

D.1 Error in magnetisation length, $\mathcal{E}_{|\mathbf{m}|}$, over time for the relaxing nano-sphere problem with $\alpha = 0$. Magnetisation length is enforced by tolerance-based renormalisation, the legend indicates the value of the tolerance ϵ_{ml}. 231

D.2 Error in magnetisation length, $\mathcal{E}_{|\mathbf{m}|}$, over time for the relaxing nano-sphere problem with $\alpha = 0.01$. Magnetisation length is enforced by tolerance-based renormalisation, the legend indicates the value of the tolerance ϵ_{ml}. 231

D.3 Error in energy over time for the relaxing nano-sphere problem with $\alpha = 0$. Magnetisation length is enforced by tolerance-based renormalisation, the legend indicates the value of the tolerance ϵ_{ml}. 232

D.4 Convergence of the maximum (over all steps) of the error in the switching time against step size for the relaxing nano-sphere problem with $\alpha = 0.01$. Magnetisation length is enforced by tolerance-based renormalisation, the legend indicates the value of the tolerance ϵ_{ml}. 233

D.5 Error in magnetisation length, $\mathcal{E}_{|\mathbf{m}|}$, over time for the relaxing nano-sphere problem with $\alpha = 0$. Magnetisation length is enforced by use of the self-correcting LLG. The legend indicates the values of β. 234

D.6 Error in magnetisation length, $\mathcal{E}_{|\mathbf{m}|}$, over time for the relaxing nano-sphere problem with $\alpha = 0.01$. Magnetisation length is enforced by use of the self-correcting LLG, the legend indicates the values of β. 234

D.7 Error in energy over time for the relaxing nano-sphere problem with $\alpha = 0$. Magnetisation length is enforced by use of the self-correcting LLG, the legend indicates the values of β. 235

D.8 Convergence of the maximum (over all steps) of the error in the switching time against step size for the relaxing nano-sphere problem with $\alpha = 0.01$. Magnetisation length is enforced by use of the self-correcting LLG or re-normalisation after every step (*i.e.* tolerance based renormalisation with $\epsilon_{\text{ml}} = 0$). The legend indicates, respectively, the values of β and the tolerance ϵ_{ml}. 236

D.9 Convergence of the maximum (over all steps) of the error in the switching time against step size for the relaxing nano-sphere problem with $\alpha = 0.01$ solved using adaptive BDF2. Magnetisation length is enforced by use of the self-correcting LLG or re-normalisation after every step (*i.e.* tolerance based renormalisation with $\epsilon_{\mathrm{ml}} = 0$). The legend indicates, respectively, the values of β and the tolerance ϵ_{ml}. 237

Abstract

Micromagnetics is a continuum mechanics theory of magnetic materials widely used in industry and academia. In this thesis we describe a complete numerical method, with a number of novel components, for the computational solution of dynamic micromagnetic problems by solving the Landau-Lifshitz-Gilbert (LLG) equation. In particular we focus on the use of the implicit midpoint rule (IMR), a time integration scheme which conserves several important properties of the LLG equation. We use the finite element method for spatial discretisation, and use nodal quadrature schemes to retain the conservation properties of IMR despite the weak-form approach.

We introduce a novel, generally-applicable adaptive time step selection algorithm for the IMR. The resulting scheme selects error-appropriate time steps for a variety of problems, including the semi-discretised LLG equation. We also show that it retains the conservation properties of the fixed step IMR for the LLG equation.

We demonstrate how hybrid FEM/BEM magnetostatic calculations can be coupled to the LLG equation in a monolithic manner. This allows the coupled solver to maintain all properties of the standard time integration scheme, in particular stability properties and the energy conservation property of IMR. We also develop a preconditioned Krylov solver for the coupled system which can efficiently solve the monolithic system provided that an effective preconditioner for the LLG sub-problem is available.

Finally we investigate the effect of the spatial discretisation on the comparative effectiveness of implicit and explicit time integration schemes (*i.e.* the stiffness). We find that explicit methods are more efficient for simple problems, but for the fine spatial discretisations required in a number of more complex cases implicit schemes become orders of magnitude more efficient.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

Firstly I would like to thank my three supervisors: Milan Mihajlović, Matthias Heil and Jim Miles (in no particular order) for their encouragement, advice and endless assistance. I would also like to thank Milan in particular for his tireless reading and rereading of various drafts of this thesis.

Secondly I would like to thank Raymond White for his assistance, good company and for many interesting discussions over the course of the last three and a half years. I would also like to thank my wonderful girlfriend Sylwia for all of her support and encouragement during the process of writing this thesis.

Finally I would like to thank everyone in the NEST group, the `oomph-lib` group and the NoWNano DTC for providing a supportive and stimulating environment to work in.

Chapter 1

Introduction

In this thesis we study numerical techniques for the modelling of ferromagnetic materials. In ferromagnetic materials, referred to in this thesis as simply "magnetic materials", exchange coupling between nearby spins causes the emergence of spontaneous magnetisation. Some magnetic materials have preferred magnetisation directions (anisotropy), for example due to the crystal lattice structure. In such materials systems with two stable states can be obtained. The system can be switched between these states by the application of a magnetic field, which can be easily generated using pulses of electrical current. These states can be extremely stable, in some cases having average lifetimes of decades at room temperature even for extremely small volumes of material. Such systems have many technological applications, particularly in the area of data storage.

In current hard disk drives collections of nanometre-sized grains of anisotropic magnetic material are magnetised using a write head (an electromagnet) in order to store bits of data [63, Sec. 14.6]. The data is read back using magnetoresistive sensors, which change their resistance based on the magnetic field. Such sensors can be extremely sensitive and extremely small, allowing them be very close to the bits and to obtain good signal-to-noise ratios despite the small size of the bits.

Another application area where ferromagnetic materials show great promise is in random access memory (RAM), used as short term data storage for computers. The current standard RAM technology (dynamic RAM or DRAM) is based on the charge stored in capacitors. These capacitors leak charge fairly rapidly, resulting

in data lifetimes on the order of milliseconds. Hence they must be continually refreshed and can require significant power usage to retain data. A widely proposed alternative is MRAM (magnetoresistive RAM) which uses arrays of nano-sized regions of magnetic material to store data and in-place magnetoresistive sensors to allow random-access reads [63, Sec. 14.4]. The vastly longer data lifetimes allowed by the use of magnetic materials results in greatly reduced standby power usage by removing the requirement for continual refreshing. Moreover, computers could be easily put into a completely powered-off hibernation state without writing data to disk. Historically there have been issues with random-access writes in MRAM: moving write heads (as used in hard drives) cannot be used because the time to move the head to the correct location is impractically long, and in-place generation of the required magnetic fields requires large currents. However recent developments in spin-torque-transfer, where spin-polarised currents are used to directly apply a torque to the magnetisation of a ferromagnet, offer a solution to these difficulties [5]. Spin torque transfer MRAM has recently been commercialised by Everspin Technologies [49] and is beginning to make its way into products. Due to the combination of non-volatility, high density, random-access, and low read/write times it is a candidate for a so-called "universal memory", where all data is stored a RAM-like memory eliminating long access times.

It is extremely desirable to be able to accurately model the behaviour of magnetic materials for both research and device design purposes. However the use of models based on quantum mechanics (such as density functional theory) to predict the behaviour of magnetic systems is extremely computationally expensive due to the length scales required. Instead a theory known as *micromagnetics* is typically employed [1]. In micromagnetics a continuum approximation of the ferromagnet's magnetisation is used – it is assumed that the magnetisation can be modelled as a continuous function of space and time (*i.e.* the contribution due to individual atoms is averaged out). The continuum approximation is good if the length scale over which the magnetisation varies is significantly larger than the inter-atomic distance. This is usually the case in ferromagnetic materials because of the smoothing effect of ferromagnetic exchange. Additionally a semi-classical approach is used: the various important quantum mechanical effects are approximated by classical representations. The result of these approximations is a powerful framework for theoretical and computational predictions of the behaviour of magnetic materials.

Micromagnetic models can be broadly split into two categories: energy based models and dynamic models. Energy based models aim to find stable minimum energy states for the system whereas dynamic models simulate the evolution of the magnetisation over time. Dynamic models may be favourable even when there are no time-dependent effects because standard energy based models are unable to distinguish which of the various stable states will be the final state of the system. Dynamic micromagnetic models require the solution of a differential equation, known as the Landau-Lifshitz-Gilbert equation (LLG). Depending on the physical system modelled the LLG may be an ordinary, partial or stochastic differential equation. The very closely related Landau–Lifshitz–Slonczewski equation can be used to model magnetisation dynamics when spin-torque-transfer effects are included [92].

In this thesis we study numerical methods with the final goal of finding more reliable and computationally efficient methods for dynamic micromagnetics simulations. In particular we focus on methods which improve the efficiency of so-called "geometric integration" schemes. Geometric integration schemes are numerical methods which are able to retain important *qualitative* properties of a differential equation in an approximate solution, such as conservation of energy in non-dissipative systems. In other areas of computational physics[1] geometric integration methods have been shown to greatly reduce the overall accumulation of numerical errors [54, p. 77]. This allows more accurate results at the same computational cost (or a reduction in the computational cost to obtain the same accuracy). In particular we focus mainly on a widely known time integration scheme with geometric integration properties when applied to dynamic micromagnetics calculations: the implicit midpoint rule.

1.1 Contents of thesis

The first two chapters, Chapters 2 and 3, comprise a basic introduction to micromagnetic models and the numerical methods commonly used in this context. In particular Section 3.4 contains a detailed description of the properties of a selection of time integration schemes. Chapter 4 gives a more detailed introduction

[1]Some examples are: the field of celestial mechanics [43] and the solution of highly oscillatory differential equations such as the Airy equation [54, p. 98].

to the finite element method, including a description of how it can be applied to dynamic micromagnetics simulations. The chapter ends with a discussion of how the basic finite element method can be extended to retain the geometric integration properties of the implicit midpoint rule. Chapter 5 introduces the hybrid FEM/BEM method, a widely used technique for the accurate calculation of magnetostatic fields.

The later chapters contain the main novel research contributions of this thesis. Chapter 6 describes efficient iterative techniques for the solution of the coupled linear and non-linear systems resulting from the use of FEM/BEM magnetostatics calculations in an LLG-based dynamic model. In particular we introduce techniques which reduce the development of an efficient solver for a monolithically coupled problem to the development of an effective preconditioner for the LLG sub-problem alone. Such a coupling strategy is required to retain the geometric integration properties of the implicit midpoint rule, and may also be useful in the time integration of the stochastic LLG.

In Chapter 7 we introduce a novel adaptive time step selection algorithm for the implicit midpoint rule. This algorithm is not specific to the LLG, and to our knowledge is the first such algorithm. The same chapter also contains numerous numerical experiments demonstrating the selection of appropriate time steps for a number of ordinary differential equations (ODEs), and demonstrating the geometric integration properties when applied to an ODE form of the LLG.

In Chapter 8 we present a number of numerical experiments using combinations of the methods developed in this thesis. The methods are validated against a number of examples: a wave-like problem with an analytical solution, relaxation under a non-uniform field, and the µmag standard problem #4 [73]. Additionally the convergence and geometric integration properties of a variety of time integration schemes are compared.

Finally, in Chapter 9 the comparative efficiency of two classes of time integration schemes (implicit and explicit) are compared for an example problem across a range of spatial discretisations.

Appendix A contains a description of two analytical solutions to the LLG equation which have proven very useful in the verification of our implementation. The other appendices contain technical details of some derivations.

Chapter 2

Continuous Micromagnetics

In this chapter we give an introduction to the theory of micromagnetics including a statement of the equations to be solved. Micromagnetics is a semi-classical (*i.e.* quantum mechanical effects are approximated using classical techniques), continuum (*i.e.* the fields involved are assumed to be "smooth") theory of ferromagnetism. It is widely used to describe the behaviour of ferromagnetic materials on scales above those accessible using quantum mechanical approaches such as density functional theory.

2.1 Definitions

We define $\mathbf{M}(\mathbf{x}, t)$ to be a vector field representing the expectation value of the magnetisation per unit volume averaged over a number of unit cells [1]. The length of $\mathbf{M}(\mathbf{x}, t)$ is fixed and given by the saturation magnetisation M_s, a material parameter. We define $\mathbf{H}(\mathbf{x}, t)$ to be the magnetic field. These quantities are related to the magnetic flux density $\mathbf{B}(\mathbf{x}, t)$ (in units of Tesla) by

$$\mathbf{B}(\mathbf{x}, t) = \mu_0 \left(\mathbf{H}(\mathbf{x}, t) + \mathbf{M}(\mathbf{x}, t) \right), \tag{2.1}$$

where μ_0 is the magnetic constant (or permeability of free space).

Our labelling of the domains is shown in Figure 2.1. We call the region of magnetic material Ω, it's boundary Γ and the external domain Ω^c. When working with equations in dimensional form we always use S.I. units, although almost

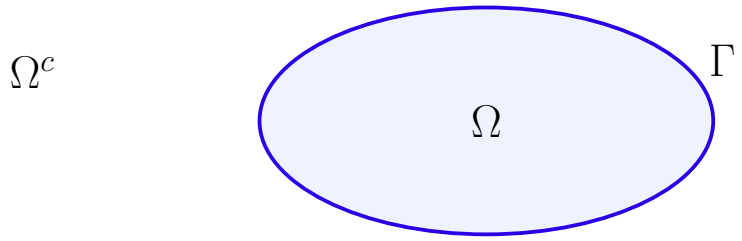

Figure 2.1: The domain labels used: Ω is the magnetic material, Γ is the boundary and Ω^c is the (infinite) external region.

all equations are written in non-dimensional form after this form is derived in Section 2.4. In general we will not make dependencies on $\mathbf{x}$ and t explicit.

For a scalar function f and a vector function $\mathbf{g}$ in three dimensions we use the standard definitions of the gradient, divergence and Laplacian operators: $\nabla f = (\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z})$, $\nabla \cdot \mathbf{g} = \frac{\partial g_x}{\partial x} + \frac{\partial g_y}{\partial y} + \frac{\partial g_z}{\partial z}$, and $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2}$ respectively. We also write the vector Laplacian as $\nabla^2 \mathbf{g} = (\nabla^2 g_x, \nabla^2 g_y, \nabla^2 g_z)$. In less than three dimensions the obvious extensions are used.

We are often concerned with energies E that depend on the values of the function $\mathbf{M}$ over all space, *i.e.* $E[\mathbf{M}(\mathbf{x})]$ is a *functional* of $\mathbf{M}$. For such a functional we define its first variation as

$$\delta E[\mathbf{M}] = E[\mathbf{M} + \delta\mathbf{M}] - E[\mathbf{M}], \tag{2.2}$$

for some "small" arbitrary function $\delta\mathbf{M}$. The functional (or variational) derivative, $\frac{\delta E}{\delta \mathbf{M}}$, is defined as the functional such that

$$\delta E = \int_\Omega \frac{\delta E}{\delta \mathbf{M}} \cdot \delta\mathbf{M} \, \mathrm{d}\Omega. \tag{2.3}$$

2.2 The energy of a magnetic body

A number of energy terms are included in standard micromagnetic models: the magnetostatic self energy, the exchange energy, the magnetocrystalline anisotropy energy and the energy due to external applied fields (created by magnetic bodies or electrical currents outside the modelled region). We begin by providing a description of each of these energies. Other effects which are sometimes important include temperature (thermal effects) and magnetostriction (the change in

magnetic behaviour due to stretching or compressing the magnet).

The energy due to an external applied field, $\mathbf{H}_{\text{ap}}$, (sometimes called the Zeeman energy) is

$$E_{\text{ap}} = -\mu_0 \int_\Omega \mathbf{H}_{\text{ap}} \cdot \mathbf{M} \, \mathrm{d}\Omega. \tag{2.4}$$

The first variation of this energy with respect to $\mathbf{M}$ is

$$\begin{aligned} \delta E_{\text{ap}} &= -\mu_0 \left[\int_\Omega \mathbf{H}_{\text{ap}} \cdot \mathbf{M} + \mathbf{H}_{\text{ap}} \cdot \delta\mathbf{M} \, \mathrm{d}\Omega - \int_\Omega \mathbf{H}_{\text{ap}} \cdot \mathbf{M} \, \mathrm{d}\Omega \right], \\ &= -\mu_0 \int_\Omega \mathbf{H}_{\text{ap}} \cdot \delta\mathbf{M} \, \mathrm{d}\Omega, \end{aligned} \tag{2.5}$$

and so by (2.3)

$$\mathbf{H}_{\text{ap}} = \frac{-1}{\mu_0} \frac{\delta E_{\text{ap}}}{\delta \mathbf{M}}. \tag{2.6}$$

Motivated by (2.6) we define an "effective field" $\mathbf{H}$ for *any* energy term E:

$$\mathbf{H} = \frac{-1}{\mu_0} \frac{\delta E}{\delta \mathbf{M}}. \tag{2.7}$$

This effective field acts exactly like a real field in calculations of the magnetisation dynamics and represents the effects of the corresponding energy term.

2.2.1 Exchange energy

In a ferromagnetic material neighbouring magnetic moments prefer to align parallel to each other due to quantum mechanical effects. This can be approximated by a classical expression for exchange energy [1] as

$$E_{\text{ex}} = \frac{A}{M_s^2} \int_\Omega (\nabla M_x)^2 + (\nabla M_y)^2 + (\nabla M_z)^2 \, \mathrm{d}\Omega, \tag{2.8}$$

where A is the material dependent exchange constant representing the strength of the exchange coupling.

It can be shown [1][18, p. 46][1] that

$$\begin{aligned}\delta E_{\text{ex}} &= \frac{2A}{M_s^2}\int_\Omega (\nabla M_x)\cdot\delta(\nabla M_x) + (\nabla M_y)\cdot\delta(\nabla M_y) + (\nabla M_z)\cdot\delta(\nabla M_z)\,\mathrm{d}\Omega,\\ &= \frac{-2A}{M_s^2}\int_\Omega (\nabla^2\mathbf{M})\cdot\delta\mathbf{M}\,\mathrm{d}\Omega.\end{aligned} \tag{2.9}$$

Hence the exchange effective field is given by

$$\mathbf{H}_{\text{ex}} = \frac{2A}{\mu_0 M_s}\nabla^2\mathbf{M}. \tag{2.10}$$

2.2.2 Magnetostatic energy

Magnetic materials generate a magnetic field, which in turn affects the magnetic material emitting the field. This field is also known as the demagnetising field as it tends to oppose the magnetisation of the body. When there are no changing electric fields the magnetic field can be calculated using only two equations rather than the full four Maxwell's equations, this case is known as magnetostatics. The first equation needed is Ampere's law [23, p. 33]

$$\nabla\times\mathbf{H} = \hat{\mathbf{j}}_c, \tag{2.11}$$

where $\hat{\mathbf{j}}_c$ is the "free current": the conduction current in electrical circuits (*i.e.* not including the current due to electron spin). The second equation needed is [23, p. 34]

$$\nabla\cdot\mathbf{H} = -\nabla\cdot\mathbf{M}. \tag{2.12}$$

These two equations can be derived from Maxwell's equations by setting $\frac{\partial\mathbf{E}}{\partial t} = 0$. We call the resulting field the magnetostatic field and label it $\mathbf{H}_{\text{ms}}$. The solutions to these equations are typically calculated in one of two ways: by using an integral method or by solving a simplified differential equation in terms of a potential.

The integral form of the magnetostatic field at a point $\mathbf{x}\in\mathbb{R}^3$ due to the magnetic body Ω with boundary Γ can be given in terms of magnetisation $\mathbf{M}$ by an integral over all volume and surface "magnetic charges" ($\nabla\cdot\mathbf{M}$ and $\mathbf{M}\cdot\hat{\mathbf{n}}$ respectively)

[1] Briefly: use the fact that $\delta\nabla\mathbf{M} = \nabla\delta\mathbf{M}$ and apply (B.3) (assuming that the boundary condition (2.24) holds) to shift the both derivatives onto the $\mathbf{M}$ term.

as

$$\mathbf{H}_{\text{ms}}(\mathbf{x}) = \frac{1}{4\pi}\left[-\int_\Omega \nabla' \cdot \mathbf{M}(\mathbf{x}')\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}\,\mathrm{d}^3\mathbf{x}' + \int_\Gamma \mathbf{M}(\mathbf{x}')\cdot\hat{\mathbf{n}}(\mathbf{x}')\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}\,\mathrm{d}^2\mathbf{x}'\right], \tag{2.13}$$

where ∇' denotes the grad operator with respect to the $\mathbf{x}'$ coordinate and $\hat{\mathbf{n}}(\mathbf{x})$ is the outer unit normal of Ω at $\mathbf{x}$. For completeness we note that single magnetic charges (magnetic monopoles) are a useful mathematical tool but have not been observed in nature.

When the magnetostatic field is produced only by magnets, and not by electric currents (*i.e.* $\hat{\mathbf{j}}_c = 0$), then it is possible to express the field in terms of a scalar potential, ϕ [23, p. 46][2]

$$\begin{aligned} \mathbf{H}_{\text{ms}} &= -\nabla\phi, \\ \nabla^2\phi &= \nabla\cdot\mathbf{M}. \end{aligned} \tag{2.14}$$

The boundary conditions on ϕ are

$$\begin{aligned} \phi^{\text{int}} - \phi^{\text{ext}} &= 0 \quad \mathbf{x}\in\Gamma, \\ \frac{\partial\phi^{\text{int}}}{\partial\hat{\mathbf{n}}} - \frac{\partial\phi^{\text{ext}}}{\partial\hat{\mathbf{n}}} &= \mathbf{M}\cdot\hat{\mathbf{n}} \quad \mathbf{x}\in\Gamma, \\ \phi &\to 0 \text{ as } |\mathbf{x}|\to\infty, \end{aligned} \tag{2.15}$$

where ϕ^{int} and ϕ^{ext} are respectively the values of ϕ just inside and outside the domain Ω. These conditions come from the requirement for continuity in the potential, the boundary conditions on the magnetic field ($\mathbf{B}^{\text{int}}\cdot\hat{\mathbf{n}} = \mathbf{B}^{\text{ext}}\cdot\hat{\mathbf{n}}$) and the requirement for finite total energy respectively. Note that (2.15) implies continuity but not smoothness of the magnetic potential ϕ across Γ. This can be thought of as the effect of (fictional) surface monopole charges.

The corresponding energy contribution is given by

$$E_{\text{ms}} = \frac{-\mu_0}{2}\int_\Omega \mathbf{H}_{\text{ms}}\cdot\mathbf{M}\,\mathrm{d}\Omega. \tag{2.16}$$

The factor of $\frac{1}{2}$ is to avoid double counting: the integral includes the contribution of the field generated by an infinitesimal region A on a region B, but also the effect of a field generated by B on A.

[2]This is because Ampere's law reduces to $\nabla\times\mathbf{H} = 0$, meaning that it is satisfied by $\mathbf{H} = \nabla\phi$ for any scalar ϕ.

2.2.3 Magnetocrystalline anisotropy energy

Many magnetic materials have a preferred direction (or directions) of magnetisation due to their crystal structure, this is known as magnetocrystalline anisotropy. The energy contribution due to this anisotropy depends on the crystal structure of the material, but is typically well approximated by one of a few simple algebraic functions of $\mathbf{M}$. Because of this the exact type of anisotropy is not especially important for the discussion of numerical methods. Here we focus on only the most technologically relevant case: in magnetic data recording interest is focused on materials with a single out of plane *easy axis* (perpendicular uniaxial anisotropy) as this gives the best stability for the stored data.

The first order approximation for the energy contribution of a perpendicular uniaxial anisotropy is given by

$$E_{\text{ca}} = -K_1 \int_\Omega \left(\frac{\mathbf{M}}{M_s} \cdot \hat{\mathbf{e}} \right)^2 \, \mathrm{d}\Omega, \tag{2.17}$$

where $\hat{\mathbf{e}}$ is the easy axis and K_1 is the material dependent first order anisotropy energy density coefficient.[3] The first order approximation is valid for most magnetic materials of interest [23].

The effective field corresponding to this case is

$$\mathbf{H}_{\text{ca}} = \frac{2K_1}{\mu_0 M_s} \left(\frac{\mathbf{M}}{M_s} \cdot \hat{\mathbf{e}} \right) \hat{\mathbf{e}}. \tag{2.18}$$

A detailed description of the various forms of the magnetocrystalline anisotropy can be found in textbooks on magnetic materials [23, 1].

2.3 Dynamic Micromagnetics and the Landau-Lifshitz-Gilbert equation

The general aim of a micromagnetic model is to predict the behaviour of a magnetic body under a variety of conditions. This can either be done by finding the minimum energy of the body or by modelling the dynamics of the magnetisation.

[3] It is possible to define many different forms for this energy. Note that the choice of definition changes the meaning of K_1.

In this thesis we are concerned only with dynamic models. Such dynamic models calculate the time evolution of the magnetisation using a differential equation. Dynamic models are often advantageous even when only the final state of the system is required because a magnetic system typically has multiple local energy minima and (standard) energy models cannot distinguish which of these minima the system will reach.

Starting with the quantum mechanical angular momentum of an electron, and converting from a single spin to a continuous magnetisation gives the differential equation [23, p. 306]

$$\frac{\partial \mathbf{M}}{\partial t} = -|\gamma_{\mathrm{L}}|\mathbf{M} \times \mathbf{H}_{\mathrm{eff}}, \tag{2.19}$$

where $\gamma \approx 1.761 \times 10^{11} s^{-1} T^{-1}$ is the electron gyromagnetic ratio and $\mathbf{H}_{\mathrm{eff}}$ is the total effective field

$$\mathbf{H}_{\mathrm{eff}} = \mathbf{H}_{\mathrm{ap}} + \mathbf{H}_{\mathrm{ca}} + \mathbf{H}_{\mathrm{ex}} + \mathbf{H}_{\mathrm{ms}}. \tag{2.20}$$

Equation (2.19) represents the precession of the magnetisation about the effective field, as shown in Figure 2.2. Other effective fields can be added to (2.20) to model other physical effects such as finite temperature or magnetostriction.

Equation (2.19) describes an undamped precessional motion, *i.e.* no energy is lost and the magnetisation continues to precess forever. This is obviously not the case for many problems of interest so an empirical damping term is added [65]:

$$\frac{\partial \mathbf{M}}{\partial t} = -|\gamma_{\mathrm{L}}|\mathbf{M} \times \mathbf{H} - \frac{\alpha_L |\gamma_{\mathrm{L}}|}{M_s}\Big(\mathbf{M} \times (\mathbf{M} \times \mathbf{H})\Big), \tag{2.21}$$

where α is an experimentally determined material parameter related to the strength of the damping and we write $\mathbf{H} = \mathbf{H}_{\mathrm{eff}}$. Equation (2.21) is referred to as the Landau-Lifshitz (LL) equation.

However, there are still problems with this equation. For large damping: as the damping constant is increased the total speed of the motion increases (since the magnitude of the damping term increases and the precession term remains unchanged), which is physically incorrect [71]. There is a similar equation introduced by Gilbert [42], called the Landau-Lifshitz-Gilbert (LLG) equation, which

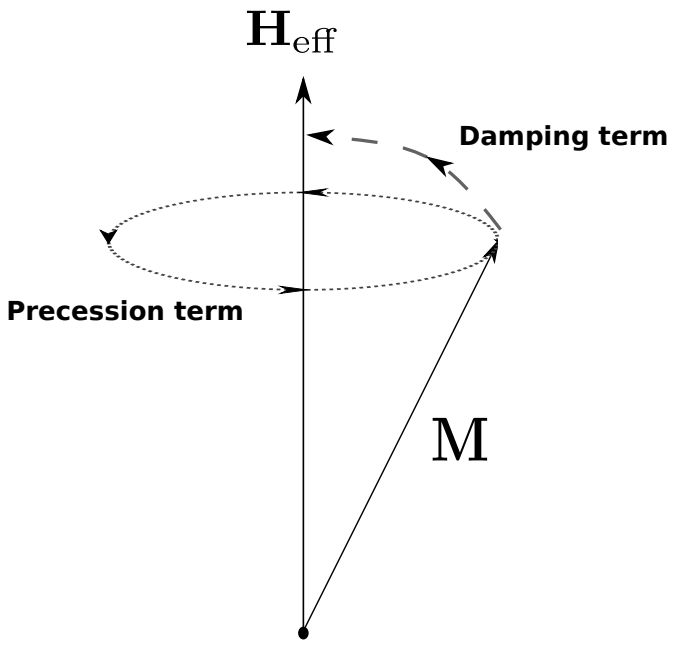

Figure 2.2: The effects of the precession and damping terms in the Landau-Lifshitz and Landau-Lifshitz-Gilbert equations on the magnetisation direction, $\mathbf{M}$*, of a single magnetic moment with a constant effective field.*

does not suffer from this problem[4]

$$\frac{\partial \mathbf{M}}{\partial t} = -|\gamma_{\mathrm{L}}|\mathbf{M} \times \mathbf{H} + \frac{\alpha}{M_s}(\mathbf{M} \times \frac{\partial \mathbf{M}}{\partial t}), \tag{2.22}$$

where $\alpha \neq \alpha_L$. Equation (2.22) can be rearranged into the same form [1, p. 181] as (2.21), giving

$$\frac{\partial \mathbf{M}}{\partial t} = \frac{-|\gamma_{\mathrm{L}}|}{1+\alpha^2}\Big[(\mathbf{M} \times \mathbf{H}) + \frac{\alpha}{M_s}\Big(\mathbf{M} \times (\mathbf{M} \times \mathbf{H})\Big)\Big], \tag{2.23}$$

which we refer to as the Landau-Lifshitz form of the Landau-Lifshitz-Gilbert equation.

Equations (2.21)–(2.23) link the unknown magnetisation vector as a function of time, $\mathbf{M}(t)$, with the effective field, $\mathbf{H}$. To determine $\mathbf{M}$ at an arbitrary time we need to integrate either of the above equations with respect to time starting

[4] A number of papers give different signs for the LLG equation, this may be due to confusion over the use of γ_L to mean $|\gamma_L| = -\gamma_L$. Alternatively it may be due to the lack of specification of the sign of the damping parameter in Gilbert's derivation of the LLG [42] (originally from his thesis), which uses the alternative notation $\alpha = -\eta\,|\gamma_L|\,M_s$ where η is the usual damping parameter. The correct signs are the ones given here and in reference [70].

from a known initial magnetisation $\mathbf{M}(0)$. This is usually done numerically by employing a time integration scheme, as will be discussed in Section 3.4.

Note that there is no spatial dependence directly contained in (2.21)–(2.23), *i.e.* they are ordinary differential equations (ODEs). However the exchange effective field and magnetostatic field usually contain spatial dependence, so the final equation to be solved is usually a partial differential equation (PDE).

2.3.1 Boundary conditions

By far the most commonly used boundary conditions on the magnetisation $\mathbf{M}$ are [1, pp. 178, 181] [79]:

$$\mathbf{M} \times \frac{\partial \mathbf{M}}{\partial \hat{\mathbf{n}}} = \mathbf{0}. \tag{2.24}$$

This condition can be arranged into a simpler form: we first note that for (2.24) to hold we require either $\mathbf{M} = \mathbf{0}$, $\frac{\partial \mathbf{M}}{\partial \hat{\mathbf{n}}} = \mathbf{0}$, or $\mathbf{M}$ parallel $\frac{\partial \mathbf{M}}{\partial \hat{\mathbf{n}}}$. Clearly $\mathbf{M}$ is not the zero vector inside the ferromagnet. We also know that the magnetisation length does not change,[5] hence $\frac{\partial \mathbf{M}}{\partial \mathbf{a}} \cdot \mathbf{M} = 0$ for any direction $\mathbf{a}$ (since the change in magnetisation projected onto the magnetisation direction is the change in magnetisation length). Therefore $\mathbf{M}$ and $\frac{\partial \mathbf{M}}{\partial \hat{\mathbf{n}}}$ cannot be parallel. So the final boundary condition is simply

$$\frac{\partial \mathbf{M}}{\partial \hat{\mathbf{n}}} = \mathbf{0}. \tag{2.25}$$

Alternatively, if the solution is periodic and the domain is infinite then periodic boundary conditions can be used [55]. In this case the value of the magnetisation must be identical on opposite sides of the domain. The domain of the simulation can then be thought of as infinite.

2.3.2 Temperature effects

All equations so far in this section are for the zero temperature case. Moderate temperature effects (significantly below the Curie temperature) can modelled by adding a stochastic term to the effective field [25]. High temperatures require variation of the material parameters with temperature, as well as additional terms

[5] Physically this is part of the definition of the problem. In Section 2.5 we demonstrate that this is embodied in the LLG equation.

to model the dynamic changes of the magnetisation length when near the Curie temperature. This is typically done by using the Landau-Lifshitz-Bloch equation instead of the Landau-Lifshitz-Gilbert equation, see for example [29]. Such models are beyond the scope of this thesis.

2.4 Non-dimensionalisation

In general it is useful to remove all unnecessary parameters from a differential equation before attempting to solve it. The main advantage of this is that we simplify the problem and greatly reduce the number of numerical experiments needed to cover all possible situations. Also for all problems the appropriate discretisation parameters (time step size and mesh size) will be roughly the same.

2.4.1 Non-dimensionalisation of the Landau-Lifshitz-Gilbert equation

We start from the Landau-Lifshitz-Gilbert equation with the magnetostatic, applied, exchange and magnetocrystalline (effective) fields. We use * to denote the dimensional variables, operators and constants that will be non-dimensionalised:

$$\frac{\partial \mathbf{M}^*}{\partial t^*} = -|\gamma_{\mathrm{L}}|\mathbf{M}^* \times \mathbf{H}^* + \frac{\alpha}{M_s}\mathbf{M}^* \times \frac{\partial \mathbf{M}^*}{\partial t^*}, \tag{2.26}$$

$$\mathbf{H}^* = \mathbf{H}^*_{\mathrm{ap}} - \nabla^*\phi^* + \frac{2A^*}{\mu_0 M_s}\nabla^{*2}\mathbf{m} + \frac{2K_1^*}{\mu_0 M_s}(\frac{\mathbf{M}^*}{M_s} \cdot \hat{\mathbf{e}})\hat{\mathbf{e}}, \tag{2.27}$$

$$\nabla^{*2}\phi^* = \nabla^* \cdot \mathbf{M}^*, \tag{2.28}$$

We assume for simplicity that M_s, A and K_1 are all constant throughout the magnetic materials used.

Let

$$\begin{aligned} \mathbf{M}^* &= M_s\mathbf{m}, \\ \phi^* &= \Phi\phi, \\ \mathbf{H}^* &= M_s\mathbf{h}, \\ K_1^* &= \mathbb{K}\mathcal{K}_1, \\ t^* &= \frac{1}{|\gamma_{\mathrm{L}}|M_s}t, \\ x_i^* &= lx_i. \end{aligned} \tag{2.29}$$

Note that $\mathbf{M}$ and $\mathbf{H}$ have the same units so we use the same normalisation factor. For dimensional purposes derivatives are equivalent to division by the variable differentiated with respect to, so $\nabla^* = \frac{1}{l}\nabla$, $\frac{\partial a}{\partial t^*} = |\gamma_{\mathrm{L}}|M_s\frac{\partial a}{\partial t}$ etc.

Combining (2.26) with the definitions (2.29) gives

$$\frac{\partial\mathbf{m}}{\partial t}|\gamma_{\mathrm{L}}|M_s^2 = -|\gamma_{\mathrm{L}}|M_s^2\mathbf{m}\times\mathbf{h} + \frac{\alpha}{M_s}|\gamma_{\mathrm{L}}|M_s^3\mathbf{m}\times\frac{\partial\mathbf{m}}{\partial t}, \tag{2.30}$$

and cancelling the various constants results in the non-dimensional Landau-Lifshitz-Gilbert equation

$$\frac{\partial\mathbf{m}}{\partial t} = -(\mathbf{m}\times\mathbf{h}) + \alpha(\mathbf{m}\times\frac{\partial\mathbf{m}}{\partial t}). \tag{2.31}$$

Similarly for (2.28) we have

$$\begin{aligned} \frac{1}{l^2}\Phi\nabla^2\phi &= \frac{1}{l}M_s\nabla\cdot\mathbf{m}, \\ \frac{\Phi}{M_s l}\nabla^2\phi &= \nabla\cdot\mathbf{m}. \end{aligned}$$

Letting $\Phi = M_s l$ we obtain

$$\nabla^2\phi = \nabla\cdot\mathbf{m}. \tag{2.32}$$

Repeating the substitutions for (2.27) gives

$$\begin{aligned} M_s\mathbf{h} &= M_s\mathbf{h}_{\text{ap}} - \frac{\Phi}{l}\nabla\phi + \frac{2A}{\mu_0 M_s}\frac{1}{l^2}\nabla^2\mathbf{m} + \frac{2\mathcal{K}_1}{\mu_0 M_s}\mathbb{K}(\mathbf{m}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}}, \\ \mathbf{h} &= \frac{M_s}{M_s}\mathbf{h}_{\text{ap}} - \frac{M_s}{M_s}\nabla\phi + \frac{2A}{\mu_0 M_s^2}\frac{1}{l^2}\nabla^2\mathbf{m} + \frac{2\mathcal{K}_1}{\mu_0 M_s^2}\mathbb{K}(\mathbf{m}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}}, \\ \mathbf{h} &= \mathbf{h}_{\text{ap}} - \nabla\phi + \frac{2A}{\mu_0 M_s^2}\frac{1}{l^2}\nabla^2\mathbf{m} + \frac{2\mathcal{K}_1}{\mu_0 M_s^2}\mathbb{K}(\mathbf{m}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}}, \end{aligned}$$

This can be further simplified by choosing the exchange length[6] as the characteristic length scale

$$l = \sqrt{\frac{2A}{\mu_0 M_s^2}}, \tag{2.33}$$

and

$$\mathbb{K} = \frac{\mu_0 M_S^2}{2}. \tag{2.34}$$

So the final system of equations is

$$\begin{aligned} \frac{\partial\mathbf{m}}{\partial t} &= -\mathbf{m}\times\mathbf{h} + \alpha\mathbf{m}\times\frac{\partial\mathbf{m}}{\partial t}, \\ \mathbf{h} &= \mathbf{h}_{\text{ap}} - \nabla\phi + \nabla^2\mathbf{m} + \mathcal{K}_1(\mathbf{m}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}}, \\ \nabla^2\phi &= \nabla\cdot\mathbf{m}. \end{aligned} \tag{2.35}$$

From (2.24) the dimensional boundary conditions on are:

$$\mathbf{M}^*\times\frac{\partial\mathbf{M}^*}{\partial\hat{\mathbf{n}}^*} = 0. \tag{2.36}$$

Using again the substitutions from above this becomes

$$\begin{aligned} \frac{M_s^2}{l}\left(\mathbf{m}\times\frac{\partial\mathbf{m}}{\partial\hat{\mathbf{n}}}\right) &= \mathbf{0}, \\ \mathbf{m}\times\frac{\partial\mathbf{m}}{\partial\hat{\mathbf{n}}} &= \mathbf{0}. \end{aligned} \tag{2.37}$$

[6]There are actually two exchange lengths: one based on the strength of exchange as compared with the magnetostatic field and another by comparison with the magnetocrystalline anisotropy. We use the magnetostatic-field-based exchange length for normalisation to avoid division by zero in the case of zero magnetocrystalline anisotropy.

2.4.2 Non-dimensionalisation of the Landau-Lifshitz form of the LLG

The dimensional Landau-Lifshitz form of the LLG is given in (2.23):

$$(1+\alpha^2)\frac{\partial \mathbf{M}^*}{\partial t^*} = -|\gamma_{\mathrm{L}}|\mathbf{M}^* \times \mathbf{H}^* - \frac{|\gamma_{\mathrm{L}}|\alpha}{M_s}\mathbf{M}^* \times (\mathbf{M}^* \times \mathbf{H}^*). \tag{2.38}$$

The non-dimensionalisation process is essentially the same as for the Gilbert form, by substituting in the normalisations in (2.29) we obtain

$$(1+\alpha^2)\frac{\partial \mathbf{m}}{\partial t} = -\mathbf{m} \times \mathbf{h} - \alpha \mathbf{m} \times (\mathbf{m} \times \mathbf{h}). \tag{2.39}$$

The field equations are identical to those in (2.35).

Alternatively the time variable could be normalised differently to remove the factor of $(1+\alpha^2)$ by using

$$t^* = \frac{1+\alpha^2}{|\gamma_{\mathrm{L}}|M_s}t. \tag{2.40}$$

This results in

$$\frac{\partial \mathbf{m}}{\partial t} = -\mathbf{m} \times \mathbf{h} - \alpha \mathbf{m} \times (\mathbf{m} \times \mathbf{h}). \tag{2.41}$$

However this means that time scale changes (*i.e.* the t^*s are different) when switching between the two forms (2.31) and (2.41), making comparisons more difficult.

2.4.3 Non-dimensionalisation of the energy

We repeat the process used in Section 2.4.1 to get a set of non-dimensionalised energy equations, starting from the equations given in Section 2.2:

$$E^*_{\mathrm{ap}} = -\mu_0 \int_\Omega \mathbf{M}^* \cdot \mathbf{H}^*_{\mathrm{ap}} \,\mathrm{d}\Omega^*, \tag{2.42}$$

$$E^*_{\mathrm{ms}} = \frac{-\mu_0}{2} \int_\Omega \mathbf{M}^* \cdot \mathbf{H}^*_{\mathrm{ms}} \,\mathrm{d}\Omega^*, \tag{2.43}$$

$$E^*_{\mathrm{ex}} = A \int_\Omega (\nabla^* \mathbf{m})^2 \,\mathrm{d}\Omega^*, \tag{2.44}$$

$$E^*_{\mathrm{ca}} = -K^*_1 \int_\Omega (\mathbf{m} \cdot \hat{\mathbf{e}})^2 \,\mathrm{d}\Omega^*. \tag{2.45}$$

We substitute definitions (2.29), (2.33) and (2.34) and $E^* = E_\ltimes e = \mu_0 M_s^2 l^d \, e$ where d denotes the number of spatial dimensions to obtain

$$e_{\mathrm{ap}} = -\mu_0 M_s^2 l^d \frac{1}{E_\ltimes} \int_\Omega \mathbf{m} \cdot \mathbf{h}_{\mathrm{ap}} \,\mathrm{d}\Omega = -\int_\Omega \mathbf{m} \cdot \mathbf{h}_{\mathrm{ap}} \,\mathrm{d}\Omega, \tag{2.46}$$

$$e_{\mathrm{ms}} = \frac{-\mu_0 M_s^2 l^d}{2} \frac{1}{E_\ltimes} \int_\Omega \mathbf{m} \cdot \mathbf{h}_{\mathrm{ms}} \,\mathrm{d}\Omega = -\frac{1}{2} \int_\Omega \mathbf{m} \cdot \mathbf{h}_{\mathrm{ms}} \,\mathrm{d}\Omega, \tag{2.47}$$

$$e_{\mathrm{ex}} = \frac{\mu_0 M_s^2 l^2}{2} \frac{l^d}{l^2} \frac{1}{E_\ltimes} \int_\Omega (\nabla \mathbf{m})^2 \,\mathrm{d}\Omega = \frac{1}{2} \int_\Omega (\nabla \mathbf{m})^2 \,\mathrm{d}\Omega, \tag{2.48}$$

$$e_{\mathrm{ca}} = -\frac{\mu_0 M_s^2}{2} l^d \mathcal{K}_1 \frac{1}{E_\ltimes} \int_\Omega (\frac{\mathbf{M}^*}{M_s} \cdot \hat{\mathbf{e}})^2 \,\mathrm{d}\Omega = \frac{-\mathcal{K}_1}{2} \int_\Omega (\mathbf{m} \cdot \hat{\mathbf{e}})^2 \,\mathrm{d}\Omega. \tag{2.49}$$

Note that we have used

$$\mathrm{d}\Omega^* = \mathrm{d}(x^*)^d = l^d \,\mathrm{d}x^d = l^d \,\mathrm{d}\Omega. \tag{2.50}$$

2.5 Geometric properties of the continuous Landau-Lifshitz-Gilbert equation

The "geometric properties" of a differential equation are those properties which do not vary in time [54, p. 73], *i.e.* conservation laws. In this section we demonstrate some geometric properties of the Landau-Lifshitz-Gilbert equation.

For this purpose it is convenient to write the non-dimensional LLG equation in the form

$$\frac{\partial \mathbf{m}}{\partial t} = -\mathbf{m} \times (\mathbf{h} - \alpha \frac{\partial \mathbf{m}}{\partial t}). \tag{2.51}$$

We will also need the identity

$$(\mathbf{a},\, \mathbf{a} \times \mathbf{b}) = 0, \tag{2.52}$$

which is true for all inner products $(\cdot\,,\,\cdot)$ because $\mathbf{a} \times \mathbf{b}$ is perpendicular to $\mathbf{a}$ by the definition of the cross product. We refer to this identity as the triple product identity. In particular, it is true for the dot product of two vectors[7] (also known as scalar multiplication) and for the L^2 inner product of two vector functions $\mathbf{a}$, $\mathbf{b}$:

$$(\mathbf{a},\, \mathbf{b})_{L^2} = \int_\Omega \mathbf{a} \cdot \mathbf{b}\, \mathrm{d}\Omega. \tag{2.53}$$

We first show that length of the magnetisation at any point is constant over time. To see this we take the dot product of (2.51) with $\mathbf{m}$ and use (2.52) to obtain

$$\mathbf{m} \cdot \frac{\partial \mathbf{m}}{\partial t} = 0. \tag{2.54}$$

This equation implies that the change of $\mathbf{m}$ over time is always perpendicular to $\mathbf{m}$, so length is conserved.[8]

Next we look at the energy properties of the system described by the Landau-Lifshitz-Gilbert equation. The change in energy over time can be derived similarly to the change in magnetisation by taking the L^2 inner product of each side of (2.51) with $\mathbf{h} - \alpha \frac{\partial \mathbf{m}}{\partial t}$. We then use the triple product identity (2.52) and the linearity of inner products to find that

$$\left(\mathbf{h},\, \frac{\partial \mathbf{m}}{\partial t}\right)_{L^2} - \alpha \left(\frac{\partial \mathbf{m}}{\partial t},\, \frac{\partial \mathbf{m}}{\partial t}\right)_{L^2} = 0. \tag{2.55}$$

Using the fact that $\mathbf{h} = -\frac{\delta e}{\delta \mathbf{m}}$ and the chain rule for functional derivatives, we

[7] Although the dot product is an inner product we follow convention and write it as $\mathbf{a} \cdot \mathbf{b}$ rather than in the bracket notation used for other inner products.

[8] Actually this is obvious from the fact that $\frac{\partial \mathbf{m}}{\partial t}$ is can be written as a cross product involving $\mathbf{m}$, but the technique used here is useful in the discussion of the properties of various discrete forms of the Landau-Lifshitz-Gilbert equation.

find that the time derivative of the energy is given by

$$\begin{aligned}\frac{\partial e[\mathbf{m}(\mathbf{x},t),t]}{\partial t} &= \left(\frac{\delta e}{\delta \mathbf{m}}, \frac{\partial \mathbf{m}}{\partial t}\right)_{L^2} - \left(\frac{\partial \mathbf{h}_{\text{ap}}}{\partial t}, \mathbf{m}\right)_{L^2} \\ &= -\left(\mathbf{h}, \frac{\partial \mathbf{m}}{\partial t}\right)_{L^2} - \left(\frac{\partial \mathbf{h}_{\text{ap}}}{\partial t}, \mathbf{m}\right)_{L^2},\end{aligned}$$

and so

$$\left(\mathbf{h}, \frac{\partial \mathbf{m}}{\partial t}\right)_{L^2} = -\frac{\partial e}{\partial t} - \left(\frac{\partial \mathbf{h}_{\text{ap}}}{\partial t}, \mathbf{m}\right)_{L^2}. \tag{2.56}$$

Finally, substituting (2.56) into (2.55) leaves

$$\frac{\partial e}{\partial t} = -\alpha \left(\frac{\partial \mathbf{m}}{\partial t}, \frac{\partial \mathbf{m}}{\partial t}\right)_{L^2} - \left(\frac{\partial \mathbf{h}_{\text{ap}}}{\partial t}, \mathbf{m}\right)_{L^2}. \tag{2.57}$$

Equation (2.57) shows that under constant applied field the energy of the system is always decreasing at a rate proportional to α. For non-constant applied fields the change in the Zeeman energy can increase or decrease the energy depending on how the field is changed. In fact, the first term of (2.57) can be easily derived from the Rayleigh dissipation functional used as the basis for the derivation of the Gilbert form of the LLG [42]. However the approach used here is useful for determining equivalent properties in discrete forms of the LLG.

Note that length conservation is a *point-wise* property, *i.e.* $|\mathbf{m}| = 1$ at every point in space. In contrast the energy decay/conservation is a *global* property, *i.e.* an integral over all space is conserved. This is related to the use of the dot product and the L^2 inner product respectively in the derivations above.

Chapter 3

Numerical Methods for Dynamic Micromagnetic Modelling

3.1 Introduction

In this chapter we give an overview of numerical methods that have previously been used in dynamic micromagnetic calculations. These calculations involve solving some form of the LLG equation (2.23) with effective fields determined by energy derivatives as discussed in Chapter 2. These systems of partial differential equations (PDEs) can only be solved analytically in a few simple cases [1], so numerical solution methods are almost always necessary.

3.1.1 Components of a numerical method for solving PDEs

Many numerical methods for solving non-linear PDEs, such as the LLG (2.23), can be thought of as a combination of various component methods, each of which handles a different part of the conversion from a continuous PDE into an algorithm which can be performed by a computer[1].

The essential components of a numerical method for solving time-dependent PDE problems are:

[1]Technically proofs of convergence etc. should be carried out for every different combination of component methods [54, p. 382]; in practice surprises seem to be rare, at least within micromagnetics.

- Spatial discretisation: convert spatial derivatives into algebraic relationships, typically between discrete points in space, *e.g.* finite elements, finite differences, macrospin models.

- Time discretisation (a.k.a. time integration): convert time derivatives into algebraic relationships, again typically between points in time, *e.g.* Runge-Kutta methods, backwards difference formulae.

For some choices of time and space discretisations additional component methods are needed:

- Linearisation procedure: Solve a system of non-linear algebraic equations, often by an iterative procedure which requires the solution of a sequence of systems of linear algebraic equations, *e.g.* Newton-Raphson method, Picard/fixed-point iteration.

- Linear solver: solve a system of linear algebraic equations, *e.g.* LU decomposition, Krylov solvers, multigrid methods.

Additionally in micromagnetic modelling the calculation of the magnetostatic field is required. If the integral form of the magnetostatic field, given in (2.13), is used then an additional integral evaluation method is needed to handle it. Alternatively, if the equivalent potential formulation, given in (2.14) and (2.15), is used then the magnetostatic field calculation simply requires solving additional PDEs, although there are some issues with the application of the boundary condition at infinity which are discussed in Section 3.3.2.

3.1.2 Desirable properties of a numerical method

Ideally all numerical methods would run quickly on cheap hardware and with sufficient accuracy. Unfortunately this is not the case in general and we are forced to make trade-offs. In this section we roughly describe some *ideal* properties of a numerical method (although obtaining all of these properties in a single method is almost certainly impossible).

The accuracy of a simulation is generally proportional to the "fineness" of the discretisations used (*i.e.* the number of points used in the approximation), but

increasing the fineness also increases the computation time. However, careful choice of algorithms can improve the accuracy at no additional cost or improve the speed with no loss of accuracy. The following properties of methods are important in this respect:

- The discretisation fineness in time should be dictated only by what is needed to accurately resolve the physics and not by numerical properties of the discretisation. In particular improved spatial fineness should not require improved time fineness.
- The optimal fineness in both time and space should be determined automatically, and should be allowed to vary as needed. This allows us to get the best "value" for our computational effort when increasing the fineness of the discretisation.
- The computation time should increase as little as possible as fineness in space or time increases. The minimum increase is linear scaling ($\mathrm{O}(N)$, where N is the number of discrete points involved) because we always have to do *something* with each value. Attaining this property for spatial discretisation usually impacts the choice of linear solver and/or magnetostatic calculation: many methods do not scale this well.
- The method should be able to handle non-trivial geometries without extortionate computational cost.

Additionally we would like our algorithms to be robust, that is they should reliably be able to give a correct answer no matter what parameters are used (up to any tolerances specified). In other words there should be few edge cases or "interesting" surprises when applying the method to various physical problems. Some important properties which improve robustness are:

- The method should obey any qualitative physical properties of the continuous equation.
- Mathematical evidence that all parts of the method will "work" on any well defined problem (*e.g.* proofs of convergence, etc.).
- The numerical method should automatically detect difficulties and apply the required effort to obtain the correct answer.

In addition to these properties there are a few practical considerations. The method should be reasonably simple: people need to understand a method before they can implement it, and even users need a surface level understanding. Finally the method should be as general as possible to reduce implementation effort and to simplify testing (by allowing a greater selection of example problems).

3.2 Spatial Discretisation

3.2.1 Macrospin models

In a granular magnetic material (a material consisting of magnetic grains separated by a non-magnetic material) the simplest way to handle spatial variation in the problem is to assume that within each grain the exchange effective field is so strong that the magnetisation is constant, *i.e.* that each grain behaves as a single *macrospin*. We assign a single value of $\mathbf{m}$ to each macrospin and proceed to calculate energy, effective field and/or magnetisation of each macrospin as required. One caveat is that the effects of the magnetostatic field of a grain on the grain itself is not automatically accounted for since there is no modelling of intra-grain effects. Hence it must be calculated and included separately to the magnetostatic interactions between grains. When included in this way the magnetostatic self field is often called the *shape anisotropy* since it is dependent on the shape of the grain and acts in a similar way to the magnetocrystalline anisotropy.

The obvious downside of a macrospin approach is that it only applies to fairly specific geometries, although the case of a granular media has been of much interest for magnetic data storage. Additionally, if there are nonuniformities in magnetisation within the regions where it has been assumed constant the model may be inaccurate.

However it is often simpler to construct a macrospin model than to use the more general methods described in Sections 3.2.2 and 3.2.3. Also the assumption that each grain has uniform magnetisation can reduce the number of calculations needed.

Closely related to macrospin models are atomistic models [28]. In these models the assumption of a continuous magnetisation $\mathbf{m}(\mathbf{x}, t)$ is dropped in favour of an assumption that each atom is the location of a single macrospin. As such they

allow the modelling of some systems that are inaccessible using micromagnetics, such as anti-ferromagnetic materials. However, the required spatial resolution for an atomistic model comes with an increase of a few orders of magnitude in the computational cost. Additionally such methods are only accurate for materials where the magnetic moment is localised to an atom, which is not always the case.

3.2.2 Finite Difference Methods

A widely used method of spatial discretisation is the finite difference method: a single magnetisation vector is assigned to each point on (or "cell" in) a square/cubic grid which covers the entire domain. Spatial derivatives in the PDE are then approximated by using truncated Taylor series expansions.

The finite difference method works well for very simple geometries when the grid can be lined up with all geometric features. For example when we are interested in how the magnetisation evolves over time in a non-granular cuboid-shaped thin film of magnetic material a finite difference method will be sufficient (*e.g.* in the µmag standard problems [73]).

However when the geometry involves curves, diagonals, hexagonal grains, bit patterned media or any other more complex geometric system alternative methods may be better suited.

3.2.3 Finite Element Methods

A more complex method of spatial discretisation is the finite element method [91]. Here the magnetic body is divided up into a finite number of non-overlapping *elements*, which can vary in both shape and size. Within each of these elements the spatial variation of the magnetisation is approximated by a polynomial function, typically a linear polynomial. Spatial derivatives are easily calculated by differentiating these polynomial functions, although higher derivatives can require some tricks. In contrast to macrospin and finite difference methods the actual equations solved by the finite element method are a *weak* form of the standard, or *strong*, form of the PDE. More details of the weak form and the finite element method are given in Section 4.1.

The main advantage of the finite element method is that it can cheaply and accurately (compared to the finite difference method) approximate non-trivial

geometrical features by an using an appropriate mesh of elements. An additional advantage is that the size of elements, and thus the accuracy of the approximation, can be varied arbitrarily as needed to give better accuracy in more complex or important regions. The choice of element size can be done automatically using *adaptive mesh refinement*: after each calculation an error estimate is calculated. If the error is determined to be too high in a region the mesh is refined near that region and the calculation is repeated (and similarly if the error is very in a region low the mesh can be unrefined). Hence, given the desired error and a method to estimate the error, a mesh giving an efficient and accurate approximation can be automatically generated [85].[2]

The major downside of finite element models is that the underlying mathematics is more complex than that required for other methods. Also the set up time and memory usage can be greater because of the additional "bookkeeping" required to keep track of the more complex meshes.

3.3 Magnetostatic Field Calculations

Methods for the calculation of magnetostatic fields belong to two categories: integral and potential methods. Integral methods are based on efficiently approximating the very large number of integrals required by (2.13). Potential methods are based on the solution of the PDE (2.14) with standard spatial discretisation methods combined with some special treatment is required for the boundary condition at infinity.

In this section we use the term "discrete magnetisations" to mean the macrospins/cells/nodes as appropriate for the spatial discretisation. We use N to denote the number of discrete magnetisations.

[2]Adaptive mesh refinement can also be done in finite difference methods, but as it results in a non-uniform mesh this removes their main advantage of simplicity and uniformity.

3.3.1 Integral Methods

Integral methods are based on the expression

$$\mathbf{H}_{\mathrm{ms}}(\mathbf{x}) = \frac{1}{4\pi}\left[-\int_{\Omega} \nabla' \cdot \mathbf{M}(\mathbf{x}') \frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}\, \mathrm{d}^3\mathbf{x}' + \int_{\Gamma} \mathbf{M}(\mathbf{x}') \cdot \hat{\mathbf{n}}(\mathbf{x}') \frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}\, \mathrm{d}^2\mathbf{x}'\right]. \tag{3.1}$$

A naive way to evaluate the integrals in (3.1) would be to calculate the field at each node due to each other node. So for each node a sum over all other nodes must be calculated. Hence this results in a computation time that scales as $\mathrm{O}(N^2)$, which is usually unacceptably slow for reasonably large N. Instead, more advanced techniques are used which treat the large number of integral evaluations in a more efficient way.

Fast Fourier transform methods

The fast Fourier transform method (FFT) is a simple and efficient method for calculating the magnetostatic field when the discrete magnetisations are positioned on a regular lattice.

The calculation of $\mathbf{H}_{\mathrm{ms}}$ in (2.13) can be thought of as applying a convolution to the list of discretised $\mathbf{m}$ values in the frequency domain. The matrix corresponding to this convolution is only dependent on the geometry, hence it can be precomputed and its Fourier transform precalculated. Then all that is needed to calculate the magnetostatic field is to apply a Fourier transform to $\mathbf{M}$, compute the convolution and transform the result back into the time domain by applying the inverse Fourier transform. Because of the regularity, applying the convolution in the frequency domain is very fast and hence the complexity of the calculation is limited by the complexity of a fast Fourier transform. This results in an overall complexity of $\mathrm{O}(N \log(N))$ [58].

The downside of this method is that points to be calculated must be on a regular lattice, similar to the finite difference method. Hence, it is most useful in combination with models using a finite difference spatial discretisation.

Alternatively the FFT may be used as part of a method applicable to less regular meshes. In such "non-uniform FFT" methods distant discrete magnetisations are approximated by magnetisations on a regular lattice so that an FFT method can be applied while the effects of nearby magnetisations are evaluated directly [58].

Fast multipole method

An alternative method of calculation of the magnetostatic field is the fast multipole method (FMM) [11]. The fast multipole method takes advantage of the fact that distant magnetic charge has a much smaller contribution to the total field at a point than a nearby magnetic charge due to the $\frac{1}{(\mathbf{x}-\mathbf{x}')^2}$ scaling in (3.1). Hence much less accuracy is needed in the calculation of these distant contributions in order to obtain the same overall accuracy.

For the field calculation at a specific point the full calculation is only performed for nearby discrete magnetisations. Groups of more distant magnetisations are approximated (lumped) as a single multipole placed at the centre of the group. The trick for quickly calculating fields at a large number of points is to pre-calculate multipole approximations for a range of accuracies over all space. Then the calculation of a field at a single point only requires the full calculation of effects from a few nearby points and from the appropriate multipoles.

The main advantage of this method over the fast Fourier transform is that it allows for arbitrary geometries.

3.3.2 Scalar potential methods

Potential methods are based on the formulation

$$\begin{aligned} \mathbf{h}_{\text{ms}} &= -\nabla\phi, \\ \nabla^2\phi &= \nabla\cdot\mathbf{m}, \end{aligned} \tag{3.2}$$

with the boundary conditions

$$\begin{aligned} \phi^{\text{int}} - \phi^{\text{ext}} &= 0 \quad \mathbf{x}\in\Gamma, \\ \frac{\partial\phi^{\text{int}}}{\partial\hat{\mathbf{n}}} - \frac{\partial\phi^{\text{ext}}}{\partial\hat{\mathbf{n}}} &= \mathbf{m}\cdot\hat{\mathbf{n}} \quad \mathbf{x}\in\Gamma, \end{aligned} \tag{3.3}$$

and

$$\phi \to 0 \text{ as } |\mathbf{x}| \to \infty, \tag{3.4}$$

see Section 2.2.2 for details.

Inside the domain ϕ can be solved for simply by applying finite element or finite difference discretisation methods. However (3.4), the boundary condition on ϕ

at infinity, is problematic. We obviously can not apply this condition directly using standard methods, since that would involve either an infinite number of elements or infinitely large elements (actually these are possible but impose heavy limitations on the allowed mesh geometries see *e.g.* [3] and [34] respectively). Hence more advanced techniques must be used, such techniques are the subject of the rest of this chapter.

The system of equations for the internal field calculations results in a well known linear algebra problem which can be solved in $O(N)$ time by well studied linear solvers (more details are given in Section 3.6 and Chapter 6). However applying or calculating boundary conditions can require additional processing time.

Asymptotic boundary conditions

One way to avoid an infinite domain is to truncate the external region at some finite distance from the magnetic domain. A similar but more sophisticated and accurate approach is to use asymptotic boundary conditions [102]. The idea here is to use a truncated external region to calculate the boundary conditions on the magnetic domain that correspond to (3.4) being applied at infinity. Additionally the fact that any solution to the Poisson equation (3.2) can be represented as an infinite series of harmonic functions is used to improve the accuracy.

Unfortunately the accuracy of this approach is still quite low, even for large truncation distances [15].

The hybrid finite element method/boundary element method

The idea of the hybrid finite element method/boundary element method (FEM/BEM) is to replace the external domain by a dipole layer placed on the surface of the magnetic domain which gives the correct boundary condition at infinity [38]. This removes the need to truncate or discretise the infinite external domain. The name comes from the close relationship between the use of a dipole layer and the boundary element method, and the fact that the standard finite element method is used to calculate the required potentials in the bulk. The full details of the method applied to magnetostatic calculations is discussed in Chapter 5.

A comparison by Bottauscio [15] found that using the FEM/BEM method was

more accurate than applying asymptotic boundary conditions (ABCs) for a calculation of the time evolution of the magnetisation of a sphere with zero exchange coupling. Even with a truncation distance of four times the size of the magnetic sphere (*i.e.* the radius of the computational domain was 4 times larger then that of the sphere) the accuracy when using asymptotic boundary conditions was worse[3] and did not improve between truncation distances of three and four times the magnetic sphere radius. Even when exchange coupling was added (giving an easier test) the ABC method was worse[4] than the FEM/BEM method.

The main downside of the FEM/BEM method is that it involves dense matrix of size $N_b \times N_b$ where where N_b is the number of boundary nodes (see Chapter 5 for details). This matrix must be stored in memory[5], and multiplied by a vector for the calculation of the boundary conditions. The speed of calculation of the boundary values in the method is limited by the dense matrix multiplication which, the cost of which is $\mathrm{O}(N_b^2)$. Similarly, the memory usage is $\mathrm{O}(N_b^2)$, which can limit the size of problems. The use of hierarchical matrix techniques can reduce both the speed and the memory usage to $\mathrm{O}(N_b \log(N_b))$ [61].

In 3D structures which are roughly spherical $N_b = \mathrm{O}\big(N^{2/3}\big)$ and so the use of a hierarchical matrix gives optimal computation speed[6] but for extremely flat structures such as thin films the number of boundary nodes can be as large as $N_b = N$. Hence the speed of the FEM/BEM method depends on the geometry.

Another downside of FEM/BEM is the increase in the complexity of the model: some components of the FEM/BEM method are fundamentally different to FEM. As such substantial additional code and mathematical knowledge may be needed for its implementation when standard FEM libraries are used as a basis.

3.4 Time Integration

Time integration schemes are used to convert the time derivatives of a PDE into a form that can be handled computationally. When discussing time integration

[3] After one precession cycle the relative error in M_x obtained using ABC method was around 1, compared to around 0.1 for FEM/BEM.

[4] A relative error of around 0.2.

[5] In theory individual regions of the matrix could be recomputed on the fly when needed but the computational cost would be inordinately high.

[6] With hierarchical matrix techniques the speed is $\mathrm{O}\big(N^{2/3}\log(N^{2/3})\big)$ but $\log(x) \ll x^{1/2}$ for large x, hence $\mathrm{O}\big(N^{2/3}\log(N^{2/3})\big) \ll \mathrm{O}(N)$, *i.e.* optimal complexity.

schemes it is helpful to use a (vector) initial value ordinary differential equation (ODE) as an example:

$$\begin{aligned} \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} &= \mathbf{f}(t, \mathbf{y}), \\ \mathbf{y}(0) &= \mathbf{y}_0, \end{aligned} \tag{3.5}$$

where $\mathbf{f}(t, \mathbf{y})$ is a known function and $\mathbf{y}_0$ is the known initial condition. After the application of one of the spatial discretisations described above a time-dependent PDE is typically reduced to the form (3.5); this is known as a *semi-discretised* PDE.

The time integration methods discussed here all bear a strong similarity to the finite difference method discussed in Section 3.2.2, except that the independent variable discretised is time instead of space. This is appropriate since time is one-dimensional, hence no complex geometry is possible and so typically there is no need for the more advanced finite element method.

Some key attributes of a time integration scheme are [7]:

- **Accuracy (order)** – An estimate of how rapidly the approximate solution converges to the exact solution as the time step size is reduced.
- **Stability** – Roughly speaking a scheme is stable if the approximate solution does not "blow up" (*i.e.* become catastrophically inaccurate, infinite, oscillate in non-physical ways, ...) for sufficiently large time steps, see Section 3.4.4 for a more rigorous description.
- **Conservation properties** – Some differential equations have properties which should ideally be conserved in the discretised system. For example the magnetisation length and energy properties of the Landau-Lifshitz-Gilbert equation discussed in Section 2.5. Schemes which conserve such properties are often referred to as "geometric" integrators.
- **Self-starting** – A scheme is self starting if it only requires a single initial value. This is desirable because methods of estimating additional initial values make the final scheme more complex and may introduce additional error.

These attributes are discussed more rigorously in the rest of this section.

3.4.1 Explicit and implicit time integration schemes

Explicit time discretisation schemes calculate the value at the next time step in terms of the value(s) at the present and/or previous time steps. The simplest such scheme is the *(forward) Euler method*

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \Delta_n \mathbf{f}(t_n, \mathbf{y}_n), \tag{3.6}$$

where $\Delta_n = t_{n+1} - t_n$ is the n-th time step size and $\mathbf{y}_i$ is the approximation to $\mathbf{y}(t_i)$. Clearly, given $\mathbf{f}(t, y)$ and an initial value for $\mathbf{y}(t_0)$, we can calculate $\mathbf{y}(t_n)$ for any n.

In contrast to explicit schemes, implicit schemes use the value at the next time step in its own calculation. Hence at each step a (linear or non-linear) system of equations must be solved. The simplest implicit scheme is the first order *backwards difference formula* (BDF1, also known as backwards Euler)

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \Delta_n \mathbf{f}\big(t_{n+1}, \mathbf{y}_{n+1}\big). \tag{3.7}$$

At first glance it seems that the additional solution of a system of equations required for one step of an implicit scheme mean would that explicit schemes are more efficient, and it is true that one step of an implicit method requires more computational effort than one step of an explicit method. However all explicit time integration schemes suffer from issues of limited stability for some problems, and so are forced to use time step sizes much smaller than would be required for reasons of accuracy (see Section 3.4.4 for details). In these cases implicit schemes can be much more efficient; problems where this is the case are often called "stiff" problems. The semi-discretisation of a PDE often results in such a stiff problem. In Chapter 9 we investigate stiffness in micromagnetics problems.

3.4.2 Some implicit time integration schemes

For the remainder of this thesis we will focus on three time integration schemes with good stability and accuracy properties that are widely employed in practice. All three are implicit schemes for reasons of stability, as mentioned in Section 3.4.1.

Trapezoid rule (TR) is the average of the forward and backward Euler methods from Section 3.4.1 and is defined as [44, p. 260]:

$$\mathbf{y}_{n+1} = y_n + \Delta_n(\mathbf{f}\big(t_{n+1}, \mathbf{y}_{n+1}\big) + \mathbf{f}(t_n, \mathbf{y}_n))/2. \tag{3.8}$$

The *second order backwards difference formula* (BDF2) is [44, p. 715]:

$$\frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{\Delta_n} = \frac{\Delta_n}{2\Delta_n + \Delta_{n-1}} \frac{\mathbf{y}_n - \mathbf{y}_{n-1}}{\Delta_{n-1}} + \frac{\Delta_n + \Delta_{n-1}}{2\Delta_n + \Delta_{n-1}} \mathbf{f}(t_{n+1}, \mathbf{y}_{n+1}). \tag{3.9}$$

The method that this thesis focuses on most of all (for reasons that will become clear through the remainder of this section) is the *implicit midpoint rule* (IMR). It is given by [44, p. 263]:

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \Delta_n \mathbf{f}\left(\frac{t_{n+1} + t_n}{2}, \frac{\mathbf{y}_n + \mathbf{y}_{n+1}}{2}\right). \tag{3.10}$$

Note that for cases where $\mathbf{f}$ is linear in both t and $\mathbf{y}$, IMR is equivalent to TR, and in general the properties of the two methods are similar.

IMR can be easily implemented in terms of BDF1 by the following simple algorithm: take a step of BDF1 (the simplest possible implicit method) using $\Delta_n^{\text{BDF1}} = \Delta_n^{\text{IMR}}/2$ to find $\mathbf{y}_{\text{mid}}$ at t_{mid}. Then calculate $\mathbf{y}_{n+1}$ using a rearrangement of $\mathbf{y}_{\text{mid}} = \frac{\mathbf{y}_n + \mathbf{y}_{n+1}}{2}$ [69]:

$$\mathbf{y}_{n+1} = 2\mathbf{y}_{\text{mid}} - \mathbf{y}_n, \tag{3.11}$$

and set $t_{n+1} = t_{\text{mid}} + \Delta_n^{\text{IMR}}/2$.

TR and IMR are both self starting while BDF2 requires an additional start-up value. This can be generated, for example, by taking a single step of IMR or TR before beginning the use of BDF2.

3.4.3 Local truncation error and order

The *local truncation error* (LTE) of a time integration scheme is the error due to a single integration step. It can be calculated by substituting $\mathbf{y}_n = \mathbf{y}(t_n)$, $\mathbf{y}_{n-1} = \mathbf{y}(t_{n-1})$, etc. into the approximation for the next time-step, then subtracting the

result from the exact solution at the next time-step, $\mathbf{y}(t_{n+1})$, *i.e.*

$$T_n = \mathbf{y}(t_{n+1}) - \hat{\mathbf{y}}_{n+1}, \tag{3.12}$$

where $\hat{\mathbf{y}}_{n+1}$ is the approximation to $\mathbf{y}$ at time t_{n+1} given when the exact solution is used for all history values (*i.e.* using $\mathbf{y}_n = \mathbf{y}(t_n)$, $\mathbf{y}_{n-1} = \mathbf{y}(t_{n-1})$, ...). For example the local truncation error of IMR is

$$T_n^{\mathrm{IMR}} = \mathbf{y}(t_{n+1}) - \mathbf{y}(t_n) - \Delta_n \mathbf{f}\left(t_{n+\frac{1}{2}}, \frac{\mathbf{y}(t_n) + \mathbf{y}_{n+1}^{\mathrm{IMR}}}{2}\right). \tag{3.13}$$

If the local truncation error of a time integration scheme is such that

$$T_n \leq c\Delta_n^{p+1} \tag{3.14}$$

then we say that the scheme is of order p.

It is important to distinguish the local truncation error from the *global error*

$$E_n = \mathbf{y}(t_{n+1}) - \mathbf{y}_{n+1}. \tag{3.15}$$

The difference is that the global error includes any error accumulated over previous time steps (*i.e.* the exact values $\mathbf{y}(t_n)$ in (3.12) are replaced by their approximations). Note that because of this accumulation the global error will typically be larger than the LTE. The global error of IMR expressed in the same form as (3.13) is

$$E_n^{\mathrm{IMR}} = \mathbf{y}(t_{n+1}) - \mathbf{y}_n^{\mathrm{IMR}} - \Delta_n \mathbf{f}\left(t_{n+\frac{1}{2}}, \frac{\mathbf{y}_n^{\mathrm{IMR}} + \mathbf{y}_{n+1}^{\mathrm{IMR}}}{2}\right). \tag{3.16}$$

The TR and BDF2 methods are both second order. Their local truncation errors are [44, p. 261]

$$T_n^{\mathrm{TR}} = -\frac{\Delta_n^3 \mathbf{y}_n'''}{12} + \mathrm{O}\big(\Delta_n^4\big), \tag{3.17}$$

and [44, p. 715]

$$T_n^{\mathrm{BDF2}} = -\frac{(\Delta_n + \Delta_{n-1})^2}{\Delta_n(2\Delta_n + \Delta_{n-1})} \frac{\Delta_n^3 \mathbf{y}_n'''}{6} + \mathrm{O}\big(\Delta_n^4\big). \tag{3.18}$$

The derivations of (3.17) and (3.18) are simple and can be found in most text books on the subject.

We now give a derivation of the local truncation error of IMR, which is somewhat complex because of the approximation $\mathbf{y} \approx \frac{\mathbf{y}_n+\mathbf{y}_{n+1}}{2}$. We first write the Taylor expansion $\mathbf{y}(t_{n+1})$ and $\mathbf{y}(t_n)$ about $t_{n+\frac{1}{2}}$. We use the midpoint rather than one of the end points (a typical choice in such calculations) because it results in simpler expressions. Assuming that $\mathbf{y}(t)$ is "sufficiently smooth" for the required derivatives to exist, its Taylor series expansion at t_{n+1} about $t_{n+\frac{1}{2}}$ is given by

$$\mathbf{y}(t_{n+1}) = \mathbf{y}(t_{n+\frac{1}{2}}+\frac{\Delta_n}{2}) = \mathbf{y}(t_{n+\frac{1}{2}})+\frac{\Delta_n}{2}\mathbf{y}'(t_{n+\frac{1}{2}})+\frac{\Delta_n^2}{8}\mathbf{y}''(t_{n+\frac{1}{2}})+\frac{\Delta_n^3}{48}\mathbf{y}'''(t_{n+\frac{1}{2}})+\mathrm{O}\big(\Delta_n^4\big). \tag{3.19}$$

Similarly, the expansion at t_n about $t_{n+\frac{1}{2}}$ is

$$\mathbf{y}(t_n) = \mathbf{y}(t_{n+\frac{1}{2}}-\frac{\Delta_n}{2}) = \mathbf{y}(t_{n+\frac{1}{2}})-\frac{\Delta_n}{2}\mathbf{y}'(t_{n+\frac{1}{2}})+\frac{\Delta_n^2}{8}\mathbf{y}''(t_{n+\frac{1}{2}})-\frac{\Delta_n^3}{48}\mathbf{y}'''(t_{n+\frac{1}{2}})+\mathrm{O}\big(\Delta_n^4\big). \tag{3.20}$$

Substituting (3.19) and (3.20) into (3.13) gives[7]

$$T_n^{\mathrm{IMR}} = \underbrace{\frac{\Delta_n^3}{24}\mathbf{y}'''(t_{n+\frac{1}{2}})}_{\mathrm{I}} + \underbrace{\Delta_n\left[\mathbf{y}'(t_{n+\frac{1}{2}}) - \mathbf{f}\left(t_{n+\frac{1}{2}}, \frac{\mathbf{y}(t_n)+\mathbf{y}_{n+1}^{\mathrm{IMR}}}{2}\right)\right]}_{\mathrm{II}} + \mathrm{O}\big(\Delta_n^4\big). \tag{3.21}$$

There are two parts to (3.21): the first term (I) is fairly standard for second order time integration schemes (*c.f.* the LTEs for TR and BDF2). The second term (II) originates from the use of an approximation to $\mathbf{y}(t_{n+\frac{1}{2}})$ in the evaluation of $\mathbf{f}$ (*i.e.* the Runge-Kutta nature of IMR). The rest of the derivation requires applying Taylor expansions to II and is carried out in Appendix C. The final result shows that IMR is of second order:

$$T_n^{\mathrm{IMR}} = \frac{\Delta_n^3}{24}\left[\mathbf{y}'''(t_{n+\frac{1}{2}}) - 3F_{n+\frac{1}{2}}\cdot\mathbf{y}''(t_{n+\frac{1}{2}})\right] + \mathrm{O}\big(\Delta_n^4\big), \tag{3.22}$$

where $F_{n+\frac{1}{2}}$ is the Jacobian matrix of $\mathbf{f}$ with respect to $\mathbf{y}$ evaluated at $t = t_{n+\frac{1}{2}}$. An additional condition required in the derivation is that the magnitude of the eigenvalues of $\frac{\Delta_n}{2}F_{n+\frac{1}{2}}$ are less than one. This condition is easily satisfied in the asymptotic case for most $\mathbf{f}$, the case where it does not hold is discussed in more detail in Section 3.4.6.

For a true comparison with the other local truncation errors we would need to

[7]If we had chosen to Taylor expand about t_n there would be an additional term in $\mathbf{y}_n''$ in (3.21).

shift this approximation to t_n. However a simple comparison can be given easily: for $\mathbf{f}$ linear in $\mathbf{y}$, t TR and IMR are the same, hence term I must be the same as TR in this case,. A second error term is then added to T_n^{IMR} corresponding to II for non-linear $\mathbf{f}$. Hence TR has the smallest LTE of the three methods discussed for most real-world ODEs. The relative magnitude of the additional term in T_n^{IMR} is highly problem dependent, see Section 3.4.6 for details, so the comparison of the LTEs of IMR and BDF2 is more complex.

Note however that for the accuracy of calculations the *accumulated* local truncation error is what matters, so these comparisons are not the whole story.

3.4.4 A-stability

To discuss the stability of time integrators the following initial value ODE is widely used:

$$\begin{aligned} \mathbf{f}(\mathbf{y}) &= \lambda \mathbf{y}, \\ \mathbf{y}_0 &= 1. \end{aligned} \tag{3.23}$$

Its analytical solution is

$$\mathbf{y}(t) = \exp(\lambda t). \tag{3.24}$$

A-stability[8] is the property that a method is has no stability restrictions on the time step size when used to integrate (3.23) for all λ with $\text{Re}(\lambda) \leq 0$.

The IMR, TR, BDF1 and BDF2 methods are all A-stable [47, pgs. 43, 251]. Linear explicit methods are never A-stable [76] ("linear" methods in this reference include all common time integration methods: Runge-Kutta, multistep, predictor-corrector, ...), which is the main reason for our focus on implicit methods. Additionally there are no A-stable linear multistep methods of order greater than two [44, p. 261], which is why we have chosen the methods in Section 3.4.2 for discussion. It should be noted that A-stable implicit Runge-Kutta methods of higher order do exist, see *e.g.* [47, p. 73], but they require the solution of significantly larger systems of equations at each time step.

[8]The A does not stand for anything, it is just "A" [47, p. 40]. In particular it does *not* mean "absolute stability".

3.4.5 Spurious numerical damping

Some time integration methods create additional, non-physical damping in approximate solutions, even when none exists in the true solution. This can be problematic for the modelling of highly oscillatory ODEs, such as the LLG with low damping parameter α.

In the solution of the ODE (3.23) with $\lambda = i\omega$ (and $\omega \in \mathbb{R}$), $|y|$ should not decrease over time. If $|y_n|$ does decrease over time in the approximate solution given by a time integration scheme, then we say that the scheme causes spurious damping.

For example, for IMR:

$$\begin{aligned} y_{n+1} &= y_n + \Delta_n i\omega(y_n + y_{n+1})/2, \\ &= \frac{1 + i\Delta_n\omega/2}{1 - i\Delta_n\omega/2} y_n, \end{aligned} \tag{3.25}$$

therefore

$$\begin{aligned} |y_{n+1}|^2 &= \frac{|1 + i\Delta_n\omega/2|^2}{|1 - i\Delta_n\omega/2|^2} |y_n|^2, \\ |y_{n+1}|^2 &= \frac{(1 + i\Delta_n\omega/2)(1 - i\Delta_n\omega/2)}{(1 - i\Delta_n\omega/2)(1 + i\Delta_n\omega/2)} |y_n|^2, \\ &= |y_n|^2, \end{aligned} \tag{3.26}$$

hence IMR does not cause spurious damping.

Since f is linear in y for this example TR and IMR are identical

$$\begin{aligned} y_{n+1} &= y_n + \Delta_n(i\omega y_n + i\omega y_{n+1})/2, \\ &= y_n + \Delta_n i\omega(y_n + y_{n+1})/2, \end{aligned} \tag{3.27}$$

and the same property applies for TR.

For BDF1, however, we have:

$$\begin{aligned} y_{n+1} &= y_n + \Delta_n i\omega y_{n+1}, \\ &= \frac{1}{1 - i\Delta_n\omega} y_n. \end{aligned} \tag{3.28}$$

Hence,

$$\begin{aligned} |y_{n+1}| &= \sqrt{\frac{1}{(1 - i\Delta_n\omega)(1 + i\Delta_n\omega)}}\,|y_n|\,, \\ &= \frac{1}{\sqrt{1 + \Delta_n^2\omega^2}}\,|y_n|\,, \end{aligned} \tag{3.29}$$

and the magnitude of y decreases over time, *i.e.* the solution is damped. The analysis for the second order BDF2 case is more complex due to the fact that it is a multistep method, but the result is the same: the solution suffers from non-physical damping [44, p. 265].

3.4.6 B-convergence and order reduction

It is known that certain implicit Runge-Kutta methods are susceptible to reduced accuracy when used to solve certain extremely stiff problems [6, p. 156] [47, p. 225]. In the case of the implicit midpoint rule this corresponds to cases when term II of the LTE (3.21) is large. This occurs when the error in $\mathbf{f}(t, \mathbf{y})$ due to a small error in $\mathbf{y}$ is large.

The concept of "B-convergence" is used to analyse this effect. Roughly speaking, a Runge-Kutta method is B-convergent of order r if the global error is $\mathrm{O}(\Delta_n^r)$ for "infinitely stiff" ODEs. The implicit midpoint rule is B-convergent of order 1 [47, p. 231],[9] which is one order less than its global error on typical ODEs. We call this effect "order reduction". In contrast, the TR and BDF methods do not suffer from order reduction [6, p. 159]. This is because all evaluations of $\mathbf{f}$ are at integer time points (*i.e.* $t_{n+1}, t_n, t_{n-1}, \ldots$ rather than $t_{n+\frac{1}{2}}$) and so there is no term in the LTE containing $\frac{\partial \mathbf{f}}{\partial \mathbf{y}}$.

This order reduction effect is a potential downside for IMR, but only when applied to specific ODEs (and no such effect has been reported in micromagnetics simulations we are aware). Additionally it should be accounted for by a good adaptive time step selection algorithm (see Section 3.4.7 for details).

A simple test ODE demonstrates this phenomenon [7, p. 157]:

$$\begin{aligned} f(t, y) &= -\lambda(y - g(t)) + g'(t), \\ y_0 &= g(0), \end{aligned} \tag{3.30}$$

[9] In this reference IMR is referred to as "the second order Gauss method", BDF1 is Radau IIA, TR is Lobatto IIIA [47, pp. 72-76].

for some function $g(t)$ and parameter $\lambda \geq 0$. The exact solution (which can be seen to be correct by substitution) is

$$y(t) = g(t). \tag{3.31}$$

Note that $\frac{\partial f}{\partial y} = \lambda$, so we can directly control the magnitude of term (II) of the IMR truncation error (3.21).

We now derive the LTE for IMR in this example when $|\lambda\Delta_n| \gg 1$. From (C.7) we have that

$$(1 + \frac{\Delta_n\lambda}{2})T_n^{\text{IMR}} = \frac{\Delta_n^3}{24}\Big[g'''(t_{n+\frac{1}{2}}) - 3\lambda g''(t_{n+\frac{1}{2}})\Big] + \mathrm{O}\big(\Delta_n^4\big). \tag{3.32}$$

Then using $|\lambda\Delta_n| \gg 1$:

$$\begin{aligned} \frac{\Delta_n\lambda}{2}T_n^{\text{IMR}} &= \frac{\Delta_n^3}{24}\Big[g'''(t_{n+\frac{1}{2}}) - 3\lambda g''(t_{n+\frac{1}{2}})\Big] + \mathrm{O}\big(\Delta_n^4\big), \\ T_n^{\text{IMR}} &= \frac{\Delta_n^2}{12\lambda}\Big[g'''(t_{n+\frac{1}{2}}) - 3\lambda g''(t_{n+\frac{1}{2}})\Big] + \mathrm{O}\big(\Delta_n^4\big). \end{aligned} \tag{3.33}$$

If additionally $|\lambda\Delta_n| \gg |g'''(t)|$, which will typically be true for very large λ and $g \neq g(\lambda)$:

$$T_n^{\text{IMR}} = \frac{-\Delta_n^2}{4}g''(t_{n+\frac{1}{2}}) + \mathrm{O}\big(\Delta_n^3\big). \tag{3.34}$$

So we can see that the order has been reduced by one power of Δ_n compared with the standard LTE (3.22) for cases when $|\lambda\Delta_n| < 1$.

3.4.7 Adaptivity

Adaptive time integration methods automatically select time step sizes in response to an estimate of the local truncation error. This can improve efficiency: for example on problems where a high frequency part of the solution is gradually damped out the time step can increase by orders of magnitude at later times without loss of accuracy. Adaptive integration also greatly simplifies the use of the numerical method: the user only has to choose the allowed LTE and the algorithm will handle the rest.

Most local truncation error estimators for implicit integrators (e.g. TR, BDF2) compute an explicit estimate of the solution $\mathbf{y}_{n+1}^E \sim \mathbf{y}(t_{n+1})$ (sometimes known as

a predictor step) to the same order of accuracy as the implicit method. The explicit method is selected such that it uses only the same derivatives as are required for the implicit method, so no additional $\mathbf{f}$ evaluations are needed. Algebraic rearrangements of the LTE expressions for the explicit and the implicit methods are then used to compute an approximation to the actual LTE [44, pp. 707-716]. This is known as Milne's device.

No such estimation is possible for IMR because of the specific form of the local truncation error (3.21). The details of this issue and the first (to our knowledge) algorithm for adaptive time integration with IMR are described in Chapter 7.

It is also possible to construct adaptive time step algorithms based on error estimates specific to the equation being solved. Such estimates have the obvious limitation that they are only (directly) applicable to the equations for which they were derived.

One example of such an algorithm for the LLG equation is to compare the effective damping with the expected damping and use the difference as an error estimator, as proposed by Albuquerque *et al.* [2]. A major disadvantage of this method is that the error estimator is unable to distinguish between spatial errors and temporal errors. Another possible method is to use the time error estimates for the LLG equation, with limited effective field terms and a specific finite element discretisation, derived for both IMR and BDF2 by Banas [8]. However the derivation is fairly complex, and may be difficult to extend to include exchange field or FEM/BEM magnetostatics.

3.4.8 Magnetisation length conservation

We now move on to the discussion of some properties specific to the LLG equation. As discussed in Section 2.5, the length of the magnetisation, $|\mathbf{m}|$, at each point in space is constant over time. However, in the approximations given by numerical methods this property is often lost, in which case it must be enforced separately. Note that this is not related to the spurious damping discussed in Section 3.4.5: in that section the key property was the conservation of the amplitude of *oscillations*.

One simple method of dealing with the constraint is to re-normalise each value of $\mathbf{m}$ after every time step. An extension of this method is to re-normalise the magnetisation only when the error in $|\mathbf{m}|$ exceeds some tolerance [34], we refer

to this method as *tolerance-based renormalisation*. Note that the computational cost of computing the errors in $|\mathbf{m}|$ is roughly the same as performing the re-normalisation[10], hence tolerance-based re-normalisation does not provide any gain in efficiency. Tolerance-based re-normalisation is used in magpar[11] with a default tolerance of 10^{-2}, while re-normalisation after every step is used in OOMMF's 4th order Runge-Kutta scheme [31]. However both of these re-normalisation approaches fundamentally change the system of equations being solved in a non-linear and unpredictable manner [66].

Another approach, which is closely related to re-normalisation, is the use of a modified LLG equation with an additional term which dynamically corrects errors in the magnetisation length [77] [35]. This method is used by Nmag [35] and is discussed in more detail later in this section.

Another way to avoid this problem is to use a spherical polar coordinate system (r, θ, ϕ) for the magnetisation. In these coordinates (2.23) can be expressed in terms of only the angles (θ, ϕ) representing the direction of $\mathbf{m}$ and the length constraint is automatically enforced since

$$\frac{\partial |\mathbf{m}|}{\partial t} = \frac{\partial r}{\partial t} \equiv 0. \tag{3.35}$$

However problems occur with this approach when the polar angle, θ, approaches zero because $\frac{\partial m_\phi}{\partial t} \propto \frac{1}{\sin(\theta)}$ [39] and so the derivative becomes singular. Also the calculation of the sum of the effective field vectors is much more complex in spherical polar coordinates (involving trigonometric identities), resulting in more complex non-linear/linear systems when using implicit time integration schemes.

A fourth approach is to use Lagrange multipliers to constrain the solution such that $|\mathbf{m}|$ is constant [97]. The main problem with this method is that an additional degree of freedom is needed for every value of $\mathbf{m}$ in space.

Geometric time integration schemes aim to solve the issue of length conservation by constructing a scheme that naturally preserves the value of $|\mathbf{m}|$. Some examples of such schemes are based on IMR [24], or a semi-implicit modification of the IMR using explicit extrapolations of some or all effective field terms [93, 87, 74, 15]. Alternatively, geometric integration methods based on Cayley transforms

[10] Both require computing $|\mathbf{m}|$ and performing some simple operation with it for every discrete value of the magnetisation.

[11] See `src/util/renormvec.c` in the source code of magpar version 0.9.

can be used [66, 16].

The self-correcting LLG

One interesting technique used to maintain the condition $|\mathbf{m}| = 1$ is to solve a modified "self-correcting" LLG equation [77] [35]:

$$\frac{\partial \mathbf{m}}{\partial t} = -\frac{1}{(1+\alpha^2)}\mathbf{m} \times \mathbf{h} - \frac{\alpha}{(1+\alpha^2)}\mathbf{m} \times (\mathbf{m} \times \mathbf{h}) + \beta \mathbf{m}\big(1 - |\mathbf{m}|^2\big), \qquad (3.36)$$

where β is some parameter which must be chosen. Equation (3.36) differs from the standard LL form of the LLG, (2.39), by the inclusion of the final term. The motivation behind this modification is as follows: if $|\mathbf{m}| > 1$ then during the next step there will be an additional term in $\frac{\partial \mathbf{m}}{\partial t}$ of $-\beta \mathbf{m}\left|1 - |\mathbf{m}|^2\right|$. As this additional term is in the $-\mathbf{m}$ direction the magnetisation length will be reduced. Similarly if $|\mathbf{m}| < 1$ there is a small additional term in the $+\mathbf{m}$ direction and the magnetisation length will be increased. For larger values of the parameter β this effect is stronger and so we expect the condition $|\mathbf{m}| = 1$ to be restored more rapidly.

The potential advantage of this method over the standard re-normalisation is that the re-normalisation is part of the differential equation rather than a separate step. This means that the time integration scheme "knows" about the re-normalisation and so may be able to choose more appropriate time step sizes (see Section 3.4.7).

A potential downside is that the resulting system is more non-linear than the standard LL/LLG, particularly when β is large. This could mean that additional steps are needed for the non-linear solver to converge, resulting in increased computation time.

Another downside is that the method is more complex to implement than re-normalisation after every step: it requires an additional term to be discretised, and adds additional terms to the Jacobian matrix.

Also an appropriate value for the parameter β must be selected somehow. To our knowledge there is no published research on the selection of appropriate values for β, we briefly study the effect of varying β in Appendix D.

Finally we note that the modification is only applicable to the LL form of the

LLG, and there is no obvious extension to the LLG form[12]. The LL form can be more complex to work with for some calculations due to the triple cross-product.

The self-correcting LLG method is used by Nmag [31] [35].

Conservation of |m| by IMR in a macrospin model

In this section we show that the discrete form of the Landau-Lifshitz-Gilbert equation that results from the use of the implicit midpoint rule maintains the magnetisation length conservation property of the continuous form shown in Section 2.5. These results are applicable to strong form equations, *i.e.* finite difference and macrospin discretisations but not finite elements. For the purposes of this section we assume that the discretised non-linear Landau-Lifshitz-Gilbert equation is satisfied exactly, *i.e.* the linear and/or non-linear solver used gives exactly correct results.

As can be seen from (3.10) the IMR is defined by the following substitutions:

$$\begin{aligned} \frac{\partial \mathbf{m}}{\partial t} &\to \frac{\mathbf{m}_{n+1} - \mathbf{m}_n}{\Delta_n}, \\ \mathbf{m} &\to \frac{\mathbf{m}_{n+1} + \mathbf{m}_n}{2}, \\ t &\to \frac{t_{n+1} + t_n}{2}. \end{aligned} \tag{3.37}$$

So the IMR-discretised version of the LLG is

$$\frac{\mathbf{m}_{n+1} - \mathbf{m}_n}{\Delta_n} = -\frac{\mathbf{m}_{n+1} + \mathbf{m}_n}{2} \times \left(\mathbf{h}\left[\frac{\mathbf{m}_{n+1} + \mathbf{m}_n}{2} \right] - \alpha \frac{\mathbf{m}_{n+1} - \mathbf{m}_n}{\Delta_n} \right). \tag{3.38}$$

The conservation properties of the IMR discretisation come from the cancellation of cross terms in the inner product $\left(\mathscr{L}[\mathbf{m}], \frac{\partial \mathbf{m}}{\partial t}\right)$ for any symmetrical linear

[12]The obvious way to proceed would be to establish a relationship between the modified LL form and the LLG form using the standard derivation of the LL form from the LLG form [1]. However this derivation assumes that $|\mathbf{m}| = 1$, meaning that the self-correcting term is zero and rendering the result meaningless.

operator $\mathscr{L}$. More precisely:

$$\begin{aligned}\left(\mathscr{L}\left[\frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}\right],\,\frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}\right) &= \frac{1}{2\Delta_n}\Big[\,(\mathbf{m}_{n+1},\,\mathscr{L}\mathbf{m}_{n+1})+(\mathbf{m}_{n+1},\,\mathscr{L}\mathbf{m}_n)\\ &\qquad -(\mathbf{m}_n,\,\mathscr{L}\mathbf{m}_{n+1})-(\mathbf{m}_n,\,\mathscr{L}\mathbf{m}_n)\,\Big]\\ &= \frac{1}{2\Delta_n}\Big[\,(\mathbf{m}_{n+1},\,\mathscr{L}\mathbf{m}_{n+1})-(\mathbf{m}_n,\,\mathscr{L}\mathbf{m}_n)\,\Big].\end{aligned} \tag{3.39}$$

This means that if $\mathscr{L}\mathbf{m}$ is orthogonal to $\frac{\partial \mathbf{m}}{\partial t}$ in the continuous or semi-discretised equations then it will also be orthogonal in the IMR-discretised version. In particular, setting $\mathscr{L}$ as the identity operator will allow us to show that $\mathbf{m}$ and $\frac{\partial \mathbf{m}}{\partial t}$ are orthogonal in this section and setting $\mathscr{L} = \mathscr{H}$ will allow us to show some nice properties of the energy in Section 3.4.9.

We now examine the change in magnetisation length using the same technique as in Section 2.5: take the dot product of (3.38) with $\frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}$ on both sides and use the triple product identity (2.52) to get

$$\frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}\cdot\frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}=0. \tag{3.40}$$

Now using (3.39) with $\mathscr{L}$ as the identity operator and the dot product as the inner product gives us

$$\frac{(\mathbf{m}_{n+1},\,\mathbf{m}_{n+1})-(\mathbf{m}_n,\,\mathbf{m}_n)}{2\Delta_n}=0. \tag{3.41}$$

Therefore

$$|\mathbf{m}_{n+1}| = |\mathbf{m}_n|\,, \tag{3.42}$$

i.e. the magnetisation length does not change between time steps.

Note that for other time integration schemes the property (3.39) typically does not hold. For example, in the case of BDF1 we have

$$\begin{aligned}\left(\mathscr{L}[\mathbf{m}_{n+1}],\,\frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}\right) &= \frac{1}{\Delta_n}[(\mathbf{m}_{n+1},\,\mathscr{L}\mathbf{m}_{n+1})-(\mathbf{m}_{n+1},\,\mathscr{L}\mathbf{m}_n)],\\ &\neq \frac{1}{\Delta_n}[(\mathbf{m}_{n+1},\,\mathscr{L}\mathbf{m}_{n+1})-(\mathbf{m}_n,\,\mathscr{L}\mathbf{m}_n)],\end{aligned} \tag{3.43}$$

i.e. orthogonality properties from the continuous form do not carry over to the discrete form.

Notice that the only requirement in the derivation of (3.40) is to use the midpoint approximation of magnetisation in the LLG itself. The derivation does not make any assumption about how the effective field is approximated. This explains why various semi-implicit modifications to IMR [93, 87, 74] which use explicit calculations of effective fields also conserve the magnetisation length.

3.4.9 Energy conservation/decay

Another geometrical property of the Landau-Lifshitz-Gilbert equation, also discussed in Section 2.5, is the conservation or monotonic decay of energy depending on the value of the damping constant α (and the applied field $\mathbf{h}_{\text{ap}}$).

IMR also conserves energy when $\alpha = 0$ and it ensures that the energy is a decreasing function of time when $\alpha > 0$ and $\mathbf{h}_{\text{ap}}$ is constant [24], a proof of this property is given in the next section. The other geometric integration schemes mentioned in Section 3.4.8, and in particular the semi-implicit modifications of IMR [93, 87, 74], do not have this property.

Proof of energy property for the IMR discretised macrospin model

In this section we show that the change in the energy of the magnetic system over time when evolved using the IMR discretised LLG equation obeys certain relationships which are similar to those of the continuous LLG equation.

Before starting the derivation we introduce some alternative notation for the effective field when considered as an operator, $\mathscr{H}[\mathbf{m}]$:

$$\begin{aligned} \mathbf{h}(\mathbf{x},t) &= \mathscr{H}[\mathbf{m}(\mathbf{x})] + \mathbf{h}_{\text{ap}}(\mathbf{x},t). \\ \mathscr{H}[\mathbf{m}(\mathbf{x})] &= \nabla^2\mathbf{m}(\mathbf{x}) + \mathbf{h}_{\text{ms}}[\mathbf{m}(\mathbf{x})] + \mathbf{h}_{\text{ca}}[\mathbf{m}(\mathbf{x})], \end{aligned} \tag{3.44}$$

We also expand the midpoint value of the applied field, $\mathbf{h}_{\text{ap}}\left(\frac{t_n+t_{n+1}}{2}\right)$, into

$$\begin{aligned} \mathbf{h}_{\text{ap}}\left(\frac{t_n+t_{n+1}}{2}\right) &= \frac{\mathbf{h}_{\text{ap}}(t_{n+1})+\mathbf{h}_{\text{ap}}(t_n)}{2} + \frac{\Delta_n^2}{4}\frac{\partial^2\mathbf{h}_{\text{ap}}}{\partial t^2}\bigg|_{\frac{t_{n+1}+t_n}{2}} + \mathrm{O}\big(\Delta_n^4\big), \\ &= \frac{\mathbf{h}_{\text{ap}}(t_{n+1})+\mathbf{h}_{\text{ap}}(t_n)}{2} + \mathcal{E}_{\text{ap}}, \end{aligned} \tag{3.45}$$

where

$$\mathcal{E}_{\text{ap}} = \frac{\Delta_n^2}{4} \frac{\partial^2 \mathbf{h}_{\text{ap}}}{\partial t^2}\bigg|_{\frac{t_{n+1}+t_n}{2}} + \mathrm{O}\big(\Delta_n^4\big), \tag{3.46}$$

is the error in this midpoint-like approximation of $\mathbf{h}_{\text{ap}}$.

Now we are ready to derive the energy property in the same manner as Section 3.4.8. We begin by taking the L^2 inner product of the IMR discretised LLG equation, (3.38), with the sum of the IMR discretised effective field and damping terms:

$$\begin{aligned} &\mathscr{H}\left[\frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}\right] + \mathbf{h}_{\text{ap}}\left(\frac{t_n+t_{n+1}}{2}\right) - \alpha\frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}, \\ &= \mathscr{H}\left[\frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}\right] + \frac{\mathbf{h}_{\text{ap}}(t_{n+1})+\mathbf{h}_{\text{ap}}(t_n)}{2} + \mathcal{E}_{\text{ap}} - \alpha\frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}. \end{aligned} \tag{3.47}$$

The result of this inner product is

$$\begin{aligned} &\left(\mathscr{H}\left[\frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}\right] + \frac{\mathbf{h}_{\text{ap}}(t_{n+1})+\mathbf{h}_{\text{ap}}(t_n)}{2} + \mathcal{E}_{\text{ap}},\ \frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}\right)_{L^2} \\ &\quad - \alpha\left(\frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n},\ \frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}\right)_{L^2} = 0, \end{aligned} \tag{3.48}$$

where the RHS is zero due to the triple product identity.

It can be shown (see *e.g.* Appendix B.1) that $\mathscr{H}$ is a linear symmetric operator on $\mathbf{m}$. So from (3.39) with $\mathscr{L} = \mathscr{H}$ we have the property

$$\left(\mathscr{H}\left[\frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}\right],\ \frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}\right)_{L^2} = \frac{1}{2\Delta_n}[(\mathbf{m}_{n+1},\ \mathscr{H}\mathbf{m}_{n+1})_{L^2} - (\mathbf{m}_n,\ \mathscr{H}\mathbf{m}_n)_{L^2}]. \tag{3.49}$$

Additionally the energy at time step n due to the effective field operator $\mathscr{H}$ can be written as[13]

$$e_{\mathscr{H},n} = -\frac{1}{2}\left(\mathbf{m}_n,\ \mathscr{H}[\mathbf{m}_n]\right)_{L^2}. \tag{3.50}$$

So using (3.49) and (3.50) we can write the $\mathscr{H}$ term of (3.48) as

$$\left(\mathscr{H}\left[\frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}\right],\ \frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}\right)_{L^2} = -\frac{e_{\mathscr{H},n+1} - e_{\mathscr{H},n}}{\Delta_n}. \tag{3.51}$$

Next we examine the applied field term of (3.48). Omitting the L^2 from the inner

[13] This essentially follows from the definitions of the non-dimensional energies and the effective fields given in Section 2.4.1, details are given in Appendix B.2.

products and temporarily writing $\mathbf{h}_{\text{ap}}(t_i)$ as $\mathbf{h}_i$ for brevity we have

$$\begin{aligned}\left(\frac{\mathbf{h}_{n+1}+\mathbf{h}_n}{2},\, \frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}\right) &= \left(\mathbf{h}_{n+1}+\mathbf{h}_n,\, \frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}\right) \\ &\quad - \left(\frac{\mathbf{h}_{n+1}+\mathbf{h}_n}{2},\, \frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}\right), \\ &= -\frac{e_{\text{ap},n+1}-e_{\text{ap},n}}{\Delta_n} + A,\end{aligned} \tag{3.52}$$

where

$$\begin{aligned}A &= \frac{1}{2\Delta_n}\Big[2\,(\mathbf{h}_n,\, \mathbf{m}_{n+1}) - 2\,(\mathbf{h}_{n+1},\, \mathbf{m}_n) \\ &\qquad - (\mathbf{h}_{n+1},\, \mathbf{m}_{n+1}) + (\mathbf{h}_{n+1},\, \mathbf{m}_n) - (\mathbf{h}_n,\, \mathbf{m}_{n+1}) + (\mathbf{h}_n,\, \mathbf{m}_n)\Big], \\ &= \frac{1}{2\Delta_n}\Big[-(\mathbf{h}_{n+1},\, \mathbf{m}_{n+1}) - (\mathbf{h}_{n+1},\, \mathbf{m}_n) + (\mathbf{h}_n,\, \mathbf{m}_{n+1}) + (\mathbf{h}_n,\, \mathbf{m}_n)\Big], \\ &= -\left(\frac{\mathbf{h}_{n+1}-\mathbf{h}_n}{\Delta_n},\, \frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}\right).\end{aligned} \tag{3.53}$$

Finally, we insert the results of (3.51)–(3.53) into (3.48) to find that the change in the total energy is

$$\begin{aligned}\frac{e_{n+1}-e_n}{\Delta_n} = &-\alpha\left|\left|\frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}\right|\right|^2_{L^2} - \left(\frac{\mathbf{h}_{\text{ap}}(t_{n+1})-\mathbf{h}_{\text{ap}}(t_n)}{\Delta_n},\, \frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}\right)_{L^2} \\ &+ \left(\mathcal{E}_{\text{ap}},\, \frac{\mathbf{m}_{n+1}-\mathbf{m}_n}{\Delta_n}\right)_{L^2}.\end{aligned} \tag{3.54}$$

The first term of this expression is exactly the discrete representation of the energy loss due to damping. The second term is the discrete representation of the energy change due to changes in the applied field. The third term is the error due to changes in the applied field which are not accurately captured by the approximations used in IMR. In particular note that the energy is conserved when $\alpha = 0$ and the applied field is constant. This relationship is illustrated in Figure 3.1.

When $\mathbf{h}_{\text{ap}}$ is piecewise linear, for example when an applied field is instantaneously switched on, time steps can be selected such that each non-linear change happens exactly at the start/end of a step. So again we have $\mathcal{E}_{\text{ap}} = 0$ and the same properties as for linear applied fields.

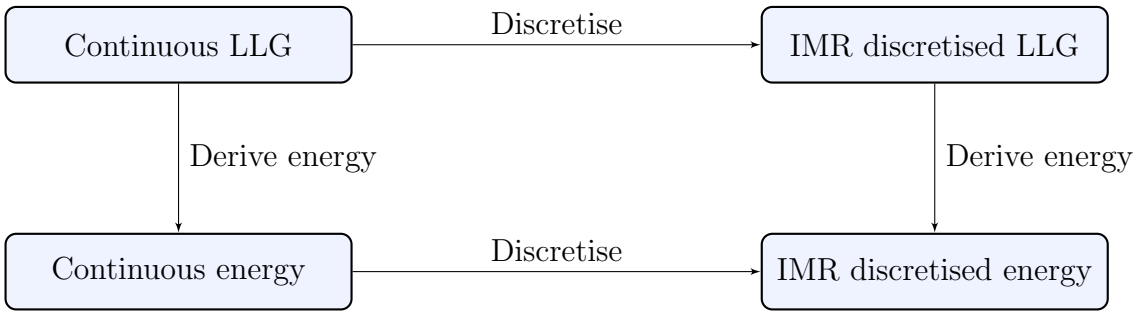

Figure 3.1: Commutation relationship between the IMR discretisation and the LLG energy derivation. In contrast to other time integration methods the result is independent of the order of operations.

It should be noted that none of the above is applicable to semi-implicit modifications of IMR since in that case $\mathscr{H}\left[\frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}\right]$ is replaced by an expression of the form $a\mathscr{H}[\mathbf{m}_n] + b\mathscr{H}[\mathbf{m}_{n-1}]$.

3.4.10 Conclusions

In the previous sections we have described a number of important properties of time integration schemes.

We also highlighted a number of interesting geometric integration properties of the implicit midpoint rule. There are a number of reasons to believe that the geometric integration properties will translate into an improvement in the accuracy and robustness of the overall solver:

- Errors in the energy dissipation [2] and magnetisation length [21] have been successfully used as error estimators for local truncation error. Hence the removal of energy or magnetisation length errors should reduce the total local truncation error.

- It is well known that geometric integration schemes can result in much smaller long-timescale error build-up than schemes that do not preserve such quantities [54, p. 77].

- The non-linear modification to the Landau-Lifshitz-Gilbert equation caused by renormalisation of the magnetisation length (as commonly used to maintain correct magnetisation length in non-conservative time integrators) may

cause significant step changes in the magnetostatic field [66]. This also modifies the balance between the various energy terms, which is similar to the methods that lead to the "flying ice cube" problem [48] in molecular dynamics.[14] Avoiding renormalisation eliminates the possibility of such errors occurring.

Additionally, the same geometric properties enable a proof of convergence for IMR when applied to a FEM semi-discretised LLG equation with a slight modification that will be discussed in Section 4.5 [10].

Note that no assumption of fixed Δ_n was made, and that there is no dependence on previous values of Δ_n. Hence the geometric properties are expected to still hold when the step sizes are varied.

3.5 Linearisation

When implicit time integration methods are applied to a non-linear differential equation (such as the LLG equation) a non-linear system of equations must be solved. Explicit time integration methods sidestep the issue of non-linearity by avoiding the need to solve any system.

In the context of implicit time integration of a semi-discretised PDE we write

$$\tilde{\mathbf{y}}_n = \left\{\mathbf{y}_{n,i}\right\}_{i=0}^{i=N}, \tag{3.55}$$

for the list of spatially discrete values of $\mathbf{y}$ at time step n. For example, for an LLG problem consisting of a single macrospin $\tilde{\mathbf{y}}_n = \mathbf{m}_n$. For an LLG problem semi-discretised by the finite element method $\tilde{\mathbf{y}}_n$ is the list of all the discrete values that define $\mathbf{m}_n$ along with any values needed for the magnetostatic calculations.

3.5.1 Functional iteration

Functional iteration is the process of repeatedly applying a function until the result converges to a fixed point. This process can be applied to find the solution

[14] In such problems the rescaling of particle velocities (to maintain constant temperature despite numerical error accumulation) can result in large amounts of kinetic energy being transferred from the motion of internal degrees of freedom to motion of the centre of mass.

of a non-linear system by constructing a function such that its fixed point solves the system.

As an example we can use functional iteration to solve the non-linear system resulting from the discretisation of a non-linear ODE, written as $\frac{\partial \mathbf{y}}{\partial t} = \mathbf{f}(t, y)$, by the implicit midpoint rule. We first solve for an auxiliary variable $\tilde{\mathbf{w}}$ using the iteration

$$\begin{aligned} \tilde{\mathbf{w}}^{(0)} &= \tilde{\mathbf{y}}_n, \\ \tilde{\mathbf{w}}^{(i+1)} &= \frac{\Delta_n}{2}\mathbf{f}(t_n + \frac{\Delta_n}{2}, \tilde{\mathbf{w}}^{(i)}) + \tilde{\mathbf{y}}_n. \end{aligned} \tag{3.56}$$

This variable can be thought of as the approximation for y used in the evaluation of f, *i.e.* $\mathbf{w} = (\tilde{\mathbf{y}}_{n+1} + \tilde{\mathbf{y}}_n)/2$. Once the iteration (3.56) has converged (as determined, for example, by $\left|\mathbf{w}^{(i+1)} - \mathbf{w}^{(i)}\right| < \epsilon$), the value of $\tilde{\mathbf{y}}_{n+1}$ is calculated using

$$\tilde{\mathbf{y}}_{n+1} = \tilde{\mathbf{y}}_n + \Delta_n \mathbf{f}(t_n + \frac{\Delta_n}{2}, \tilde{\mathbf{w}}). \tag{3.57}$$

The main advantage of functional iteration is that convergence proofs can be easily constructed via the Banach fixed point theorem (albeit under some fairly restrictive assumptions on time step size) [54, p. 125].

Additionally, the cost of each iteration can be very cheap if the function $\mathbf{f}(t, \mathbf{y})$ can be written explicitly (*i.e.* the ODE is not implicit in $\frac{\partial \mathbf{y}}{\partial t}$), since it only requires the application of a function rather than a linear solve required by the Newton-Raphson method. This is not the case for the LLG, but the linear systems resulting from functional iteration are simpler than those resulting from other methods, which can simplify the construction of an efficient linear solver [10].

However, the functional iteration only converges for time steps of a limited size. This limit is related to the spatial discretisation, resulting in similar problems to explicit methods. This makes the method useful for applying implicit methods to non-stiff problems, but less useful for stiff ones [54].

3.5.2 Newton-Raphson method

A non-linear problem can be written in residual form as

$$\text{Find } \tilde{\mathbf{y}}_{n+1} \text{ such that } \mathbf{r}(t_{n+1}, \tilde{\mathbf{y}}_{n+1}) = 0. \tag{3.58}$$

The motivation for the Newton-Raphson method comes from a simple Taylor expansion of the residual, $\mathbf{r}$. If the exact root of the residual is $\tilde{\mathbf{y}}^E_{n+1}$ and we have an initial guess for this root $\tilde{\mathbf{y}}^{(0)}_{n+1}$ then we can obtain a correction to this initial guess from:

$$\begin{aligned} 0 &= \mathbf{r}(t_{n+1}, \tilde{\mathbf{y}}^E_{n+1}), \\ &= \mathbf{r}(t_{n+1}, \tilde{\mathbf{y}}^{(0)}_{n+1} + \boldsymbol{\delta}), \\ &= \mathbf{r}(t_{n+1}, \tilde{\mathbf{y}}^{(0)}_{n+1}) + \mathrm{J}(t_{n+1}, \tilde{\mathbf{y}}^{(0)}_{n+1}) \cdot \boldsymbol{\delta} + \mathrm{O}\big(\boldsymbol{\delta}^2\big), \end{aligned} \tag{3.59}$$

where $\mathrm{J}(t_{n+1}, \tilde{\mathbf{y}}^{(0)}_{n+1})$ is the Jacobian matrix of the residual with respect to $\tilde{\mathbf{y}}_{n+1}$ evaluated at $\tilde{\mathbf{y}}^{(0)}_{n+1}$. From (3.59) we have

$$\begin{aligned} \tilde{\mathbf{y}}^E_{n+1} &= \tilde{\mathbf{y}}^{(0)}_{n+1} + \boldsymbol{\delta}, \\ &= \tilde{\mathbf{y}}^{(0)}_{n+1} + \mathrm{J}^{-1}(t_{n+1}, \tilde{\mathbf{y}}^{(0)}_{n+1}) \cdot \mathbf{r}(\tilde{\mathbf{y}}^{(0)}_{n+1}) + \mathrm{O}\big(\boldsymbol{\delta}^2\big). \end{aligned} \tag{3.60}$$

This suggests the iteration

$$\tilde{\mathbf{y}}^{(i+1)}_{n+1} = \tilde{\mathbf{y}}^{(i)}_{n+1} - \mathrm{J}^{-1}(t_{n+1}, \tilde{\mathbf{y}}^{(i)}_{n+1}) \cdot \mathbf{r}(\tilde{\mathbf{y}}^{(i)}_{n+1}), \tag{3.61}$$

in which, if the residual is sufficiently smooth and the required correction is "small", the error is *squared* after each iteration (*i.e.* convergence is quadratic).

The assumption that we can discard terms in $\mathrm{O}\big(\boldsymbol{\delta}^2\big)$ is fairly strong: it means that we need a good initial guess so that the initial correction, $\boldsymbol{\delta}$, is small. Fortunately in time integration problems the value at the previous time step provides a good initial guess: $\tilde{\mathbf{y}}^{(0)}_{n+1} = \tilde{\mathbf{y}}_n$. It is also possible to construct an even better initial guess by taking a single cheap explicit "predictor" step, as used in Milne's device for adaptive time integration (and if such an adaptive scheme is in use then this improved initial guess is "free").

The iteration is terminated when

$$\big|\big|\mathbf{r}(\tilde{\mathbf{y}}^{(i+1)})\big|\big|_\infty < \epsilon_{\mathrm{N}}, \tag{3.62}$$

for some user defined tolerance ϵ_{N}. Alternatively the absolute tolerance condition (3.62) can be replaced by, or combined with, a relative tolerance condition

$$\big|\big|\mathbf{r}(\tilde{\mathbf{y}}^{(i+1)})\big|\big|_\infty < \big|\big|\mathbf{r}(\tilde{\mathbf{y}}^{(0)})\big|\big|_\infty \epsilon_{\mathrm{Nr}}. \tag{3.63}$$

If the residual is normalised, *i.e.* has initial magnitude of O(1), then (3.62)

and (3.63) are roughly equivalent.

As discussed in Section 3.5.1 the main downside of the Newton-Raphson method is that it always requires the solution of a system of linear equations to obtain each correction $\boldsymbol{\delta}$, and that these systems are more complex than those produced by function iteration. The efficient solution of these systems is the subject of Section 3.6.

Another potential downside is that the Newton-Raphson method is only guaranteed to converge if the initial guess "good enough". However for non-linear systems resulting from time integration we always have a very good initial guess, so this is not often a problem.

3.6 Solution of sparse linear systems

If we are using an implicit time integration scheme with the Newton-Raphson method for linearisation, then we are left with the problem of solving a sequence of sparse linear systems. Each of these linear systems can be stated as the problem: Given a sparse $N_r \times N_r$ matrix B and a vector $\mathbf{b}$, find $\mathbf{x}$ such that

$$\mathrm{B}\mathbf{x} = \mathbf{b}. \tag{3.64}$$

This is a very widely studied problem, see for example [82], but efficient techniques for very large N_r (which corresponds to a large spatial problem and/or good spatial resolution) are complex and problem dependent. There are two general classes of linear solvers: direct solvers and iterative solvers. Direct solvers are very general, but are typically computationally expensive in both memory and time. Effective iterative solvers are usually problem specific but can be much more efficient and can even achieve optimal scaling (*i.e.* of computational complexity $\mathrm{O}(N_r)$ in time and memory).

3.6.1 Direct methods

Direct methods for the solution of linear systems seek exact solutions using methods based on Gaussian elimination. The direct method most commonly used in the solution of systems resulting from discretised PDEs is LU decomposition.

The main advantage of direct methods is their robustness: for reasonably non-singular systems and given enough time/memory a direct solver will be able to solve the system. However as matrix sizes become large direct solvers become increasingly demanding in both time and memory. The problem is that the factors of a typical sparse matrix can be significantly more dense than the original matrix, *i.e.* there is some amount of fill-in. This means that the time and memory requirements for the solution rarely scale as $\mathrm{O}(N_r)$ except for very specific matrices. However the exact level of fill-in depends on the sparsity pattern of the initial matrix, and so a single simple complexity estimate cannot be given.

The computational complexity of LU decomposition for a banded matrix is $\mathrm{O}(s^2 N_r)$, where s is the bandwidth (*i.e.* s such that the matrix entries $a_{kl} = 0$ if $|k - l| > s$) [54, p. 236]. As an example we consider the complexity for the system resulting from the discretisation of a Poisson problem in two dimensions on an $a \times a$ grid (by the finite element or finite difference method). In this case $s \propto a$ because there is an interaction between nodes in neighbouring rows and there are a nodes per row. Since $a \propto \sqrt{N_r}$ the overall complexity is $\mathrm{O}(N_r^2)$.

In three dimensional PDE problems there are typically more non-zero entries in each row due to the increased connectivity of the nodes, increasing the complexity of the method. For example, in the three dimensional equivalent of the example above (an $a \times a \times a$ grid) $s \propto a^2$ and $a \propto N_r^{1/3}$, hence the complexity is $\mathrm{O}\left(N_r^{7/3}\right)$.

3.6.2 Krylov solvers

The first class of iterative methods for the solution of linear systems that we discuss are the Krylov solvers [82, p. 151]. These methods are based on finding an approximation to the solution, $\mathbf{x}$, in the sequence of Krylov subspaces of the matrix B:

$$\mathcal{K}_m(\mathrm{B}, \mathbf{y}) = \mathrm{span}\left\{\mathbf{y}, \mathrm{B}\mathbf{y}, \mathrm{B}^2\mathbf{y}, \ldots, \mathrm{B}^m\mathbf{y}\right\}. \tag{3.65}$$

Note that computing a basis for the next Krylov subspace $\mathcal{K}_{m+1}$ only requires a matrix vector product with the matrix B applied to a basis vector of $\mathcal{K}_m$ (although in practice the simple basis in (3.65) is modified to improve the properties of the method). Hence constructing the subspace is computationally cheap for a sparse B. At each iteration of a Krylov method a new approximation to the solution in the space $\mathcal{K}_{m+1}$ is found by minimising some easy to calculate form of the error

in the direction given by the new Krylov basis vector.

The two Krylov solvers used in this thesis are the *method of conjugate gradients* (CG) and the *generalised minimum residual method* (GMRES). The basic procedure for these methods is: 1) Find a new search direction in the m-th Krylov subspace which is "orthogonal" to all previous directions. 2) Find the point in that direction which in some sense gives the "optimally accurate" solution and add this vector to the solution. The details of what is meant by "orthogonal" and "optimal" depends on the method.

The key difference between the two methods is that CG is only applicable to symmetric positive definite matrices, whereas GMRES is applicable to general matrices. However, CG is the more efficient of the two: it takes only $\mathrm{O}(N_r)$ computation time for each step of the solve, whereas GMRES takes $\mathrm{O}(N_r k)$ time for the k-th step (and similarly for the memory requirements). This is because CG uses the Lanczos algorithm (which can be represented as a 3-term recurrence relationship) for the orthogonalisation of the search directions, while GMRES uses Arnoldi's method [82, p. 153]. This limitation of GMRES can be avoided by restarting the method after some number of steps with the current approximation as the new initial guess, or by using an alternative Krylov solver such as BiCGStab [91, p. 172]. However, these methods lose some of the convergence guarantees of GMRES and CG, hence we avoid using them in this thesis.

The size of the Krylov subspace needed for an effective approximation (*i.e.* the number of iterations needed) is strongly dependent on the properties of the matrix, in particular its eigenvalues and eigenvectors. The maximum number of iterations to find a solution using CG or GMRES can be shown to be N_r, if numerical error is ignored. However, for the solver to be efficient the number of iterations should be far smaller than N_r, and preferably constant with respect to N_r.

In order to achieve optimal performance we often instead solve a modified system of the form

$$\mathrm{D}\mathbf{x} = \mathbf{c}, \quad \mathrm{D} = \mathcal{P}^{-1}\mathrm{B}, \quad \mathbf{c} = \mathcal{P}^{-1}\mathbf{b}, \tag{3.66}$$

where $\mathcal{P}$ is a *preconditioner* which is chosen such that the the matrix $\mathcal{P}^{-1}\mathrm{B}$ has the required eigenvalue and eigenvector properties for rapid convergence. Since the preconditioner must be applied at every step of the Krylov solve it should

be computationally cheap to compute $\mathcal{P}^{-1}\mathbf{x}$. The construction of a preconditioner which results in iteration counts independent of the matrix size while also being computationally cheap to set up and apply is a difficult and problem dependent task. In the next section we briefly discuss some approaches for such preconditioners.

Note that the preconditioner, $\mathcal{P}$, must be the same at each step in order for the analysis of most Krylov methods to hold. This limitation can be avoided by the use of the flexible GMRES method (FGMRES) [81], which has performance and memory usage very similar to standard GMRES.

3.6.3 Preconditioning of Krylov solvers

As mentioned above, the preconditioning of Krylov solvers is a critical component in obtaining good performance. However the construction of a good preconditioner is typically problem specific.

There are a number of "general purpose" preconditioners which have some beneficial effect on the convergence of the Krylov solver for most linear systems. However they typically do not result in low numbers of iterations independent of the number of rows in the matrix. One widely used general purpose preconditioner is the incomplete LU decomposition (ILU) [82, p. 287]. This method uses an LU decomposition, but with the number of additional non-zero values (the fill-in) limited in such a way that the matrix remains sparse and operations remain cheap. The simplest way to limit the fill-in is to allow only n "generations" of fill-in, *i.e.* with $n = 0$ no fill-in is allowed, with $n = 1$ only the original matrix elements are allowed to generate new elements, etc. We denote the ILU method with n levels of fill-in by ILU(n).

Another useful class of preconditioners are based on (algebraic) *multigrid* methods which are discussed in Section 3.6.4. Multigrid-based preconditioners are highly effective for the preconditioning of linear systems arising from Poisson problems (*i.e.* $\nabla^2\phi(\mathbf{x}) = f(\mathbf{x})$), and other similar problems [50].

An interesting preconditioning approach for linear systems arising from PDEs containing multiple physical effects is *block preconditioning*. This involves the division of the matrix B into blocks corresponding to the different physical degrees of freedom. A preconditioner can then be constructed by manipulating this

matrix on the block level, typically based on the physics of the problem. For example, by removing or eliminating off-diagonal blocks the preconditioner can be solved by inverting the diagonal blocks and using block back/forward substitution. The inverse of the diagonal blocks can be approximated using known effective preconditioners for the equivalent single-physics problem. In Chapter 6 we introduce such a preconditioning approach for the solution of the coupled LLG-magnetostatics system.

3.6.4 Multigrid methods

There are a number of simple iterative methods, called relaxation methods, which are fairly ineffective when applied directly to most linear systems but are very cheap to apply. Examples of these methods are Jacobi, Gauss-Seidel and SOR methods [82, p. 103].

Multigrid methods are a class of linear solvers which combine these relaxation methods with a hierarchical algorithm. They can be highly efficient and scalable solvers for the discrete systems arising from certain classes of PDEs [17]. Multigrid methods are based on two observations: firstly that relaxation methods are very effective at reducing the high-frequency (in space) error of a solution, but fail on the low frequency parts. The second observation is that many of the low frequency errors can be converted to high frequency errors by projecting them onto a coarser grid. So by solving the problem on a sequence of meshes all frequencies of error except for the very lowest can be treated effectively using only cheap relaxation methods. The lowest frequency errors can be well represented on a very coarse grid (*i.e.* a very small linear system), and so can be cheaply resolved using a direct solve.

Typical multigrid solvers are based on something similar to the following algorithm:

1 Start with initial guess for the solution on the finest mesh;
2 Apply a few iterations of a relaxation method;
3 Project the solution onto a coarser mesh;
4 Apply a few iterations of a relaxation method;
5 Project the solution onto a coarser mesh;
6 ⋮
7 On the coarsest mesh solve the resulting problem directly;
8 Project the solution onto a finer mesh;
9 Apply a few iterations of a relaxation method;
10 ⋮
11 End with the corrected solution on the finest mesh;

The algorithm above is called a V-cycle because a diagrammatic representation of the mesh sizes used draws a "V" shape. There are a number of similar but more complex cycles possible. The details of the cycle(s), along with the relaxation method used, the projection between meshes, the selection of coarse meshes etc. all depend on the particular variant of the multigrid method used.

It turns out that for certain problems multigrid solvers can achieve computation times which scale as $\mathrm{O}(N_r \log N_r)$, *i.e.* their performance does not severely degrade as the number of nodes in the mesh is increased. The memory usage of multigrid methods is around $\mathrm{O}(N_r)$, depending on the method used to obtain the coarse meshes. In particular, as noted in the previous section, multigrid methods are very effective on systems involving the matrix corresponding to a discrete Laplace operator. This makes them useful for a number of problems because the Laplace operator is an important part of many PDEs.

Standard multigrid methods make use of the geometric properties of the mesh underlying the PDE, which makes them difficult to apply to most unstructured meshes. However the same techniques can be applied without any knowledge of the underlying geometry using *algebraic multigrid* (AMG) methods. These methods use information purely from the matrix itself to construct "coarse" versions of the matrix based, for example, on the strength of the coupling between matrix elements.

3.7 Compatibility of methods

While in theory any of the methods discussed above can be combined with any other, there are certain combinations which are more effective than others. In this section we discuss which combinations of methods are particularly compatible or incompatible.

The use of integral methods for magnetostatic field calculations with implicit time integration is problematic: the integral formulation couples magnetisation at every point to every other point, meaning that the Jacobian representing the derivatives of each magnetisation with respect to each other magnetisation is a full matrix (and hence solution times are extremely slow). Explicit time integration methods avoid this issue because no system solve is required.

For problems with very simple geometry (*i.e.* thin films, cuboid shapes) the finite difference method with FFT for magnetostatic calculations is sufficient. Additionally: for simple geometry there is often no need for well refined meshes, so the problem can be only mildly stiff and explicit time integration methods can be efficient. The most well known example of such a model is OOMMF [26]. This combination of methods is very simple to implement and so is often used for GPU based code [99].

For problems with more complex geometry the finite element method must be used for spatial discretisation to obtain good accuracy. Magnetostatic calculations on the resulting non-uniform mesh are typically carried out using FEM/BEM (*e.g.* in Nmag [36], magpar [84] and femme) although a non-uniform FFT approach has been used in at least one model [20]. The use of refined meshes to fully resolve the geometry can often result in stiff problems, requiring the use of implicit time integration schemes for good efficiency (see Chapter 9 or [89]). In all "real world" models that we know of using implicit time integration the Newton-Raphson method is used for linearisation. Typically some form of preconditioned Krylov solver is then used to solve the resulting linear systems.

3.7.1 Solvers and geometric integration

It should be noted that the geometric properties of IMR derived in Sections 3.4.8 and 3.4.9 are only true up to the accuracy with which we solve the discrete LLG.

This is the tolerance of the linearisation method used if the problem is non-linear or the tolerance of the iterative solver if it is linear.

Additionally the derivations in Sections 3.4.8 and 3.4.9 assume that all fields are calculated at the midpoint without any additional approximations, *i.e.* that we are using a fully implicit (monolithic) method. In contrast to this, some models use an explicit approximation to the magnetostatic field in order to avoid a dense contribution due to the long-range magnetostatic interactions[15] in the non-linear solve. Unfortunately, in this case the energy properties will be lost due to the replacement of $\mathscr{H}\left[\frac{\mathbf{m}_{n+1}+\mathbf{m}_n}{2}\right]$ by an alternative approximation (in a similar way to the semi-implicit modifications of IMR). However, the magnetisation length property will be retained because it only relies on $\mathbf{m}$ itself being calculated implicitly, and not the effective field.

One way to allow a completely implicit treatment of the magnetostatic field without a dense contribution was explored by d'Aquino *et al.* [24], and is used by Nmag [30]. They used a quasi-Newton method where the dense contribution was simply left out of the Jacobian matrix. This greatly slows down the convergence of the non-linear solve: in the example given in the paper 10-14 iterations were needed (compared to ~ 2 with a full Newton-Raphson method). However, in the general case it is not clear that this quasi-Newton method will converge at all!

In Chapter 6 we describe these issues in more detail and demonstrate a novel efficient and robust solver allowing fully implicit magnetostatic calculations.

3.8 Conclusions

In this chapter we have described a number of numerical methods for each component of a dynamic micromagnetic solver. For the rest of this thesis we focus on the methods appropriate for modelling general geometries: the FEM for spatial discretisation, with implicit time integration methods to avoid issues with stiffness, and FEM/BEM magnetostatic calculations. We use a Newton-Raphson linearisation method because of the faster convergence and the lack of time step limitations. We aim to solve the resulting linear systems with preconditioned Krylov solvers for high computational efficiency.

[15] At least a dense sub-block arises from all accurate and commonly used magnetostatic calculation methods, in particular: FEM/BEM, FFT and FMM. See Section 3.3 for more details.

Chapter 4

A finite element method for the Landau-Lifshitz-Gilbert equation

In Section 4.1 we give a general introduction to the finite element method using the magnetostatic potential equation (2.14) in one spatial dimension as an example. Then in Sections 4.2 and 4.3 we extend this discussion to the full (non-linear) Landau-Lifshitz-Gilbert equation with magnetostatics in three spatial dimensions. The linearisation of the LLG equation is handled using the Newton-Raphson method (see Section 3.5.2). We also derive an analytical expression for the Jacobian matrix in Section 4.4.

Finally in Section 4.5 we show how the geometric integration properties of IMR (as discussed in Section 3.4) can be extended from the strong form case to the weak form case as used in the finite element method.

For the purposes of this chapter we will assume that suitable boundary conditions on the magnetostatic potential have been already been determined; the details of this process are discussed in Chapter 5.

4.1 Introduction to the finite element method

Before we can start we need some definitions: The function space $C^n(\Omega)$ is the set of all functions f such that f and its first n derivatives are continuous on Ω.

The function space $L^2(\Omega)$ consists of all functions g such that the norm $||g||_{L^2} =$

$\sqrt{\int_\Omega |g(x)|^2 \, \mathrm{d}\Omega}$ is finite, this can be thought of as the set of functions such that their squares are integrable.

The n-th Sobolev space, $\mathcal{H}^n(\Omega)$, is the space of all functions g such that g and all its first n derivatives are in $L^2(\Omega)$, *e.g.* for real valued functions on the interval $[0, 1]$ the second Sobolev space is

$$\mathcal{H}^2([0,1]) = \left\{ g : [0,1] \to \mathbb{R} \;\middle|\; g, \frac{\partial g}{\partial x}, \frac{\partial^2 g}{\partial x^2} \in L^2([0,1]) \right\}. \tag{4.1}$$

Note that in higher dimensions *all combinations* of derivatives in all directions (*e.g.* $\frac{\partial^2 \phi}{\partial x \partial y}$) must be integrable. Another notation sometimes used for Sobolev spaces is $W^{n,p}(\Omega)$, which denotes the set of functions g such that g and its first n derivatives are in $L^p(\Omega)$, *i.e.* the extension of $\mathcal{H}^n(\Omega)$ to other $L^p(\Omega)$ norms.

In a slight abuse of notation we sometimes write the Jacobian of a vector function $\mathbf{a}$ (with n values) with respect to the vector $\mathbf{b}$ (with m values) as

$$\frac{\partial \mathbf{a}(\mathbf{b}, \ldots)}{\partial \mathbf{b}} = \begin{pmatrix} \frac{\partial a_0}{\partial b_0} & \frac{\partial a_0}{\partial b_1} & \cdots & \frac{\partial a_0}{\partial b_m} \\ \frac{\partial a_1}{\partial b_0} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \frac{\partial a_n}{\partial b_0} & \cdots & \cdots & \frac{\partial a_n}{\partial b_m} \end{pmatrix}. \tag{4.2}$$

The following simple problem is used to illustrate the finite element method: Find $\phi(x) \in C^2(0, 1)$ such that

$$-\phi''(x) = f(x), \qquad x \in [0, 1], \tag{4.3}$$

with boundary conditions

$$\phi(0) = \phi_a, \quad \phi(1) = \phi_b, \tag{4.4}$$

and where $f(x) \in C^0([0, 1])$. Setting $f = -\nabla \cdot \mathbf{m} = -\frac{\partial m_x}{\partial x}$ results in the magnetostatic potential problem (2.14) with Dirichlet boundary conditions.

4.1.1 A weak formulation

The formulation of the problem in (4.3) (called a strong or classical formulation) is too restrictive for our purposes. It disallows some useful approximating functions and restricts the geometries that can be modelled [91, p. 14]. For "well behaved" functions and domains a *weak formulation* is equivalent but without these restrictions. The weak formulation of (4.3) is: find $\phi \in V_s$ such that

$$-\int_0^1 \phi'' v \, \mathrm{d}x = \int_0^1 f v \, \mathrm{d}x \qquad \forall v \in V_T, \tag{4.5}$$

where V_T is some appropriate space of *test functions*. We need $V_T \subseteq \mathcal{H}^0([0,1])$ and $V_s \subseteq \mathcal{H}^2([0,1])$ in order to ensure that the integrals in (4.5) are well defined. So as the solution space we choose

$$V_s = \left\{ \phi \in \mathcal{H}^2([0,1]) \,\middle|\, \phi(0) = \phi_a,\, \phi(1) = \phi_b \right\}, \tag{4.6}$$

i.e. the space of functions that are sufficiently smooth for the integrals to be finite and that satisfy the boundary conditions. Since the boundary conditions are automatically enforced by the choice of V_s we can choose test functions that are zero on the boundary, so an appropriate test function space is:

$$V_T = \left\{ v \in \mathcal{H}^0([0,1]) \,\middle|\, v(0) = 0,\; v(1) = 0 \right\}. \tag{4.7}$$

To see (in an intuitive sense) why (4.5) is equivalent to (4.3) consider the case when $f(x)+\phi''(x) > 0$ for some interval $X \subset (0,1)$, *i.e.* when the strong form (4.3) is not satisfied on X. Since (4.5) must hold for all (sufficiently smooth) functions in V_T we can choose a function which is non-zero only on X and (4.5) will fail. The rigorous extension of this argument also covers the case when $f(x) + \phi''(x)$ is non-zero on a set of measure zero (*e.g.* at a finite set of points).

The smoothness restrictions on the solutions of our weak form approximation can be further reduced by integrating the left hand side of (4.5) by parts to obtain

$$-\int_0^1 \phi'' v \, \mathrm{d}x = -[\phi' v]_0^1 + \int_0^1 \phi' v' \, \mathrm{d}x = \int_0^1 f v \, \mathrm{d}x. \tag{4.8}$$

Note that the square bracketed term in (4.8) is zero because the test functions, v, are set to zero on the boundary. Hence we are left with the problem: Find

$\phi \in V_S'$ such that

$$\int_0^1 \phi' v' \, \mathrm{d}x = \int_0^1 f v \, \mathrm{d}x \quad \forall v \in V_T', \tag{4.9}$$

where

$$\begin{aligned} V_S' &= \left\{ \phi \in \mathcal{H}^1([0,1]) \,\middle|\, \phi(0) = \phi_a,\, \phi(1) = \phi_b \right\}, \\ V_T' &= \left\{ \in \mathcal{H}^1([0,1]) \,\middle|\, v(0) = 0,\, v(1) = 0 \right\}. \end{aligned} \tag{4.10}$$

Note that our test and solution spaces are now identical apart from the boundary values.

An alternative class of boundary condition, called Neumann conditions, specify the values of ϕ' at the boundary. In this case we do not set the test functions to be zero on the boundary since the boundary *values* must still be determined. However, since we are given the value of ϕ', we can calculate the term $[\phi' v]_0^1$ of (4.8) directly.

4.1.2 Discretisation

Equation (4.9) is still continuous (*i.e.* the test and solution spaces are still infinite dimensional) and so it cannot be implemented as a computational method. To convert our continuous weak form to a discrete version we approximate the infinite dimensional test function space by an N dimensional subset $V_T^h \subset V_T'$ such that $V_T^h \to V_T'$ as $N \to \infty$. Since V_T^h is finite dimensional we can write

$$V_T^h = \left\{ v_h \,\middle|\, v_h = \sum_{n=0}^{N} \alpha_n \varphi_n,\, \alpha_n \in \mathbb{R} \right\}, \tag{4.11}$$

for some finite set of linearly independent test basis functions φ_n (with the obvious limitation on the Dirichlet boundary values). Note that Similarly, for $V_S^h \subset V_S'$:

$$V_S^h = \left\{ \phi_h \,\middle|\, \phi_h = \sum_{l=0}^{N_l} \tilde{\phi}_l \psi_l,\, \tilde{\phi}_l \in \mathbb{R} \right\}, \tag{4.12}$$

with the equivalent requirement when $N \to \infty$. We call ψ_l the solution basis functions but they are often known as the shape functions. Different choices of φ_n and ψ_l lead to different methods, the choice corresponding to the finite element method will be discussed in Section 4.1.3. For now we continue with a description of the conversion to a general discrete form.

Replacing v from (4.9) by φ_n gives an equivalent set of conditions

$$\int_0^1 \phi_h' \varphi_n' \, \mathrm{d}x = \int_0^1 f \varphi_n \, \mathrm{d}x \qquad n = 0, \ldots, N, \tag{4.13}$$

Since $\phi_h \in V_S^h$ we can replace ϕ_h by a linear combination of the solution basis functions, *i.e.*

$$\phi_h = \sum_{l=0}^{N_l} \tilde{\phi}_l \psi_l, \tag{4.14}$$

where $\tilde{\phi}_l \in \mathbb{R}$ are the unknown coefficients which define the approximate solution ϕ_h. Substituting (4.14) into (4.13) gives

$$\sum_{l=0}^{N_l} \tilde{\phi}_l \int_0^1 \psi_l' \varphi_n' \, \mathrm{d}x = \int_0^1 f \varphi_n \, \mathrm{d}x \qquad n = 0, \ldots, N. \tag{4.15}$$

The functions f, φ and ψ are known and so the integrals in (4.15) can be evaluated. We introduce a matrix A and a vector $\mathbf{b}$ containing the results of these integrals:

$$\begin{aligned} \mathrm{A}_{ln} &= \int_0^1 \psi_l' \varphi_n' \, \mathrm{d}x, \\ b_j &= \int_0^1 f \varphi_n \, \mathrm{d}x, \end{aligned} \tag{4.16}$$

and the problem is reduced to a system of linear equations. Writing $\tilde{\phi}$ for the vector of $\tilde{\phi}_l$ the system is

$$\mathrm{A}\tilde{\phi} = \mathbf{b}. \tag{4.17}$$

Assuming that A is non-singular this system can be solved to find $\tilde{\phi}$, which can then be substituted into (4.14) to give an approximation for $\phi(x)$.

4.1.3 Finite Elements in One Dimension

To create a usable implementation of the methodology derived in so far in this chapter we must need to define basis functions for the spaces V_T^h and V_S^h. We choose the test and solution basis functions to be the same except at the boundaries. This choice defines a Galerkin method and results in a symmetric matrix A [103, p. 215].

The aim now is to approximate an arbitrary function by a linear combination of a finite set of easy-to-work-with functions, φ_n. A good class of functions for this purpose is the polynomials: they are trivial to add, subtract and differentiate; and can be easily numerically integrated (see Section 4.1.4).

Additionally it is desirable that the final matrix A be sparse (to allow efficient linear algebra). For this to happen we need $\int_0^1 \psi_l' \varphi_n' \, \mathrm{d}x = 0$ for most pairs of l, n. An obvious way to achieve this is to have each basis function only overlap (in space) with a few other basis functions, for example by having each basis function non-zero only on a portion of the domain. Basis functions with this property are said to have "finite support".

These two ideas lead us to the use of a carefully constructed set of *piecewise polynomials* as our basis functions. We first split the domain into N_e non-overlapping regions called *elements*, each with a *node* at either end. We then define two "local" basis functions within each element as

$$\varphi_0^{(e)} = \frac{x - x_1^{(e)}}{x_0^{(e)} - x_1^{(e)}}, \qquad \varphi_1^{(e)} = \frac{x - x_0^{(e)}}{x_1^{(e)} - x_0^{(e)}}, \tag{4.18}$$

where $x_0^{(e)}$ and $x_1^{(e)}$ are the positions of the nodes of element e. This definition results in following useful properties of $\varphi_n^{(e)}(x)$:

$$\varphi_n^{(e)}(x_n) = \begin{cases} 1 & k = n, \\ 0 & k \neq n. \end{cases} \tag{4.19}$$

This means that the coefficient of each basis function in (4.14) corresponds directly to the value of the solution at the node where that test function has its maximum value. In other words the list of coefficients, $\tilde{\phi}$, is just a list of the nodal values of $\phi(x)$.

Next we define the nth (global) basis function as:

$$\varphi_n(x) = \begin{cases} \varphi_n^{(e)}(x) & \text{if } x \in [x_0^{(e)}, x_1^{(e)}] \text{ and node } n \text{ is in element } e, \\ 0 & \text{otherwise.} \end{cases} \tag{4.20}$$

So each global basis function is only non-zero on the elements neighbouring the node where it has a maximum value. We call the set of such polynomial basis functions $S^1(\Omega)$. The relationship between the local and global test functions is

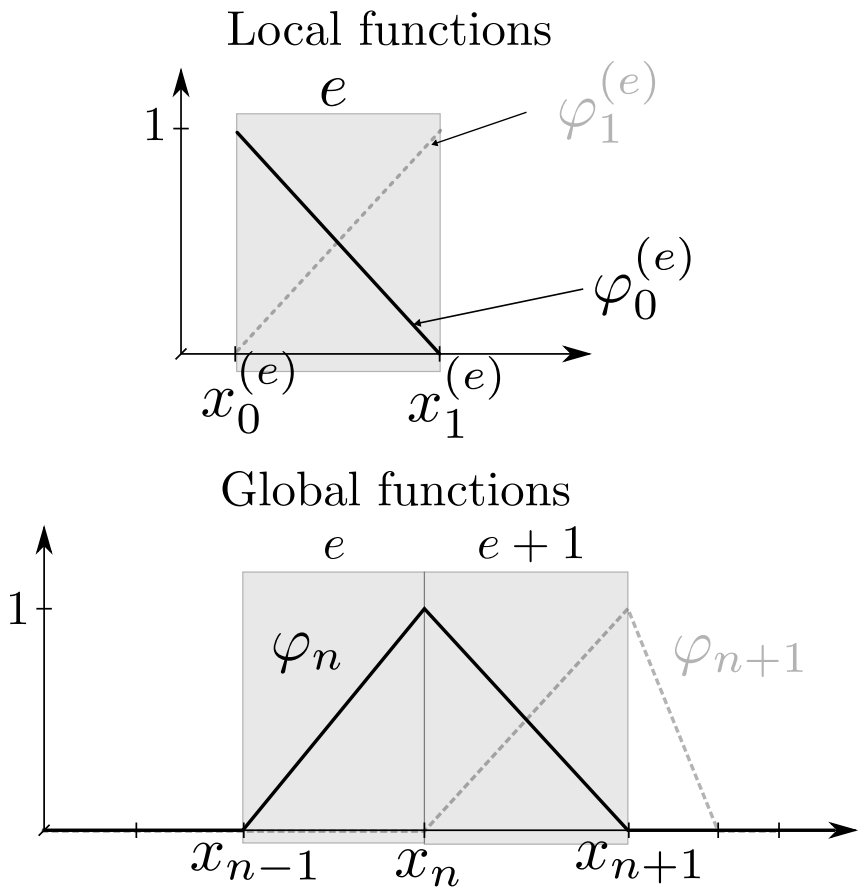

Figure 4.1: The local and global piecewise polynomial basis functions and numbering schemes for a one-dimensional finite element method.

illustrated in Figure 4.1.

It is advantageous to work with such low order polynomials even for very complex models because the evaluation of integrals and interpolated values is simple and cheap to compute. Also note that an arbitrary level of accuracy can be achieved when approximating a smooth function by piecewise polynomials by increasing the number of polynomials used (*i.e.* by adding more nodes and elements to the mesh in appropriate places).

The description here generalises to higher dimensions fairly easily [91, p. 20]. In 2D square or triangle elements are typically used and nodes are placed at every corner of each element. In 3D "brick" or tetrahedral elements are used in the same way. In both cases the polynomial basis functions are replaced by higher dimensional ones with equivalent properties.

The method can also be generalised to higher order polynomials, although more

nodes are required inside each element in order to provide enough values to fully define the polynomials, the use of higher order polynomials increases the order of accuracy. However there is a trade-off: the resulting linear systems become more dense as the order of the polynomials is increased due to more basis functions overlapping with each other.

To calculate the matrix A and the vector $\mathbf{b}$ from (4.16) we first calculate the local contributions from each element: $\mathrm{A}^{(e)}$ and $\mathbf{b}^{(e)}$. We then assemble the global A and $\mathbf{b}$ by summing the appropriate local contributions. For our one-dimensional Poisson example the calculation of $\mathrm{A}^{(e)}$ is quite simple: substituting the two local basis functions (4.18) into (4.16) and writing $h = x_1^{(e)} - x_0^{(e)}$ we are left with

$$\mathrm{A}^{(e)} = \frac{1}{h}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}, \tag{4.21}$$

where h is the element size, $h = x_1^{(e)} - x_0^{(e)}$ (note that h can vary between elements). Similarly for $\mathbf{b}^{(e)}$ we obtain

$$\mathbf{b}^{(e)} = \frac{1}{h}\begin{pmatrix} -\int_{x_0^{(e)}}^{x_1^{(e)}} (x - x_1^{(e)})\, f(x)\, dx \\ \int_{x_0^{(e)}}^{x_1^{(e)}} (x - x_0^{(e)})\, f(x)\, dx \end{pmatrix} \tag{4.22}$$

which we compute numerically since the entries depend on $f(x)$.

4.1.4 Numerical evaluation of integrals

Finally we need a way to evaluate the required integrals computationally. The most flexible way to do this to use a numerical quadrature scheme. A quadrature scheme is an approximation to an integral of the form

$$\int_\Omega f(\mathbf{x})\, \mathrm{d}\mathbf{x} \approx \sum_i w_i f(\mathbf{x}_i), \tag{4.23}$$

where $\mathbf{x}_i$ are the *knots* (or evaluation points) and w_i the weights. The choice of $\mathbf{x}_i$ and w_i defines the quadrature scheme.

There exist many quadrature schemes suited for different applications. Since we only need to evaluate integrals of polynomial functions a single simple and accurate class of quadrature is sufficient: Gaussian quadrature [59, p. 492]. A

Gaussian quadrature scheme using n knots is able to exactly evaluate integrals of polynomials up to order $2n+1$. This is the highest possible polynomial order of exact evaluation for a quadrature scheme.

In higher dimensions the integration is not quite so simple. Typically a quadrature scheme only applies for integration over a specific shape of element (or at least the scheme must be non-trivially modified for alternative geometries). To get around this issue we use a coordinate transformation to a reference element [91, p. 29], *e.g.* in two dimensions any four sided element is mapped to the square $[-1,1]\times[-1,1]$, meaning that standard quadrature schemes can be used.

We call the coordinates on the reference element the "local coordinates" and denote them as $\mathbf{s}$, we call the original coordinates the "global coordinates". We use a transformation (from local to global coordinates) that can be written similarly to (4.14):

$$\mathbf{x}(\mathbf{s}) = \sum_j \mathbf{x}_j \psi_j(\mathbf{s}), \tag{4.24}$$

this is known as an isoparametric transformation. The final difficulty is to calculate the Jacobian of this transformation:

$$J_e = \det \frac{\partial \mathbf{s}}{\partial \mathbf{x}} = \frac{1}{\det \frac{\partial \mathbf{x}}{\partial \mathbf{s}}}, \tag{4.25}$$

where we have used the notation defined in (4.2) for the Jacobian of derivatives of a function. Note that the elements of the Jacobian matrix $\frac{\partial \mathbf{x}}{\partial \mathbf{s}}$ can be calculated by simply differentiating (4.24). When using linear polynomial basis functions the Jacobian is constant for 1D line elements, 2D triangular elements and 3D tetrahedral elements.

So the final integration scheme is

$$\begin{aligned} \int_{\Omega_e} f(\mathbf{x})\,\mathrm{d}\mathbf{x} &= \int_{\Omega_e} f(\mathbf{x}(\mathbf{s}))J_e(\mathbf{s})\,\mathrm{d}\mathbf{s}, \\ &\approx \sum_{i=0}^{N} w_i f(\mathbf{x}(\mathbf{s}_i))J_e(\mathbf{s}_i). \end{aligned} \tag{4.26}$$

Gaussian quadratures for square and cube elements can be constructed from the one dimensional quadrature using a tensor product formula. Similar quadratures for triangular and tetrahedral elements exist *e.g.* [27, 80].

4.2 The FEM applied to the LLG equation

We now apply the finite element method as discussed in Section 4.1, together with the Newton-Raphson method discussed in Section 3.5.2, to the solution of the Landau-Lifshitz-Gilbert equation. We use the Gilbert form of the LLG, (2.22), with the effective fields described in Section 2.2 and use the potential method described in Section 3.3.2 to calculate the magnetostatic field.

We choose the Gilbert form of the LLG even though it is less intuitive than the Landau-Lifshitz form because it only contains single cross product terms, reducing the complexity of derivations. A side effect of this choice is that explicit time integration schemes cannot be used since the time derivative is only defined implicitly.

4.2.1 The strong form

After non-dimensionalisation (see Section 2.4.1) the strong form of our system of equations is given by

$$\begin{aligned} \frac{\partial \mathbf{m}}{\partial t} &= -\mathbf{m} \times \mathbf{h} + \alpha \mathbf{m} \times \frac{\partial \mathbf{m}}{\partial t}, \\ \mathbf{h} &= \mathbf{h}_{\text{ap}} - \nabla \phi + \nabla^2 \mathbf{m} + \mathcal{K}_1 (\mathbf{m} \cdot \hat{\mathbf{e}}) \hat{\mathbf{e}}, \\ \nabla^2 \phi &= \nabla \cdot \mathbf{m}. \end{aligned} \tag{4.27}$$

We prefer the vector notation used here over tensor (*i.e.* subscript) notation throughout this thesis because of the common occurrence of cross-products and the complexity of representing a cross-product as a dummy summation over the Levi-Civita tensor.

Recall that we are using the Newton-Raphson method to find a solution to the non-linear problem. Hence we need to rearrange (4.27) into the residual form

$$\boldsymbol{\mathcal{R}}(\mathbf{m}, \phi) = \frac{\partial \mathbf{m}}{\partial t} + \mathbf{m} \times \mathbf{h} - \alpha \mathbf{m} \times \frac{\partial \mathbf{m}}{\partial t}, \tag{4.28}$$

$$\mathcal{S}(\mathbf{m}, \phi) = \nabla^2 \phi - \nabla \cdot \mathbf{m}, \tag{4.29}$$

where $\mathbf{h}$ is as in (4.27). A pair of functions $\mathbf{m} \in (C^2(\Omega))^3$, $\phi \in C^2(\Omega)$ is a solution to (4.28) and (4.29) if $\boldsymbol{\mathcal{R}}(\mathbf{m}, \phi) = 0$ and $\mathcal{S}(\mathbf{m}, \phi) = 0$ for all $\mathbf{x} \in \Omega$.

For now we consider the magnetostatic potential only within the magnetic domain Ω with Neumann or Dirichlet boundary conditions on the boundary Γ. Details of the extension to include the effects of the external region using the FEM/BEM method are given in Chapter 5, for which both the Dirichlet and Neumann cases will be required. We define $\Gamma_{\mathcal{D}}$ and $\Gamma_{\mathcal{N}}$ respectively to represent the part of the boundary where a Dirichlet condition or Neumann is imposed on the potential,[1] *i.e.*

$$\begin{aligned} \phi(\mathbf{x}) &= g_{\mathcal{D}}(\mathbf{x}) \quad \mathbf{x} \in \Gamma_{\mathcal{D}}, \\ \nabla\phi(\mathbf{x}) \cdot \hat{\mathbf{n}}(\mathbf{x}) &= g_{\mathcal{N}}(\mathbf{x}) \quad \mathbf{x} \in \Gamma_{\mathcal{N}}, \end{aligned} \tag{4.30}$$

where $\hat{\mathbf{n}}(\mathbf{x})$ is the outward unit normal. If the boundary conditions are purely Neumann then this problem is actually singular: the solution can vary by an arbitrary constant. This constant has no effect on the magnetostatic field and so the singularity can be dealt with, for example, by fixing the value of ϕ to zero at some point in the domain.

From (2.37) we have that the boundary condition on the magnetisation is

$$\mathbf{m} \times \frac{\partial \mathbf{m}}{\partial \hat{\mathbf{n}}} = \mathbf{0} \quad \mathbf{x} \in \Gamma. \tag{4.31}$$

4.2.2 The weak form

We now convert (4.28) and (4.29) into the residual weak form as described in Section 4.1.1. In the following we will need the identity[2]

$$(\nabla^2\phi)v = \nabla \cdot (v\nabla\phi) - \nabla\phi \cdot \nabla v. \tag{4.32}$$

We will also make use of the divergence theorem

$$\int_\Omega \nabla \cdot \mathbf{f} \, \mathrm{d}\Omega = \int_\Gamma \mathbf{f} \cdot \hat{\mathbf{n}} \, \mathrm{d}\Gamma. \tag{4.33}$$

Multiplying (4.29) by a test function and integrating over the magnetic domain we obtain the weak form residual for the magnetostatic potential

$$s(\phi, v) = \int_\Omega (\nabla^2\phi)v \, \mathrm{d}\Omega - \int_\Omega (\nabla \cdot \mathbf{m})v \, \mathrm{d}\Omega, \tag{4.34}$$

[1] These must be such that $\Gamma_{\mathcal{N}} \cap \Gamma_{\mathcal{D}} = \emptyset$ and $\Gamma_{\mathcal{N}} \cup \Gamma_{\mathcal{D}} = \Gamma$.
[2] This can be easily derived by applying the product rule to $\nabla \cdot (v\nabla\phi)$.

where $\phi \in V_{S\phi} = \{\phi \in \mathcal{H}^2(\Omega) \,|\, \phi(\mathbf{x}) = g_{\mathcal{D}}(\mathbf{x}) \,\forall \mathbf{x} \in \Gamma_{\mathcal{D}}\}$ and $\mathbf{m} \in \mathcal{H}^1(\Omega)$. Recall that aim is to find some ϕ such that $s(\phi, v) = 0$ for all test functions $v \in V_T = \{\phi \in \mathcal{H}^0(\Omega) \,|\, \phi(\mathbf{x}) = 0 \,\forall \mathbf{x} \in \Gamma_{\mathcal{D}}\}$. Note that while the use of the Newton-Raphson method is unnecessary for the magnetostatic potential alone (since (4.34) is linear in ϕ) it will be necessary later for the coupled problem.

We now reduce the order of the derivatives on ϕ, as discussed in Section 4.1.1, by "transferring" the derivatives onto the test functions. Integrating (4.32) over the domain and applying the divergence theorem (4.33) we obtain

$$\begin{aligned} \int_\Omega (\nabla^2 \phi) v \, \mathrm{d}\Omega &= \int_\Omega \nabla \cdot (v \nabla \phi) \, \mathrm{d}\Omega - \int_\Omega \nabla\phi \cdot \nabla v \, \mathrm{d}\Omega, \\ &= \int_\Gamma v (\nabla \phi \cdot \hat{\mathbf{n}}) \, \mathrm{d}\Gamma - \int_\Omega \nabla\phi \cdot \nabla v \, \mathrm{d}\Omega. \end{aligned} \tag{4.35}$$

Substituting (4.35) into (4.34) gives

$$s(\phi, v) = \int_\Gamma v (\nabla \phi \cdot \hat{\mathbf{n}}) \, \mathrm{d}\Gamma - \int_\Omega \nabla v \cdot \nabla \phi \, \mathrm{d}\Omega - \int_\Omega (\nabla \cdot \mathbf{m}) v \, \mathrm{d}\Omega, \tag{4.36}$$

where the function spaces are now $V'_{S\phi} = \{\phi \in \mathcal{H}^1(\Omega) \,|\, \phi(\mathbf{x}) = g_{\mathcal{D}}(\mathbf{x}) \,\forall \mathbf{x} \in \Gamma_{\mathcal{D}}\}$ and $V'_T = \{\phi \in \mathcal{H}^1(\Omega) \,|\, \phi(\mathbf{x}) = 0 \,\forall \mathbf{x} \in \Gamma_{\mathcal{D}}\}$.

Note that the boundary integral is always zero on $\Gamma_{\mathcal{D}}$ by the definition of the test functions. Hence the boundary integral is only non-zero on $\Gamma_{\mathcal{N}}$ where $\nabla\phi \cdot \hat{\mathbf{n}} = g_{\mathcal{N}}$ and we have

$$s(\phi, v) = \int_{\Gamma_{\mathcal{N}}} v g_{\mathcal{N}} \, \mathrm{d}\Gamma - \int_\Omega \nabla v \cdot \nabla \phi \, \mathrm{d}\Omega - \int_\Omega (\nabla \cdot \mathbf{m}) v \, \mathrm{d}\Omega. \tag{4.37}$$

Next we apply the same procedure to the Landau-Lifshitz-Gilbert residual (4.28). For now we sidestep the details of time discretisation by assuming $\frac{\partial \mathbf{m}}{\partial t}$ to be just another (sufficiently smooth) function of $\mathbf{x}$. The weak form of the LLG residual is

$$\begin{aligned} \mathbf{r}(\mathbf{m}, v) = \int_\Omega \Big(&\frac{\partial \mathbf{m}}{\partial t} + (\mathbf{m} \times \mathbf{h}_{\mathrm{ca}}) + (\mathbf{m} \times \mathbf{h}_{\mathrm{ap}}) \\ &- (\mathbf{m} \times \nabla\phi) + (\mathbf{m} \times \nabla^2 \mathbf{m}) - \alpha \left(\mathbf{m} \times \frac{\partial \mathbf{m}}{\partial t} \right) \Big) v \, \mathrm{d}\Omega. \end{aligned} \tag{4.38}$$

The problem is solved by $\mathbf{m} \in \mathcal{H}^2(\Omega)$ such that $\mathbf{r}(\mathbf{m}, v) = \mathbf{0} \;\forall v \in \mathcal{H}^0(\Omega)$.

Note that in (4.38) the residual is a vector. However, we can also express the requirement for each component of (4.38) to be zero as

$$r(\mathbf{m}, \mathbf{v}) = \int_\Omega \Big(\dots \Big) \cdot \mathbf{v} \, \mathrm{d}\Omega = 0, \tag{4.39}$$

where $\mathbf{v} \in (\mathcal{H}^0(\Omega))^3$ and the bracketed term is exactly the same as in (4.38). To see that $r(\mathbf{m}, \mathbf{v}) = 0 \; \forall \mathbf{v} \in (\mathcal{H}^0(\Omega))^3$ implies $\mathbf{r}(\mathbf{m}, v) = \mathbf{0} \; \forall v \in \mathcal{H}^0(\Omega)$ consider the vector test function $(v, 0, 0)$. Clearly $r(\mathbf{m}, (v, 0, 0)) = [\mathbf{r}(\mathbf{m}, v)]_x$, and similarly for the y and z cases. To see the inverse relationship note that if $\mathbf{r}(\mathbf{m}, v) = \mathbf{0} \; \forall v \in \mathcal{H}^0(\Omega)$ then each term of every dot product in $r(\mathbf{m}, \mathbf{v})$ is zero, implying that the total residual is zero $\forall \mathbf{v} \in (\mathcal{H}^0(\Omega))^3$. The use of vector test functions makes certain manipulations simpler so, for the remainder of this section, we will work with vector test functions.

We again wish to reduce the order of the required derivatives, this time on $\mathbf{m}$ in the exchange term. First we rearrange to find that

$$\begin{aligned} I_{\text{ex}} &= \int_\Omega \nabla^2 \mathbf{m} \cdot (\mathbf{v} \times \mathbf{m}) \, \mathrm{d}\Omega, \\ &= \int_\Omega \nabla^2 \mathbf{m} \cdot \mathbf{b} \, \mathrm{d}\Omega, \\ &= \sum_{i=1}^{3} \int_\Omega b_i (\nabla^2 m_i) \, \mathrm{d}\Omega, \end{aligned} \tag{4.40}$$

where $\mathbf{b} = (\mathbf{v} \times \mathbf{m})$. Note that each term of the sum in (4.40) is very similar to (4.34), and so we follow the same procedure to obtain

$$I_{\text{ex}} = \sum_{i=1}^{3} \left[\int_\Gamma b_i (\nabla m_i \cdot \hat{\mathbf{n}}) \, \mathrm{d}\Gamma - \int_\Omega \nabla b_i \cdot \nabla m_i \, \mathrm{d}\Omega \right]. \tag{4.41}$$

Rearranging the first term of (4.41) we find that

$$\begin{aligned} \sum_{i=1}^{3} \int_\Gamma b_i (\nabla m_i \cdot \hat{\mathbf{n}}) \, \mathrm{d}\Gamma &= \int_\Gamma \mathbf{b} \cdot \frac{\partial \mathbf{m}}{\partial \hat{\mathbf{n}}} \, \mathrm{d}\Gamma, \\ &= \int_\Gamma (\mathbf{v} \times \mathbf{m}) \cdot \frac{\partial \mathbf{m}}{\partial \hat{\mathbf{n}}} \, \mathrm{d}\Gamma, \\ &= \int_\Gamma (\mathbf{m} \times \frac{\partial \mathbf{m}}{\partial \hat{\mathbf{n}}}) \cdot \mathbf{v} \, \mathrm{d}\Gamma, \end{aligned} \tag{4.42}$$

which is zero by the Neumann-like boundary condition on $\mathbf{m}$, (4.31).

Alternatively if we have periodic boundary conditions we can write this integral as a sum over opposite boundaries Γ_j and $\hat{\Gamma}_j$. On opposite boundaries $\hat{\mathbf{n}}$ becomes $-\hat{\mathbf{n}}$ but the geometry and $\mathbf{m}$ values must be identical, so we have

$$\begin{aligned}\int_\Gamma (\mathbf{m}\times\frac{\partial\mathbf{m}}{\partial\hat{\mathbf{n}}})\cdot\mathbf{v}\,\mathrm{d}\Gamma &= \sum_j\left[\int_{\Gamma_j}\mathbf{m}\times\frac{\partial\mathbf{m}}{\partial\hat{\mathbf{n}}}\,\mathrm{d}\Gamma_j+\int_{\hat{\Gamma}_j}\mathbf{m}\times\frac{\partial\mathbf{m}}{\partial(-\hat{\mathbf{n}})}\,\mathrm{d}\hat{\Gamma}_j\right],\\ &= \sum_j\quad 0,\end{aligned} \tag{4.43}$$

and the result is identical to the Neumann condition.

Next we need an expression for ∇b_i. To avoid the requirement for tensor index notation it is helpful to define, for a vector $\mathbf{a}$

$$\widetilde{\nabla}\mathbf{a} = \begin{pmatrix}\nabla a_0\\ \nabla a_1\\ \nabla a_2\end{pmatrix}. \tag{4.44}$$

Then using the product rule, we have

$$\begin{aligned}\widetilde{\nabla}\mathbf{b} &= \widetilde{\nabla}(\mathbf{v}\times\mathbf{m}),\\ &= (\widetilde{\nabla}\mathbf{v})\times\mathbf{m}+\mathbf{v}\times(\widetilde{\nabla}\mathbf{m}).\end{aligned} \tag{4.45}$$

Using this notation we can write (4.41) as

$$I_{\mathrm{ex}} = -\int_\Omega\widetilde{\nabla}\mathbf{m}:\left(\widetilde{\nabla}\mathbf{v}\times\mathbf{m}\right)\mathrm{d}\Omega-\int_\Omega\widetilde{\nabla}\mathbf{m}:\left(\mathbf{v}\times\widetilde{\nabla}\mathbf{m}\right)\mathrm{d}\Omega, \tag{4.46}$$

where ":" is the component-wise scalar product, *i.e.* $\widetilde{\nabla}\mathbf{a}:\widetilde{\nabla}\mathbf{c} = \nabla a_0\cdot\nabla b_0+\nabla a_1\cdot\nabla b_1+\nabla a_2\cdot\nabla b_2$. Note that the ":" operator obeys the usual triple product identity

$$\mathbf{a}:(\mathbf{a}\times\mathbf{b}) = \mathbf{0}. \tag{4.47}$$

Hence the second term of (4.46) is zero and after reordering the first term we are left with

$$\int_\Omega(\mathbf{m}\times\nabla^2\mathbf{m})\cdot\mathbf{v}\,\mathrm{d}\Omega = -\int_\Omega\left(\mathbf{m}\times\widetilde{\nabla}\mathbf{m}\right):\widetilde{\nabla}\mathbf{v}\,\mathrm{d}\Omega. \tag{4.48}$$

The final residual for the LLG equation, using a vector test function, is

$$r(\mathbf{m},\mathbf{v}) = \int_\Omega \frac{\partial \mathbf{m}}{\partial t}\cdot\mathbf{v} + (\mathbf{m}\times\mathbf{h}_{\text{ca}})\cdot\mathbf{v} + (\mathbf{m}\times\mathbf{h}_{\text{ap}})\cdot\mathbf{v} - (\mathbf{m}\times\nabla\phi)\cdot\mathbf{v} - \left(\mathbf{m}\times\widetilde{\nabla}\mathbf{m}\right) : \widetilde{\nabla}\mathbf{v} - \alpha\left(\mathbf{m}\times\frac{\partial\mathbf{m}}{\partial t}\right)\cdot\mathbf{v}\,\mathrm{d}\Omega. \tag{4.49}$$

The final weak formulation is:

> Find $\mathbf{m} \in (\mathcal{H}^1(\Omega))^3$, $\phi \in V'_{S\phi}$ such that $r(\mathbf{m},\mathbf{v}) = 0$ for all $\mathbf{v} \in (\mathcal{H}^1(\Omega))^3$ and $s(\phi, v) = 0$ for all $v \in V'_T$. (4.50)

4.2.3 Spatial Discretisation

The next step is to choose a finite dimensional approximation for the infinite dimensional test and solution function spaces such that the residuals (4.37) and (4.49) become discrete in space. Just as in Section 4.1.3 we choose to represent the finite dimensional spaces using a set of piecewise linear polynomials, $S^1(\Omega)$, as basis functions.

The approximations to the unknown functions $\mathbf{m}$ and ϕ are represented as a sum over the solution basis functions $\psi_k \in S^1(\Omega)$ (or, for ϕ, $\psi_k \in S^1_\phi(\Omega) = \{\psi_k \in S^1(\Omega) \,|\, \psi_k(\mathbf{x}) = 0 \forall \mathbf{x} \in \Gamma_{\mathcal{D}}\}$) as

$$\begin{aligned}\mathbf{m}(\mathbf{x}) \approx \mathbf{m}_h &= \sum_{k=0}^{N} \psi_k\,\tilde{\mathbf{m}}_k,\\ \phi(\mathbf{x}) \approx \phi_h &= \sum_{k=0}^{N} \psi_k\,\phi_k.\end{aligned} \tag{4.51}$$

Similarly the approximations to the test functions are represented as a sum over the test basis functions, $\varphi_n \in S^1(\Omega)$ (or $S^1_\phi(\Omega)$ as applicable), as

$$v \approx v_h = \sum_{n=0}^{N} \varphi_n\,a_n. \tag{4.52}$$

Note that in our method $\psi_k \equiv \varphi_k$ except for the basis functions of ϕ at the Dirichlet boundary. Despite this we continue to write the test and solution basis functions differently for generality.

So substituting the basis representations, (4.51) and (4.52) into the continuous residuals we obtain approximations for the residuals in terms of a finite number of nodal values. Next we replace the requirement that the residuals be zero for all test functions in the infinite dimensional set $\mathcal{H}^1(\Omega)$ with the equivalent requirement for all the test basis functions in the finite set $S^1(\Omega)$. This results in a finite number of conditions (the residuals) on a finite number of variables (the nodal values) and the spatial discretisation is complete.

4.3 Time Discretisation

In this section we discuss how to deal with the time derivative in the LLG residual (4.49). As discussed in Chapter 3, we will apply standard integration methods widely used for ODEs. This technique of applying independent spatial and time discretisations is known as semi-discretisation (or the method of lines).

However, we cannot simply apply the time integration methods in the way they are written in Section 3.4.2 (and in most text books) because our LLG residual (4.49) is an *implicit* function of the time derivative and so cannot be written in the form $\frac{\partial \mathbf{y}}{\partial t} = \mathbf{f}(t, \mathbf{y})$. Instead we rearrange the time integration method to give discrete substitutions for the time derivative, the unknown function $\mathbf{y}$ and t.

Let Δ_n be the size of the nth time step and x_n denote value of x at the n-th time-step.[3] The simplest case is the first order backwards difference method (BDF1), which is defined by the substitution

$$\frac{\partial \mathbf{y}}{\partial t} \to \frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{\Delta_n}, \quad \mathbf{y} \to \mathbf{y}_{n+1}, \quad t \to t_{n+1}. \tag{4.53}$$

This substitution can be obtained by simple algebraic rearrangements of (3.7). Similarly, the second order backwards difference method (BDF2) can be found (with some more complex algebra) to be:

$$\frac{\partial \mathbf{y}}{\partial t} \to \mathbf{y}_{n+1}\left(\frac{1}{\Delta_n} + \frac{1}{\Delta_n + \Delta_{n-1}}\right) - \mathbf{y}_n \frac{\Delta_n + \Delta_{n-1}}{\Delta_n \Delta_{n-1}} + \mathbf{y}_{n-1}\frac{\Delta_n}{(\Delta_n + \Delta_{n-1})\Delta_{n-1}},$$
$$\mathbf{y} \to \mathbf{y}_{n+1}, \quad t \to t_{n+1}. \tag{4.54}$$

[3]This notation has some potential to be confused with the nodal value notation of Section 4.2.3 but the meaning will be clear from the context.

The implicit midpoint rule (IMR) is defined by the substitutions

$$\frac{\partial \mathbf{y}}{\partial t} \to \frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{\Delta_n}, \quad \mathbf{y} \to \frac{\mathbf{y}_{n+1} + \mathbf{y}_n}{2}, \quad t \to \frac{t_{n+1} + t_n}{2}. \tag{4.55}$$

Alternatively, as mentioned in Section 3.4.2, we can define the implicit midpoint rule in terms of a step of BDF1 with $\Delta_n^{\mathrm{BDF1}} = \Delta_n^{\mathrm{IMR}}/2$ to obtain $\mathbf{y}_{n+\frac{1}{2}}$ followed by the algebraic update

$$\mathbf{y}_{n+1} = 2\mathbf{y}_{n+\frac{1}{2}} - \mathbf{y}_n. \tag{4.56}$$

The trapezoid rule (TR) is defined (recursively) by

$$\begin{aligned} \frac{\partial \mathbf{y}}{\partial t} \to \left.\frac{\partial \mathbf{y}}{\partial t}\right|_{n+1} &= \frac{2(\mathbf{y}_{n+1} - \mathbf{y}_n)}{\Delta_n} - \left.\frac{\partial \mathbf{y}}{\partial t}\right|_n, \\ \mathbf{y} \to \mathbf{y}_{n+1}, \quad t &\to t_{n+1}. \end{aligned} \tag{4.57}$$

By applying one of the substitutions (4.53)-(4.57) to the system of spatially discretised residuals we obtain a fully discrete problem. The properties of these time discretisations are exactly as discussed in Section 3.4.

4.3.1 The fully discretised residuals

Here we write the residual using the *scalar* test function form because it is the form most suitable for use in a computer implementation. We define the effect of $\widetilde{\nabla}$ on a scalar function φ to be $\widetilde{\nabla}\varphi = (\nabla\varphi, \nabla\varphi, \nabla\varphi)$, this allows us to use scalar and vector test functions in the same way in the following.

We write $\frac{\partial \mathbf{m}_h}{\partial t}$ for the fully discrete time derivative approximations at step $n+1$; and $\mathbf{m}_h$, ϕ_h for the fully discrete approximations to $\mathbf{m}$, ϕ as required for the time derivative.[4] Then the fully discrete residuals are

$$\begin{aligned} \mathbf{r}_n = \int_\Omega \frac{\partial \mathbf{m}_h}{\partial t}\varphi_n + \mathbf{m}_h \times \left(\mathbf{h}_{\mathrm{ap}} + \mathbf{h}_{\mathrm{ca}} - \nabla\phi - \alpha\frac{\partial \mathbf{m}_h}{\partial t}\right)\varphi_n \\ - (\mathbf{m}_h \times \widetilde{\nabla}\mathbf{m}_h) : \widetilde{\nabla}\varphi_n)\,\mathrm{d}\Omega, \end{aligned} \tag{4.58}$$

[4]These are given by (4.53)–(4.57) as appropriate, with the $\mathbf{y}_i$ values replaced by the spatially discrete approximations defined in (4.51).

and

$$s_n = \int_{\Gamma_{\mathcal{N}}} \varphi_n g_{\mathcal{N}} \, \mathrm{d}\Gamma - \int_\Omega \nabla\varphi_n \cdot \nabla\phi_h \, \mathrm{d}\Omega - \int_\Omega (\nabla \cdot \mathbf{m}_h)\varphi_n \, \mathrm{d}\Omega. \tag{4.59}$$

for each test basis function φ_n.

4.4 The Newton-Raphson Jacobian

To solve our fully discrete system by the Newton-Raphson method (see Section 3.5.2) we also need to know the Jacobian matrix of each residual differentiated with respect to each variable at the $(n+1)$th time step. This can be handled by numerical differentiation (*e.g.* by approximating each derivative using a finite difference method), but analytical Jacobians are typically faster to compute and give better convergence properties than such numerical approximations. Additionally, for the solution of the linear systems, it is often useful to know analytically the block structure of the system being solved.

Note that the "real" implementation of the implicit midpoint rule (4.55) introduces a factor of $1/2$ in various places due to the substitution $\mathbf{y} \to \frac{\mathbf{y}_{n+1}+\mathbf{y}_n}{2}$. For simplicity, and because we prefer the simpler BDF1-based implementation, we do not include this factor in the derivations below.

Throughout this section we use the notation of $\frac{\partial \mathbf{a}(\mathbf{b},\ldots)}{\partial \mathbf{b}}$ for the Jacobian of derivatives of $\mathbf{a}$ with respect to $\mathbf{b}$ as shown in (4.2).

We first note that the effect of differentiation by a nodal value on the approximation function ϕ_h is quite simple:

$$\begin{aligned} \frac{\partial \phi_h}{\partial \tilde{\phi}_l} &= \frac{\partial}{\partial \tilde{\phi}_l}\left[\sum_k \tilde{\phi}_k \psi_k\right] = \psi_l, \\ \frac{\partial \phi_h}{\partial \tilde{\mathbf{m}}_l} &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l}\left[\sum_k \tilde{\phi}_k \psi_k\right] = \mathbf{0}^T, \end{aligned} \tag{4.60}$$

and similarly for the interpolated magnetisation, $\mathbf{m}_h$:

$$\begin{aligned}\frac{\partial \mathbf{m}_h}{\partial \tilde{\mathbf{m}}_l} &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l}\left[\sum_k \tilde{\mathbf{m}}_k \psi_k\right] = \psi_l \mathrm{I}_3, \\ \frac{\partial \mathbf{m}_h}{\partial \tilde{\phi}_l} &= \frac{\partial}{\partial \tilde{\phi}_l}\left[\sum_k \tilde{\mathbf{m}}_k \psi_k\right] = \mathbf{0},\end{aligned} \tag{4.61}$$

where I_3 is the 3×3 identity matrix.

4.4.1 Poisson Jacobian

Starting from the Poisson residual, (4.37), and omitting terms that contain no dependence on ϕ_i we get

$$\begin{aligned}\mathrm{A}_{n,l} &= \frac{\partial}{\partial \tilde{\phi}_l} \sum_k \int_\Omega -(\nabla \varphi_n \cdot \nabla \psi_k)\phi_k \,\mathrm{d}\Omega, \\ &= -\int_\Omega \nabla \varphi_n \cdot \nabla \psi_l \,\mathrm{d}\Omega.\end{aligned} \tag{4.62}$$

This block of the Jacobian is comparatively simple because the Poisson equation is linear. It corresponds to the well known discrete Laplacian operator [91, p. 57].

4.4.2 LLG Jacobian

Unfortunately in most of the Jacobian calculations we have multiple terms depending on $\mathbf{m}_h$ joined together by a cross product. The process of differentiating these terms can be made much easier by making use of the "skew operator" which represents a cross product as a matrix-vector multiplication. For $\mathbf{a} = (a_0, a_1, a_2)^T$ the skew operator is given by

$$\Lambda\left[\mathbf{a}\right] = \mathrm{skew}(\mathbf{a}) = \begin{pmatrix} 0 & -a_2 & a_1 \\ a_2 & 0 & -a_0 \\ -a_1 & a_0 & 0 \end{pmatrix}. \tag{4.63}$$

Note that

$$\Lambda\left[\mathbf{a}\right]\cdot\mathbf{b}=\begin{pmatrix}0 & -a_2 & a_1\\ a_2 & 0 & -a_0\\ -a_1 & a_0 & 0\end{pmatrix}\cdot\begin{pmatrix}b_0\\ b_1\\ b_2\end{pmatrix}=\begin{pmatrix}-a_2b_1+a_1b_2\\ a_2b_0-a_0b_2\\ -a_1b_0+a_0b_1\end{pmatrix}=\mathbf{a}\times\mathbf{b}. \tag{4.64}$$

Some useful properties of this skew operator are:

- The skew operator commutes with differentiation, *i.e.*

$$\frac{\partial}{\partial x}\Lambda\left[\mathbf{a}\right]=\Lambda\left[\frac{\partial\mathbf{a}}{\partial x}\right]. \tag{4.65}$$

- The skew operator is linear

$$\Lambda\left[\mathbf{a}+c\mathbf{b}\right]=\Lambda\left[\mathbf{a}\right]+c\Lambda\left[\mathbf{b}\right]. \tag{4.66}$$

- The Jacobian of a cross-product can be easily represented using the skew operator as

$$\frac{\partial}{\partial\mathbf{a}}\Big(\Lambda\left[\mathbf{a}\right]\cdot\mathbf{b}\Big)=\begin{pmatrix}0 & b_2 & -b_1\\ -b_2 & 0 & b_0\\ b_1 & -b_0 & 0\end{pmatrix}=-\Lambda\left[\mathbf{b}\right]. \tag{4.67}$$

  To see this we calculate the derivative with respect to a_0:

$$\frac{\partial}{\partial a_0}\Big(\Lambda\left[\mathbf{a}\right]\cdot\mathbf{b}\Big)=\begin{pmatrix}0 & 0 & 0\\ 0 & 0 & -1\\ 0 & 1 & 0\end{pmatrix}\mathbf{b}=\begin{pmatrix}0\\ -b_2\\ b_1\end{pmatrix}, \tag{4.68}$$

  this is the first column of $-\Lambda\left[\mathbf{b}\right]$. Repeating the calculation for the derivatives with respect to a_1 and a_2 results in (4.67). This simplification is the main advantage of the skew operator notation in the following Jacobian calculations.

Due to the linearity of the skew operator we can write the Jacobian of a cross

product including $\mathbf{m}_h$ as

$$\begin{aligned}\frac{\partial}{\partial\tilde{\mathbf{m}}_l}\Big(\Lambda\left[\mathbf{m}_h\right]\mathbf{b}\Big) &= \frac{\partial}{\partial\tilde{\mathbf{m}}_l}\Lambda\left[\sum_k \psi_k\mathbf{m}_k\right]\mathbf{b},\\ &= \sum_k \psi_k\frac{\partial}{\partial\tilde{\mathbf{m}}_l}\Big(\Lambda\left[\tilde{\mathbf{m}}_k\right]\mathbf{b}\Big),\end{aligned} \tag{4.69}$$

then using the fact that $\frac{\partial\tilde{\mathbf{m}}_k}{\partial\tilde{\mathbf{m}}_l} = \delta_{lk}$ we see that the summation can be dropped, leaving

$$\frac{\partial}{\partial\tilde{\mathbf{m}}_l}\Big(\Lambda\left[\mathbf{m}_h\right]\mathbf{b}\Big) = \psi_l\frac{\partial}{\partial\tilde{\mathbf{m}}_l}\Big(\Lambda\left[\tilde{\mathbf{m}}_l\right]\cdot\mathbf{b}\Big). \tag{4.70}$$

Next, by using (4.67) we obtain

$$\frac{\partial}{\partial\tilde{\mathbf{m}}_l}\Big(\Lambda\left[\mathbf{m}_h\right]\mathbf{b}\Big) = -\psi_l\Lambda\left[\mathbf{b}\right], \tag{4.71}$$

and finally by combining (4.71) with the product rule we obtain the identity

$$\frac{\partial}{\partial\tilde{\mathbf{m}}_l}\Big(\Lambda\left[\mathbf{m}_h\right]\cdot\mathbf{g}(\mathbf{m}_h)\Big) = \Lambda\left[\mathbf{m}_h\right]\cdot\frac{\partial\mathbf{g}(\mathbf{m}_h)}{\partial\tilde{\mathbf{m}}_l} - \psi_l\Lambda\left[\mathbf{g}(\mathbf{m}_h)\right]. \tag{4.72}$$

This provides us with a simple representation for almost all of the terms in the LLG Jacobian.

Now we calculate the Jacobian itself. Written in terms of the skew operator the fully discrete LLG residual, (4.58), becomes

$$\begin{aligned}\mathbf{r}_n = \mathbf{r}(\mathbf{m}_h,\varphi_n) = \int_\Omega \frac{\partial\mathbf{m}_h}{\partial t}\varphi_n + \Lambda\left[\mathbf{m}_h\right]\cdot\left(\mathbf{h}_{\text{ap}} + \mathbf{h}_{\text{ca}} - \nabla\phi - \alpha\frac{\partial\mathbf{m}_h}{\partial t}\right)\varphi_n\\ - \left(\Lambda\left[\mathbf{m}_h\right]\nabla\mathbf{m}_h\right) : \nabla\varphi_n\big)\,\mathrm{d}\Omega.\end{aligned} \tag{4.73}$$

To simplify the following calculations we write the Jacobian as the sum of the

individual contributions:

$$\begin{aligned}
\mathrm{F}_{n,l} &= \frac{\partial \mathbf{r}_n}{\partial \tilde{\mathbf{m}}_l} = \mathrm{A}_{n,l} + \mathrm{B}_{n,l} + \mathrm{C}_{n,l} + \mathrm{D}_{n,l}, \\
\mathrm{A}_{n,l} &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l} \int_\Omega \frac{\partial \mathbf{m}_h}{\partial t} \varphi_n \, \mathrm{d}\Omega, \\
\mathrm{B}_{n,l} &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l} \int_\Omega -\Lambda\left[\mathbf{m}_h\right] \cdot \left(\nabla \mathbf{m}_h : \nabla \varphi_n\right) \mathrm{d}\Omega, \\
\mathrm{C}_{n,l} &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l} \int_\Omega \Lambda\left[\mathbf{m}_h\right] \cdot \mathbf{h}_{\mathrm{ca}} \varphi_n \, \mathrm{d}\Omega, \\
\mathrm{D}_{n,l} &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l} \int_\Omega \Lambda\left[\mathbf{m}_h\right] \cdot \left(\mathbf{h}_{\mathrm{ap}} - \nabla\phi - \alpha \frac{\partial \mathbf{m}_h}{\partial t}\right) \varphi_n \, \mathrm{d}\Omega.
\end{aligned} \tag{4.74}$$

The first contribution is simple

$$\begin{aligned}
\mathrm{A}_{n,l} &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l} \int_\Omega \frac{\partial \mathbf{m}_h}{\partial t} \varphi_n \, \mathrm{d}\Omega \\
&= \frac{c_{\mathrm{ts}}}{\Delta_n} \mathrm{I}_3 \int_\Omega \psi_l \varphi_n \, \mathrm{d}\Omega,
\end{aligned} \tag{4.75}$$

where $\frac{c_{\mathrm{ts}}}{\Delta_n}$ is the constant that multiplies $\mathbf{y}_{n+1}$ in the approximation of the time derivative, for example when using the implicit midpoint rule $c_{\mathrm{ts}} = 1$. So A is a 3×3 block diagonal matrix where each block is a mass matrix multiplied by a constant depending on the time integration scheme.

Next we calculate the contribution from the damping, applied field and magnetostatic field. If we write the damping term in the form of (4.72) we have $\mathbf{g}(\mathbf{m}_h) = \frac{\partial \mathbf{m}_h}{\partial t}$, and

$$\frac{\partial \mathbf{g}(\mathbf{m}_h)}{\partial \tilde{\mathbf{m}}_l} = \frac{\partial}{\partial \tilde{\mathbf{m}}_l} \left(\frac{\partial \mathbf{m}_h}{\partial t}\right) = \frac{c_{\mathrm{ts}}}{\Delta_n} \psi_l \mathrm{I}_3. \tag{4.76}$$

Now using (4.72) with (4.76) we immediately obtain

$$\begin{aligned}
\mathrm{D}_{n,l} &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l} \int_\Omega \Lambda\left[\mathbf{m}_h\right] \cdot \left(\mathbf{h}_{\mathrm{ap}} - \nabla\phi - \alpha \frac{\partial \mathbf{m}_h}{\partial t}\right) \varphi_n \, \mathrm{d}\Omega, \\
&= \int_\Omega \varphi_n \big(- \psi_l \Lambda\left[\mathbf{h}_{\mathrm{ap}}\right] + \psi_l \Lambda\left[\nabla\phi\right] + \alpha \psi_l \Lambda\left[\frac{\partial \mathbf{m}_h}{\partial t}\right] - \alpha \psi_l \frac{c_{\mathrm{ts}}}{\Delta_n} \Lambda\left[\mathbf{m}_h\right] \mathrm{I}_3 \big) \, \mathrm{d}\Omega, \\
&= \int_\Omega \varphi_n \psi_l \left(\Lambda\left[-\mathbf{h}_{\mathrm{ap}} + \nabla\phi + \alpha \frac{\partial \mathbf{m}_h}{\partial t} - \alpha \frac{c_{\mathrm{ts}}}{\Delta_n} \mathbf{m}_h \right] \right) \mathrm{d}\Omega.
\end{aligned} \tag{4.77}$$

This is a 3×3 skew-symmetric matrix where each element is a mass matrix

element multiplied by a collection of field/magnetisation components.

We turn now to the contribution due to magnetocrystalline anisotropy. Writing the residual in the form of (4.72) we have

$$\mathbf{g}(\mathbf{m}_h) = \mathbf{h}_{\text{ca}} = \mathcal{K}_1(\mathbf{m}_h \cdot \hat{\mathbf{e}})\hat{\mathbf{e}}, \tag{4.78}$$

and thus

$$\begin{aligned}\frac{\partial \mathbf{g}(\mathbf{m}_h)}{\partial \tilde{\mathbf{m}}_l} &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l}(\mathcal{K}_1(\mathbf{m}_h \cdot \hat{\mathbf{e}})\hat{\mathbf{e}}), \\ &= \mathcal{K}_1\left(\frac{\partial}{\partial m_{x,l}}(\mathbf{m}_h \cdot \hat{\mathbf{e}}), \frac{\partial}{\partial m_{y,l}}(\mathbf{m}_h \cdot \hat{\mathbf{e}}), \frac{\partial}{\partial m_{z,l}}(\mathbf{m}_h \cdot \hat{\mathbf{e}})\right) \otimes \hat{\mathbf{e}}, \\ &= \mathcal{K}_1 \psi_l \, \hat{\mathbf{e}} \otimes \hat{\mathbf{e}}.\end{aligned} \tag{4.79}$$

Then combining (4.72) and (4.79) gives

$$\begin{aligned}\mathrm{C}_{n,l} &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l}\int_\Omega \Lambda\left[\mathbf{m}_h\right] \cdot \mathbf{h}_{\text{ca}} \varphi_n \, \mathrm{d}\Omega, \\ &= \underbrace{\int_\Omega -\varphi_n \psi_l \Lambda\left[\mathbf{h}_{\text{ca}}\right] \, \mathrm{d}\Omega}_{\mathrm{C}'_{n,l}} + \underbrace{\mathcal{K}_1 \int_\Omega \varphi_n \psi_l \Lambda\left[\mathbf{m}_h\right] \, (\hat{\mathbf{e}} \otimes \hat{\mathbf{e}}) \, \mathrm{d}\Omega}_{\mathrm{J}_{\text{ca},n,l}},\end{aligned} \tag{4.80}$$

where $\mathbf{a} \otimes \mathbf{b}$ is the tensor product of $\mathbf{a}$ and $\mathbf{b}$. Note that C' is of the same form as D and J_{ca} is a 3×3 block matrix with no empty blocks in the general case. However, if $\hat{\mathbf{e}}$ is aligned with any of the Cartesian axes (as is typical) then only one entry of the matrix $(\hat{\mathbf{e}} \otimes \hat{\mathbf{e}})$ is non-zero and only one block of J_{ca} is non-empty.

Finally we calculate the exchange contribution. We first derive the identity

$$\begin{aligned}\frac{\partial}{\partial \tilde{\mathbf{m}}_l}\left(\widetilde{\nabla}\mathbf{m}_h : \widetilde{\nabla}\varphi\right) &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l}\begin{pmatrix}(\sum_k \tilde{\mathbf{m}}_k \nabla\psi_k) \cdot \nabla\varphi \\ (\sum_k \tilde{\mathbf{m}}_k \nabla\psi_k) \cdot \nabla\varphi \\ (\sum_k \tilde{\mathbf{m}}_k \nabla\psi_k) \cdot \nabla\varphi\end{pmatrix}, \\ &= \sum_k \frac{\partial \tilde{\mathbf{m}}_k}{\partial \tilde{\mathbf{m}}_l}\begin{pmatrix}\nabla\psi_k \cdot \nabla\varphi \\ \nabla\psi_k \cdot \nabla\varphi \\ \nabla\psi_k \cdot \nabla\varphi\end{pmatrix}, \\ &= \mathrm{I}_3(\nabla\psi_l \cdot \nabla\varphi).\end{aligned} \tag{4.81}$$

Then using (4.72) and (4.81) we get

$$\begin{aligned}
\mathrm{B}_{n,l} &= \frac{\partial}{\partial \tilde{\mathbf{m}}_l} \int_\Omega -\Lambda\left[\mathbf{m}_h\right] \cdot \left(\widetilde{\nabla}\mathbf{m}_h : \widetilde{\nabla}\varphi_n\right) \mathrm{d}\Omega, \\
&= \int_\Omega \psi_l \Lambda\left[\widetilde{\nabla}\mathbf{m}_h : \widetilde{\nabla}\varphi_n\right] - \Lambda\left[\mathbf{m}_h\right]\left(\nabla\psi_l \cdot \nabla\varphi_n\right) \mathrm{d}\Omega, \\
&= \int_\Omega \Lambda\left[\psi_l \widetilde{\nabla}\mathbf{m}_h : \widetilde{\nabla}\varphi_n - \mathbf{m}_h\left(\nabla\psi_l \cdot \nabla\varphi_n\right)\right] \mathrm{d}\Omega,
\end{aligned} \tag{4.82}$$

which is a skew-symmetric 3×3 matrix with each element reminiscent of an element of the discrete Laplacian (4.62).

When written in block form the total LLG Jacobian is

$$\mathrm{F} = \begin{pmatrix} \frac{c_{\mathrm{ts}}}{\Delta_n}\mathrm{M} & -\mathrm{K}_z & \mathrm{K}_y \\ \mathrm{K}_z & \frac{c_{\mathrm{ts}}}{\Delta_n}\mathrm{M} & -\mathrm{K}_x \\ -\mathrm{K}_y & \mathrm{K}_x & \frac{c_{\mathrm{ts}}}{\Delta_n}\mathrm{M} \end{pmatrix} + \mathrm{J}_{\mathrm{ca}}, \tag{4.83}$$

where M is the mass matrix

$$\mathrm{M}_{i,j} = \int_\Omega \varphi_i \psi_j \, \mathrm{d}\Omega; \tag{4.84}$$

K_i are the skew-symmetric contributions B, C′ and D of (4.77), (4.80) and (4.82); and J_{ca} is from (4.80).

4.4.3 LLG-magnetostatics coupling Jacobians

The non-linear problem as written in (4.50) is the "monolithic" form, *i.e.* all equations are solved simultaneously. The Jacobian for this complete non-linear system is

$$\mathrm{J} = \begin{pmatrix} \mathrm{F} & \mathrm{P} \\ \mathrm{Q} & \mathrm{A} \end{pmatrix}, \tag{4.85}$$

where A and F have been derived in Section 4.4.1 and Section 4.4.2 respectively; $\mathrm{P} = \frac{\partial \mathbf{r}}{\partial \phi}$ and $\mathrm{Q} = \frac{\partial s}{\partial \mathbf{m}_h}$. We now derive expressions for P and Q. Note that it is also possible (and common) to solve the non-linear problem (4.50) as two separate systems: one for $\mathbf{m}_h$ then one for ϕ. More details and the advantages of each approach are discussed in Chapter 6.

Now we derive $\mathrm{P}_{n,l}$. It is a 3×1 block matrix because we are dealing with a vector

of three residuals differentiated by one variable ($\tilde{\phi}_l$). Most of the terms in the LLG residual are zero when differentiated with respect to any $\tilde{\phi}_l$ so the calculation is quite simple. From (4.58), using discrete approximations and dropping the zero terms we have

$$\begin{aligned} \mathrm{P}_{n,l} &= \frac{\partial \mathbf{r}_n}{\partial \tilde{\phi}_l} = \frac{\partial}{\partial \tilde{\phi}_l} \int_\Omega -\varphi_n \mathbf{m}_h \times \nabla \phi_h \, \mathrm{d}\Omega, \\ &= -\int_\Omega \varphi_n \mathbf{m}_h \times \nabla \psi_l \, \mathrm{d}\Omega, \end{aligned} \tag{4.86}$$

Next we derive $\mathrm{Q}_{n,l}$, which is a 1×3 block matrix because we differentiate by the three components of $\mathbf{m}_h$. From (4.59) we have

$$\begin{aligned} \mathrm{Q}_{n,l} &= \frac{\partial s_n}{\partial \tilde{\mathbf{m}}_l} = \frac{\partial}{\partial \tilde{\mathbf{m}}_l} \int_\Omega -\varphi_n \nabla \cdot \mathbf{m}_h \, \mathrm{d}\Omega, \\ &= \sum_j \frac{\partial}{\partial \tilde{\mathbf{m}}_l} \int_\Omega -\varphi_n \left(\frac{\partial \psi_j}{\partial x} m_{h,x,j} + \frac{\partial \psi_j}{\partial y} m_{h,y,j} + \frac{\partial \psi_j}{\partial z} m_{h,z,j} \right) \mathrm{d}\Omega, \\ &= -\int_\Omega \varphi_n \left(\frac{\partial \psi_l}{\partial x}, \frac{\partial \psi_l}{\partial y}, \frac{\partial \psi_l}{\partial z} \right) \mathrm{d}\Omega, \\ &= -\int_\Omega \varphi_n (\nabla \psi_l)^T \, \mathrm{d}\Omega. \end{aligned} \tag{4.87}$$

So in block form the complete Jacobian is

$$\mathrm{J} = \left(\begin{array}{ccc|c} \frac{c_{\mathrm{ts}}}{\Delta_n}\mathrm{M} & -\mathrm{K}_z & \mathrm{K}_y & \mathrm{P}_x \\ \mathrm{K}_z & \frac{c_{\mathrm{ts}}}{\Delta_n}\mathrm{M} & -\mathrm{K}_x & \mathrm{P}_y \\ -\mathrm{K}_y & \mathrm{K}_x & \frac{c_{\mathrm{ts}}}{\Delta_n}\mathrm{M} & \mathrm{P}_z \\ \hline \mathrm{Q}_x & \mathrm{Q}_y & \mathrm{Q}_z & \mathrm{A} \end{array} \right) + \mathrm{J}_{\mathrm{ca}}, \tag{4.88}$$

where J_{ca} is appropriately zero padded.

4.5 Geometric integration with the finite element method

In weak form based methods it turns out that the magnetisation length conservation property of the IMR does not automatically apply. The problem is that (outside of the infinite node limit) the weak form equations enforce properties of the *integral* of the solution, so with the FEM and IMR $\int_{\Omega_e} |\mathbf{m}_h| - 1 \, \mathrm{d}\Omega_e = 0$ is

enforced rather than the equivalent nodal property. The solution is to evaluate the integrals using a quadrature scheme which somehow directly links the nodal values with the integral values. The downside of such a scheme is that the accuracy of the evaluation of integrals is reduced (since the standard quadratures are chosen for their optimal accuracy).

In the micromagnetics literature this choice of quadrature is known as "reduced integration" [22]. However in other finite element literature reduced integration refers to the use of a Gaussian quadrature whose order is not sufficient to evaluate the integrals exactly, typically it is used to reduce the computational cost of the quadrature. *Nodal integration* is the standard term for quadrature schemes where the nodal values are used directly in the quadrature (typically with mesh-free methods *e.g.* [78]), but we prefer the term "nodal quadrature" to avoid confusion with time integration.

In this section we first demonstrate that $|\mathbf{m}_j| = 1$ is conserved when a nodal integration scheme is used. Then we demonstrate that the energy property of IMR (as discussed in Section 3.4.9) applies in our FEM + IMR model. Finally we show in detail why the conservation of magnetisation length is lost with standard quadrature schemes.

4.5.1 A local nodal quadrature scheme

In order to regain conservation properties in a weak-form-based method we introduce a nodal quadrature scheme as used by Bartels et. al. [10]:

$$\int_{\Omega_e} f(\mathbf{x})\, \mathrm{d}\mathbf{x} \approx \sum_{l \in \{\text{nodes in } \Omega_e\}} \beta_l f(\mathbf{x}_l), \tag{4.89}$$

where β_l is a weight. This is simply the weighted sum of the value of the integrand at the finite element nodes. As an additional benefit this can greatly simplify the calculations since no interpolation of the values to the knots (integration points) is needed.

We now need to derive suitable values for the β_l. To do so we represent the quadrature scheme as an integral of an appropriate interpolating polynomial (in fact the interpolating polynomial defines the quadrature scheme), then rearrange the equation to find the weights (see, *e.g.* [59, p. 480]). In FEM codes quadrature

is typically done in local coordinates, as discussed in Section 4.1.4, so we calculate weights applicable in a local coordinate quadrature.

A set of quadrature-interpolation-polynomials which give the nodal quadrature property (4.89) are the test basis functions. Using these we obtain:

$$\begin{aligned}\int_{\Omega_e} f(\mathbf{x})\,\mathrm{d}\mathbf{x} &= \int_{\Omega_e} f(\mathbf{s})J(\mathbf{s})\,\mathrm{d}\mathbf{s},\\ &\approx \int_{\Omega_e}\left(\sum_l f(\mathbf{s}_l)\varphi_l(\mathbf{s})\right)J(\mathbf{s})\,\mathrm{d}\mathbf{s},\\ &\approx \sum_l f(\mathbf{s}_l)\int_{\Omega_e}\varphi_l(\mathbf{s})J(\mathbf{s})\,\mathrm{d}\mathbf{s},\end{aligned} \tag{4.90}$$

where $J = \left|\frac{\partial \mathbf{s}}{\partial \mathbf{x}}\right|$ is the Jacobian of the transformation from local to global coordinates. Comparing (4.89) with (4.90) we see that

$$\beta_l = \int_{\Omega_e}\varphi_l(\mathbf{s})J(\mathbf{s})\,\mathrm{d}\mathbf{s}\left(= \int_{\Omega_e}\varphi_l\,\mathrm{d}\mathbf{x}\right). \tag{4.91}$$

If $f(\mathbf{s})$ is a first order polynomial then our nodal quadrature scheme is exact. This is worse than the Gaussian quadrature with the same number of quadrature points, where polynomials of up to 3rd order are integrated exactly. As such most terms of the LLG residual are not integrated exactly. However the exchange residual contribution is only a first order polynomial, for example the x component is

$$r_{\mathrm{exch},x} = -\int_\Omega m_y(\nabla\varphi\cdot\nabla m_z)\,\mathrm{d}\Omega + \int_\Omega m_z(\nabla\varphi_n\cdot\nabla m_y)\,\mathrm{d}\Omega. \tag{4.92}$$

The gradient terms are constants (m_i and φ_n are linear polynomials so their first derivatives are constant), hence the exchange contribution is integrated exactly.

Note that with this quadrature scheme we have that

$$\int_\Omega \psi_k\varphi_n\,\mathrm{d}\Omega = c\delta_{kn}, \tag{4.93}$$

meaning that the mass matrix, (4.84), is diagonal. This makes the nodal quadrature scheme a form of *mass-lumping* which is often used in the FEM in combination with explicit time integration to remove the requirement for a linear solve.

4.5.2 Conservation of $|\mathbf{m}|$

After reducing the norm of the discrete LLG residual to tolerance ϵ_{N} we have for each j (using vector test function notation):

$$\left|\left|\int_\Omega \left[\frac{\partial \mathbf{m}_h}{\partial t} + \mathbf{m}_h \times \left(\mathbf{h} - \alpha \frac{\partial \mathbf{m}_h}{\partial t}\right)\right] \cdot \boldsymbol{\varphi}_j + (\mathbf{m}_h \times \nabla \mathbf{m}_h) : \nabla \boldsymbol{\varphi}_j \, \mathrm{d}\Omega \right|\right| < \epsilon_{\mathrm{N}}, \tag{4.94}$$

where $||\cdot||$ is the norm used to assess the convergence of the linearisation method (in our implementation this is $||\cdot||_\infty$) and $\boldsymbol{\varphi}_j$ is the vector test basis function corresponding to node j. Hence, by writing any test function $\mathbf{v}$ as a linear combination of the basis functions and then using the linearity of $\boldsymbol{\varphi}$ in (4.94) and the triangle inequality, we have

$$\left|\left|\int_\Omega \left[\frac{\partial \mathbf{m}_h}{\partial t} + \mathbf{m}_h \times \left(\mathbf{h} - \alpha \frac{\partial \mathbf{m}_h}{\partial t}\right)\right] \cdot \mathbf{v} + (\mathbf{m}_h \times \nabla \mathbf{m}_h) : \nabla \mathbf{v} \, \mathrm{d}\Omega \right|\right| < \epsilon_{\mathrm{N}} C, \tag{4.95}$$

for all $\mathbf{v} \in (\mathcal{H}^1_h(\Omega))^3$ where $C = C(\mathbf{v}) = \sum_j |c_{x,j}| + |c_{y,j}| + |c_{z,j}|$ is the sum of all coefficients in the basis representation of $\mathbf{v}$.

In this section we write the midpoint value (in time) of the magnetisation at the jth node as $\mathbf{m}_{j,\frac{1}{2}} = (\mathbf{m}_{j,n} + \mathbf{m}_{j,n+1})/2$. Similarly for the magnetic field we write $\mathbf{h}_{j,\frac{1}{2}} = \mathbf{h}((t_n + t_{n+1})/2, \mathbf{x}_j, \mathbf{m}_j)$ and the time derivative of the magnetisation $\frac{\partial \mathbf{m}}{\partial t}_{j,\frac{1}{2}} = (\mathbf{m}_{j,n+1} - \mathbf{m}_{j,n})/\Delta_n$.

For the purposes of this derivation we choose the test function to be $\mathbf{v} = \mathbf{m}_{j,\frac{1}{2}}\varphi_j$, *i.e.* the midpoint value of the magnetisation at node j multiplied by the corresponding test basis function. We know that $\mathbf{m}_{j,\frac{1}{2}}\varphi_j \in (\mathcal{H}^1_h(\Omega))^3$ because it is simply a vector valued multiple of a basis function of $\mathcal{H}^1_h(\Omega)$.

Using the new notation and this test function we have that

$$\begin{aligned}\left|\left| \sum_l \beta_l \Bigg[\left(\frac{\partial \mathbf{m}}{\partial t}_{l,\frac{1}{2}} + \mathbf{m}_{l,\frac{1}{2}} \times \left(\mathbf{h}_{l,\frac{1}{2}} - \alpha \frac{\partial \mathbf{m}}{\partial t}_{l,\frac{1}{2}} \right) \right) \cdot \mathbf{m}_{j,\frac{1}{2}} \varphi_j(\mathbf{x}_l) \\ + \left(\mathbf{m}_{l,\frac{1}{2}} \times \nabla \mathbf{m}_{l,\frac{1}{2}} \right) : \nabla (\mathbf{m}_{j,\frac{1}{2}} \varphi_j(\mathbf{x}_l)) \Bigg] \right|\right| < \epsilon_{\mathrm{N}} C.\end{aligned} \tag{4.96}$$

Using $\varphi_j(\mathbf{x}_l) = \delta_{jl}$ we can eliminate the summation, leaving only

$$\left|\left|\beta_j\left[\left(\frac{\partial \mathbf{m}}{\partial t}_{j,\frac{1}{2}} + \mathbf{m}_{j,\frac{1}{2}} \times \left(\mathbf{h}_{j,\frac{1}{2}} - \alpha \frac{\partial \mathbf{m}}{\partial t}_{j,\frac{1}{2}}\right)\right) \cdot \mathbf{m}_{j,\frac{1}{2}} + \left(\mathbf{m}_{j,\frac{1}{2}} \times \nabla \mathbf{m}_{j,\frac{1}{2}}\right) : \nabla(\mathbf{m}_{j,\frac{1}{2}})\right]\right|\right| < \epsilon_{\mathrm{N}} C. \tag{4.97}$$

Now, by the properties of the triple product, the precession and damping terms vanish leaving

$$\left|\left|\beta_j \frac{\partial \mathbf{m}}{\partial t}_{j,\frac{1}{2}} \cdot \mathbf{m}_{j,\frac{1}{2}}\right|\right| < \epsilon_{\mathrm{N}} C. \tag{4.98}$$

Substituting in the formula for the IMR from (4.55) gives the desired conservation result:

$$\begin{aligned} \left|\left|\frac{\beta_j}{2\Delta_n}(\mathbf{m}_{j,n+1} - \mathbf{m}_{j,n}) \cdot (\mathbf{m}_{j,n+1} + \mathbf{m}_{j,n})\right|\right| &< \epsilon_{\mathrm{N}} C, \\ \left|\left|\,|\mathbf{m}_{j,n+1}|^2 - |\mathbf{m}_{j,n}|^2\right|\right| &< \frac{2\Delta_n}{\beta_j}\epsilon_{\mathrm{N}} C. \end{aligned} \tag{4.99}$$

This is applicable for all nodes j. Note that for this test function $C \sim \left|\mathbf{m}_{j,\frac{1}{2}}\right|^2 = \mathrm{O}(1)$ assuming that the solver has not gone catastrophically wrong.[5] Hence all nodal magnetisation lengths are conserved up to an error which can be controlled by decreasing the tolerance of the linearisation method.

4.5.3 Energy properties

We now show that the IMR/FEM discretisation discussed above also retains the energy conservation property of the IMR applied to an ODE discussed in Section 3.4.9.

For now we assume an additional property of the solution and test function spaces: all functions must be in $\mathcal{H}_h^2(\Omega)$ (*i.e.* the second derivatives are integrable). This constraint is required so that we can use a simpler form of the LLG residual and can be relaxed by using nodal quadrature and more technical arguments which are discussed following the derivation. We also require that $\mathbf{h}_{\mathrm{ap}}$ can be written as a linear combination of basis functions. Finally we make use of the fact that the test and solution spaces coincide for the magnetisation (due to the lack of any Dirichlet boundary conditions).

[5] Even if $|\mathbf{m}| = 1$ does not hold (and for the purposes of this proof we cannot assume that it does), the length of the magnetisation cannot be too far away from one.

As in the previous section we start from the LLG residual:

$$\begin{aligned} &\left|\left|\int_\Omega \left[\frac{\partial \mathbf{m}_h}{\partial t} + \mathbf{m}_h \times \left(\mathbf{h} - \alpha \frac{\partial \mathbf{m}_h}{\partial t}\right)\right] \cdot \mathbf{v} \, \mathrm{d}\Omega \right|\right| < \epsilon_{\mathrm{N}} C(\mathbf{v}), \\ &\mathbf{h} = \nabla^2 \mathbf{m}_h - \nabla \phi + \mathbf{h}_{\mathrm{ap}}(t, \mathbf{x}) + \mathbf{h}_{\mathrm{ca}}(\mathbf{m}_h), \end{aligned} \tag{4.100}$$

and choose as the test function

$$\mathbf{v} = \mathbf{h} - \alpha \frac{\partial \mathbf{m}_h}{\partial t}. \tag{4.101}$$

The term $\alpha \frac{\partial \mathbf{m}_h}{\partial t}$ belongs to the space of test functions since it is a linear combination of $\mathbf{m}_{h,n}$ and $\mathbf{m}_{h,n+1}$, each of which is in the solution space. Since we are using $\mathbf{v} \in \mathcal{H}_h^2(\Omega)$ the function $\nabla^2 \mathbf{m}_h$ is clearly in the test function space. The $\mathbf{h}_{\mathrm{ca}}$ effective field is similarly contained in the space of test functions as it is a linear function of $\mathbf{m}_h$. Finally, since the space of test functions for $\mathbf{m}$ is non-zero on the boundary (*i.e.* there are no Dirichlet conditions on $\mathbf{m}$) we can represent $\nabla \phi$ using them, for any boundary conditions on ϕ.

We substitute (4.101) into (4.100) and apply the triple product identity to find that

$$\left|\left|\int_\Omega \frac{\partial \mathbf{m}_h}{\partial t} \cdot \mathbf{h} \, \mathrm{d}\Omega - \alpha \int_\Omega \frac{\partial \mathbf{m}_h}{\partial t} \cdot \frac{\partial \mathbf{m}_h}{\partial t} \, \mathrm{d}\Omega \right|\right| < \epsilon_{\mathrm{N}} C. \tag{4.102}$$

After substituting in the IMR approximations and using a midpoint approximation for the applied field in (4.102) the expression inside the norm coincides with (3.48) of the derivation of the energy loss property for the strong form of the LLG. Hence we proceed similarly to Section 3.4.9: we substitute (3.51)–(3.53) into (4.102) to obtain

$$\begin{aligned} &\left|\left| \frac{e_{n+1} - e_n}{\Delta_n} + \alpha \left|\left| \frac{\mathbf{m}_{h,n+1} - \mathbf{m}_{h,n}}{\Delta_n} \right|\right|_{L^2}^2 + \left(\frac{\mathbf{h}_{\mathrm{ap}}(t_{n+1}) + \mathbf{h}_{\mathrm{ap}}(t_n)}{\Delta_n}, \, \frac{\mathbf{m}_{h,n+1} + \mathbf{m}_{h,n}}{2} \right)_{L^2} \right. \\ &\quad \left. - \left(\mathcal{E}_{\mathrm{ap}}, \, \frac{\mathbf{m}_{h,n+1} - \mathbf{m}_{h,n}}{\Delta_n} \right)_{L^2} \right|\| \quad < \epsilon_{\mathrm{N}} C. \end{aligned} \tag{4.103}$$

Finally, by using the triangle inequality[6], we have

$$\begin{aligned}&\left|\left|\frac{e_{n+1}-e_n}{\Delta_n}+\alpha\left|\left|\frac{\mathbf{m}_{h,n+1}-\mathbf{m}_{h,n}}{\Delta_n}\right|\right|_{L^2}^2+\left(\frac{\mathbf{h}_{\mathrm{ap}}(t_{n+1})+\mathbf{h}_{\mathrm{ap}}(t_n)}{\Delta_n},\ \frac{\mathbf{m}_{h,n+1}+\mathbf{m}_{h,n}}{2}\right)_{L^2}\right|\right|\\&<\left|\left|\left(\mathcal{E}_{\mathrm{ap}},\ \frac{\mathbf{m}_{h,n+1}-\mathbf{m}_{h,n}}{\Delta_n}\right)_{L^2}\right|\right|+\epsilon_{\mathrm{N}}C.\end{aligned}\tag{4.104}$$

The inequality (4.104) has the same interpretations as in Section 3.4.9. In particular: if $\alpha = 0$ and $\mathbf{h}_{\mathrm{ap}}$ is constant in time we have energy conservation (up to an error which can be controlled using the linearisation tolerance).

Note that the energy integrals and L^2 norms in (4.104) are not necessarily exact because we are using nodal quadrature. The exchange energy is exact for the same reasons as discussed in Section 4.5.1, and the energy due to magnetostatics is exact for the same reason. In contrast the magnetocrystalline anisotropy energy and $\frac{\partial \mathbf{m}}{\partial t}$ norm integrals are not exact and the Zeeman energy depends on the spatial variation of the applied field: if the field is constant in space then the conserved energy will be exact.

Also note that for the test function given in (4.101) the value of C is fairly large

$$\begin{aligned}C&=\mathrm{O}\Big(N_n\Big(1+\mathcal{K}_1+\max_{\mathbf{x}}\mathbf{h}_{\mathrm{ap}}(t,\mathbf{x})\Big)\Big),\\&=\mathrm{O}(N_n),\end{aligned}\tag{4.105}$$

and so the bound may be lose for very large numbers of nodes. However this a pessimistic bound because it assumes that every coefficient of the basis expansion has the same sign, *i.e.* all effective fields are in the same direction over all of Ω, and $\frac{\partial \mathbf{m}}{\partial t}$ is in the opposite direction over all of Ω. The actual error is not likely to be this large except in extremely symmetric cases.

The restriction of $\mathbf{v} \in \mathcal{H}_h^2(\Omega)$ introduced at the start of this section can be relaxed by considering an implicitly defined discrete Laplacian operator defined for $\mathbf{a} \in \mathcal{H}^1(\Omega)$, $\mathbf{b} \in \mathcal{H}_h^1(\Omega)$ as

$$-\Big(\widetilde{\Delta}\mathbf{a}(\mathbf{x}),\ \mathbf{b}(\mathbf{x})\Big)_{\mathrm{nodal}}=\left(\nabla\mathbf{a}(\mathbf{x}),\ \nabla\mathbf{b}(\mathbf{x})\right)_{L^2},\tag{4.106}$$

[6]The standard triangle inequality is $||x+y|| \leq ||x|| + ||y||$, by choosing $y = -a$ we see that $||x-a|| \leq ||x|| + ||-a|| = ||x|| + ||a||$ giving the form used here.

where

$$\left(\mathbf{a},\, \mathbf{b}\right)_{\text{nodal}} = \sum_{j \in \text{nodes}} \beta_j \mathbf{a}(\mathbf{x}_j) \cdot \mathbf{b}(\mathbf{x}_j), \tag{4.107}$$

i.e. the nodally-integrated version of the L^2 inner product. It can be shown that [9]:

$$\left|\left|\widetilde{\Delta}\mathbf{a}(\mathbf{x})\right|\right|_\infty = \left(\widetilde{\Delta}\mathbf{a}(\mathbf{x}),\, \widetilde{\Delta}\mathbf{a}(\mathbf{x})\right)_\infty < \infty, \tag{4.108}$$

so $\left|\left|\widetilde{\Delta}\mathbf{a}(\mathbf{x})\right|\right|_{L^2} < \infty$ and the required integrals are finite when calculated using nodal quadrature. Note that $\left(\widetilde{\Delta}\mathbf{m}_h,\, \mathbf{m}_h\right)_{\text{nodal}} = \left(\nabla\mathbf{m}_h,\, \nabla\mathbf{m}_h\right)_{L^2} = e_{\text{ex}}$. This means that we can replace the Laplacian operator everywhere in the derivation by the discrete Laplace operator to obtain the derivation without the restriction $\mathbf{v} \in \mathcal{H}^2_h(\Omega)$.

4.5.4 Non-conservation with Gaussian quadrature

We now show (by a counterexample) that the conservation result does not hold with a standard quadrature scheme. To obtain our result we examine the case where $\mathbf{v} = \mathbf{m}_h$ (recall that $\mathbf{m}_h \in (\mathcal{H}^1_h(\Omega))^3$ because we are using identical test and solution function spaces and each component of $\mathbf{m}_h$ is a linear combination of solution basis functions).

For this section we assume that the non-linear solver has converged to a tolerance of exactly zero for simplicity, *i.e.* $\epsilon_{\text{N}} = 0$. By substituting $\mathbf{v} = \mathbf{m}_h$ and $\epsilon_{\text{N}} = 0$ into (4.95) then using the triple product identities we obtain

$$\begin{aligned} 0 &= \int_\Omega \frac{\partial \mathbf{m}_h}{\partial t} \cdot \mathbf{m}_h \,\mathrm{d}\Omega, \\ &= \frac{1}{2\Delta_n} \int_\Omega \left(\mathbf{m}_{h,n+1} + \mathbf{m}_{h,n}\right) \cdot \left(\mathbf{m}_{h,n+1} - \mathbf{m}_{h,n}\right) \mathrm{d}\Omega, \\ &= \frac{1}{2\Delta_n} \int_\Omega \left|\mathbf{m}_{h,n+1}\right|^2 - \left|\mathbf{m}_{h,n}\right|^2 \mathrm{d}\Omega. \end{aligned} \tag{4.109}$$

Equation (4.109) suggests that we have conservation of magnetisation length. However it is an *integral* relationship, meaning that the values of the integrand at the nodes are not necessarily zero.[7]

We can see this in more detail by writing out the Gaussian quadrature. Dropping

[7] Since $\int_\Omega f(\mathbf{x})\,\mathrm{d}\Omega = 0 \nRightarrow f(\mathbf{x}) \equiv 0$ on Ω.

the constant factor of $2\Delta_n$ and assuming that $|\mathbf{m}_{h,n}| = 1$ everywhere this gives

$$\begin{aligned} 1 &= \int_\Omega \left| \sum_k \mathbf{m}_{k,n+1} \varphi_k(\mathbf{x}) \right|^2 \mathrm{d}\Omega, \\ &= \sum_l w_l \left| \sum_k \mathbf{m}_{k,n+1} \varphi_k(\mathbf{x}_l) \right|^2 . \end{aligned} \tag{4.110}$$

Equation (4.110) can be satisfied without the requirement that $|\mathbf{m}_{k,n+1}| = 1 \ \forall k$. For example if we set the number of nodes to two (*i.e.* a 1D problem with linear basis functions) and assume magnetisation only along the z-axis (*i.e.* $\mathbf{m}_{h,n+1} = (0, 0, m)$) then condition (4.110) becomes (dropping the subscript $n + 1$):

$$\begin{aligned} 1 &= \sum_l w_l \left| \sum_k \mathbf{m}_k \varphi_k(\mathbf{x}_l) \right|^2 , \\ &= w_0(m_0 a + m_1 b)(m_0 a + m_1 b) + w_1(m_0 c + m_1 d)(m_0 c + m_1 d), \\ &= m_0^2(w_0 a^2 + w_1 c^2) + m_1^2(w_0 b^2 + w_1 c^2) + m_0 m_1(2w_0 ab + 2w_1 cd), \\ &= m_0^2 \alpha + m_1^2 \beta + m_0 m_1 \gamma, \end{aligned} \tag{4.111}$$

where a, b, c, d are the values of basis functions at the knots and α, β, γ are constants. So given any m_0 we can solve the above expression to find an m_1 that satisfies the constraint. Since we have chosen $\mathbf{m} = (0, 0, m)$ this means we can vary the magnetisation length at node 0 arbitrarily while still satisfying the constraint.

Finally, we note that if the magnetisation is constant in space at times t_n and t_{n+1} then the integral condition in (4.109) *is* sufficient to give conservation of the magnetisation length at nodes.

Chapter 5

Hybrid finite/boundary element method

The boundary element method (BEM) is a spatial discretisation method similar to the FEM except that the problem on a domain is converted to an equivalent formulation in terms of integrals over the boundary of the domain before the discretisation is applied. The main downside of such a formulation is that it results in a dense matrix after the discretisation is applied.

The hybrid finite/boundary element method (FEM/BEM) is a spatial discretisation method which applies FEM and BEM to different parts of the same problem. In particular this method can be applied in the context of magnetostatic calculations to enforce the boundary condition at infinity in (2.15) without meshing an infinite region of space or arbitrarily truncating it. This is done by applying the BEM to a part of the problem on the infinite external domain in order to convert it into a problem on the boundary of the magnetic domain. The use of FEM for the remaining parts of the problem means that most of the obtained linear systems remain sparse.

In this chapter we first give an overview of the FEM/BEM approach and then show how splitting the magnetostatic problem into two parts and applying some results from potential theory can lead to a formulation which involves only finite domains. We then demonstrate how this formulation can be discretised, and discuss the evaluation of the required integrals. Methods of coupling the FEM/BEM magnetostatic calculations to the FEM-discretised LLG problem (as described in Chapter 4) are discussed in Chapter 6, along with strategies for the solution of

the resulting linear/non-linear systems.

5.1 The continuous FEM/BEM formulation

We want to use the boundary element method in the calculation of the magnetostatic scalar potential, ϕ, (defined in (2.14) and (2.15)) to avoid performing calculations that involve the infinite external domain. However, applying the boundary element method alone would require the solution of dense matrix equations for the entire problem. We can circumvent this issue by applying a hybrid method using both FEM and BEM. We use the linearity of the Poisson operator to split the potential into two parts $\phi = u + v$ in such a way that u can be calculated in the magnetic domain, Ω, using only the finite element method. This results in conditions on v that must be satisfied to give the correct total potential, including the problematic condition at infinity.

It turns out that ϕ can be split in such a way that the v is equivalent to the potential resulting from a specific arrangement of fictional magnetic charges, which is known to solve the required differential equation including the boundary condition at infinity. The details of this "charge distribution", which gives rise to a double-layer potential, depend on the boundary values of u. The use of such a charge distribution is equivalent to applying the BEM to v.

So given u and using the known solution for this charge distribution we can calculate v anywhere. However the calculation of v at a point using this method involves an expensive dense matrix operation, so we only use it to calculate Dirichlet boundary conditions for v on the boundary of the magnetic domain, Γ. These boundary conditions allow us to apply the finite element method to find v inside the magnetic domain Ω using standard sparse matrix techniques.

The magnetostatic field $\mathbf{h}_{\text{ms}} = \nabla\phi$ constructed by the procedure outlined above is *the* magnetostatic field by the uniqueness (up to a constant) of solution for Poisson's equation with Dirichlet or Neumann boundary conditions.[1]

In Section 5.1.1 we describe in more detail the splitting of the potential that allows this process. Then in Section 5.1.2 we describe the properties of the fictional

[1] To see the uniqueness take the difference of two solutions of Poisson's equation: $\phi = \psi_1 - \psi_2$. Then using identity (4.32), the linearity of Poisson's equation and the divergence theorem we obtain $\int_\Gamma \phi\nabla\phi \cdot \hat{\mathbf{n}}\, \mathrm{d}\Gamma = \int_\Omega (\nabla\phi)^2\, \mathrm{d}\Omega$. Applying Dirichlet, Neumann or mixed boundary conditions shows that the boundary integral is zero and hence $\nabla\phi = 0$.

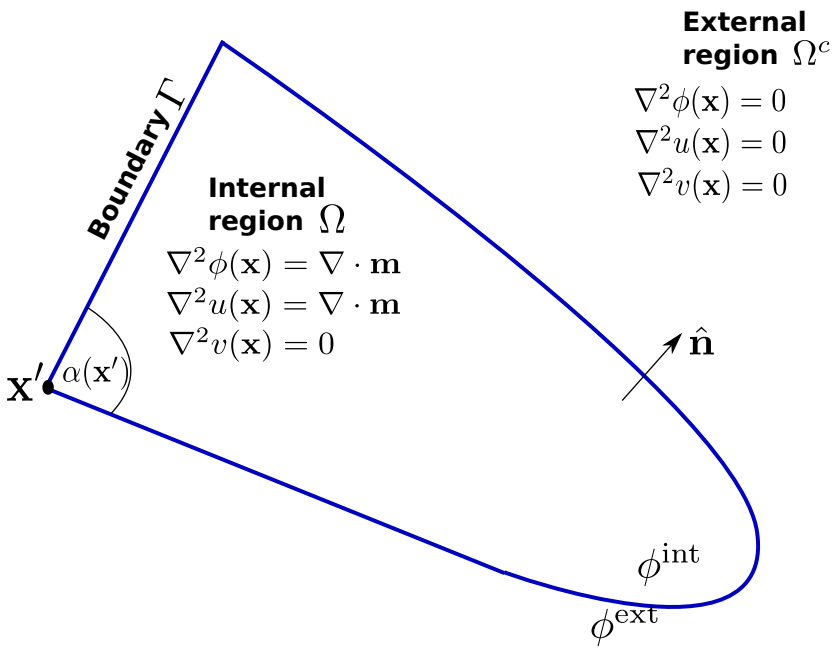

Figure 5.1: A 2D representation of the geometry showing the labels used in this chapter. The point $\mathbf{x}'$ is a singular point of the boundary Γ, the angle $\alpha(\mathbf{x}')$ is as shown.

charge distribution. Finally in Section 5.1.3 we combine these parts to give a formulation of the magnetostatic problem involving only finite sized domains.

5.1.1 The potential splitting

First we need to introduce some notation: we write $f^{\text{int}}(\mathbf{x}) = \lim_{\mathbf{x}\to\Gamma} f(\mathbf{x})$ from inside the magnetic domain, $f^{\text{ext}}(\mathbf{x}) = \lim_{\mathbf{x}\to\Gamma} f(\mathbf{x})$ from outside the magnetic domain. That is, f^{int} denotes the value of f "close" to the boundary Γ and "just inside" the magnetic domain and f^{ext} for the value of f "just outside" the magnetic domain (see Figure 5.1).

From Section 2.2.2 we have certain *physical* conditions on the magnetostatic potential ϕ. Both inside and outside the magnetic domain, Ω, we have

$$\nabla^2\phi(\mathbf{x}) = \nabla\cdot\mathbf{m}(\mathbf{x}), \tag{5.1}$$

where we define $\mathbf{m} = \mathbf{0}$ outside of magnetic materials. We also have the boundary conditions

$$\frac{\partial\phi^{\text{int}}(\mathbf{x})}{\partial\hat{\mathbf{n}}} - \frac{\partial\phi^{\text{ext}}(\mathbf{x})}{\partial\hat{\mathbf{n}}} = \mathbf{m}\cdot\hat{\mathbf{n}} \qquad \mathbf{x}\in\Gamma, \tag{5.2}$$

and

$$\phi^{\text{int}}(\mathbf{x}) - \phi^{\text{ext}}(\mathbf{x}) = 0 \qquad \mathbf{x} \in \Gamma, \tag{5.3}$$

on the boundary of the magnetic material, Γ. Finally we have the condition that

$$\phi(\mathbf{x}) \to 0 \qquad \text{as } |\mathbf{x}| \to \infty. \tag{5.4}$$

The key observation for this part of the derivation is that (5.1)–(5.4) are linear, *i.e.* if $\phi = u+v$ then $\nabla^2\phi = \nabla^2 u + \nabla^2 v$ and similarly for the boundary conditions. Based on this observation we define a new potential u which satisfies

$$\phi(\mathbf{x}) = u(\mathbf{x}) + v(\mathbf{x}), \tag{5.5}$$

everywhere in $\mathbb{R}^d$ and

$$\nabla^2 u(\mathbf{x}) = \nabla \cdot \mathbf{m}(\mathbf{x}) \qquad \mathbf{x} \in \Omega, \tag{5.6}$$

$$\frac{\partial u^{\text{int}}(\mathbf{x})}{\partial \hat{\mathbf{n}}} = \mathbf{m} \cdot \hat{\mathbf{n}} \qquad \mathbf{x} \in \Gamma, \tag{5.7}$$

$$u(\mathbf{x}) = 0 \qquad \mathbf{x} \in \Omega^c, \tag{5.8}$$

where $\Omega^c = \mathbb{R}^d \backslash \Omega$ is the infinite region of free space outside the magnetic domain. Note that (5.6) and (5.7) give a self contained Poisson Neumann problem for $u \in \Omega \cup \Gamma$, and that (5.8) implies

$$\frac{\partial u^{\text{ext}}(\mathbf{x})}{\partial \hat{\mathbf{n}}} = 0 \qquad \mathbf{x} \in \Gamma. \tag{5.9}$$

Equations (5.1)–(5.4) for ϕ combined with (5.5)–(5.8) for u give a number of conditions that v must satisfy. From (5.1), (5.5) and (5.6) and the fact that the problem is linear we have a differential equation for v:

$$\nabla^2 v(\mathbf{x}) = 0, \tag{5.10}$$

everywhere in $\mathbb{R}^d$. Substituting (5.5) into (5.2) and using (5.7) and (5.9) gives

$$\frac{\partial v^{\text{int}}(\mathbf{x})}{\partial \hat{\mathbf{n}}} - \frac{\partial v^{\text{ext}}(\mathbf{x})}{\partial \hat{\mathbf{n}}} = \frac{\partial u^{\text{ext}}(\mathbf{x})}{\partial \hat{\mathbf{n}}} - \frac{\partial u^{\text{int}}(\mathbf{x})}{\partial \hat{\mathbf{n}}} + \mathbf{m} \cdot \hat{\mathbf{n}} \qquad \mathbf{x} \in \Gamma, \tag{5.11}$$

thus the first boundary condition on v is

$$\frac{\partial v^{\text{int}}(\mathbf{x})}{\partial \hat{\mathbf{n}}} = \frac{\partial v^{\text{ext}}(\mathbf{x})}{\partial \hat{\mathbf{n}}} \qquad \mathbf{x} \in \Gamma. \tag{5.12}$$

Finally from (5.3), (5.5) and (5.8) we have a second boundary condition on v:

$$\begin{aligned} (u^{\text{int}} + v^{\text{int}}) - (u^{\text{ext}} + v^{\text{ext}}) &= 0, \\ v^{\text{int}} - v^{\text{ext}} &= -u^{\text{int}} \qquad \mathbf{x} \in \Gamma. \end{aligned} \tag{5.13}$$

As mentioned previously, the solution of the Laplace equation (5.10) subject to the boundary conditions (5.12) and (5.13) is satisfied by a certain arrangement of virtual magnetic charges. This arrangement of charges is discussed in the next section.

5.1.2 Double Layer Potentials

A double layer potential can be thought of as the potential due to a layer of dipoles (*i.e.* two charges separated by an infinitesimal distance) of magnitude $\mu(\mathbf{x})$ in direction $\hat{\mathbf{n}}$ over the surface Γ [94].[2] In this section we give some (mathematical) properties of a double layer potential.

The double layer potential at a point $\mathbf{x} \in \mathbb{R}^d$ (including $\mathbf{x} \in \Gamma$) is defined as

$$\phi(\mathbf{x}) = \int_\Gamma \mu(\mathbf{y}) \frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial \hat{\mathbf{n}}(\mathbf{y})} \, \mathrm{d}\mathbf{y}, \tag{5.14}$$

where G is the Green's function for the Laplacian operator and $\hat{\mathbf{n}}(\mathbf{y})$ is the outward unit normal at $\mathbf{y} \in \Gamma$. For $d = 2$

$$G(\mathbf{x}, \mathbf{y}) = \frac{-1}{2\pi} \ln(|\mathbf{x} - \mathbf{y}|), \tag{5.15}$$

and for $d = 3$

$$G(\mathbf{x}, \mathbf{y}) = \frac{-1}{4\pi} \frac{1}{|\mathbf{x} - \mathbf{y}|}. \tag{5.16}$$

We have the following results from potential theory:

[2] Note that our potentials are a factor of $\frac{-1}{4\pi}$ different from those in reference [94] and that our ϕ^{int} and ϕ^{ext} definitions correspond to their ϕ^- and ϕ^+.

1. The double layer potential satisfies Laplace's equation $\nabla^2\phi = 0$ (in other words the double layer potential is harmonic) [94].

2. The double layer potential undergoes a jump of $\mu(\mathbf{x})$ moving in the direction $\hat{\mathbf{n}}$ across a smooth surface Γ [94, pp. 136-140], *i.e.*

$$\phi^{\text{int}}(\mathbf{x}) - \phi^{\text{ext}}(\mathbf{x}) = \mu(\mathbf{x}). \tag{5.17}$$

3. Defining the fractional angle at a point as

$$\gamma(\mathbf{x}) = \frac{\alpha(\mathbf{x})}{\alpha_{\max}}, \tag{5.18}$$

where, in two dimensions, $\alpha(\mathbf{x})$ is the angle subtended by the domain at $\mathbf{x}$ and $\alpha_{\max} = 2\pi$. Similarly in three dimensions $\alpha(\mathbf{x})$ is the solid angle and $\alpha_{\max} = 4\pi$.

Then for $\mathbf{x} \in \Gamma$ the following relationship holds [94, pp. 137-139, 155]

$$\phi^{\text{int}}(\mathbf{x}) = (1 - \gamma(\mathbf{x}))\mu(\mathbf{x}) + \phi(\mathbf{x}). \tag{5.19}$$

Note that the angle α at any smooth point on the surface (*i.e.* not at an infinitely sharp corner) is π in 2D (or 2π in 3D). So in these cases $\gamma(\mathbf{x}) = \frac{\alpha(\mathbf{x})}{\alpha_{\max}} = \frac{1}{2}$.

4. If μ is continuous and has continuous first and second derivatives along the boundary (*i.e.* $\mu(\mathbf{x}) \in C^2(\Gamma)$) then the limits $\frac{\partial\phi^{\text{int}}(\mathbf{x})}{\partial\hat{\mathbf{n}}}$ and $\frac{\partial\phi^{\text{ext}}(\mathbf{x})}{\partial\hat{\mathbf{n}}}$ exist and are equal [94, pp. 145-153].

Note that some derivations of the results above rely on the surface being a "Lyapunov surface" which imposes a number of smoothness conditions on the surface, in particular excluding surfaces with corners. The proofs given in [94] allow a finite number of sharp corners, *i.e.* a polygonal domain.

5.1.3 Application to Magnetostatic Calculations

From Section 5.1.2 we can see that conditions (5.10), (5.12) and (5.13) on v are satisfied by a double layer potential with magnitude[3]

$$\mu(\mathbf{x}) = -u^{\text{int}}(\mathbf{x}). \tag{5.20}$$

By the uniqueness of solution for Poisson's equation this gives us, up to an additive constant, the only solution for $v(\mathbf{x})$ in the external domain. This is good enough for a potential since it is only used in $\mathbf{h}_{\text{ms}}(\mathbf{x}) = -\nabla\phi(\mathbf{x})$ and addition of a constant has no effect.

Substituting (5.20) into (5.14) and (5.19) we have:

$$v^{\text{int}}(\mathbf{x}) = -\int_{\Gamma} u^{\text{int}}(\mathbf{y}) \frac{\partial G(\mathbf{x},\mathbf{y})}{\partial \hat{\mathbf{n}}(\mathbf{y})}\, \mathrm{d}\mathbf{y} + \big(\gamma(\mathbf{x}) - 1\big) u^{\text{int}}(\mathbf{x}). \tag{5.21}$$

After substituting our definition for the three dimensional Green's function (5.16) into (5.21) we obtain the same equation as given by Koehler et. al. [62].

Also note that $\phi = u + v$ everywhere, so

$$\phi^{\text{int}}(\mathbf{x}) = -\int_{\Gamma} u^{\text{int}}(\mathbf{y}) \frac{\partial G(\mathbf{x},\mathbf{y})}{\partial \hat{\mathbf{n}}(\mathbf{y})}\, \mathrm{d}\mathbf{y} + \gamma(\mathbf{x}) u^{\text{int}}(\mathbf{x}). \tag{5.22}$$

Either (5.21) or (5.22) can be used to give boundary conditions for $v \in \Omega$ or $\phi \in \Omega$ respectively. We will proceed using (5.22) since it eliminates v from later calculations and is slightly simpler. We also omit the distinction between interior and exterior since all exterior values have now been eliminated.

The final continuous problem is

$$\begin{aligned} \nabla^2\phi(\mathbf{x}) &= \nabla \cdot \mathbf{m}(\mathbf{x}), \\ \nabla^2 u(\mathbf{x}) &= \nabla \cdot \mathbf{m}(\mathbf{x}), \end{aligned} \tag{5.23}$$

[3]It is also possible to directly derive the double layer potential formula from (5.10), (5.12) and (5.13), as described in [60, Appendix 2].

in the magnetic domain Ω, with boundary conditions

$$\begin{aligned} \phi(\mathbf{x}) &= \mathcal{G}\big[u(\mathbf{x})\big] \quad \mathbf{x} \in \Gamma, \\ \frac{\partial u(\mathbf{x})}{\partial \hat{\mathbf{n}}} &= \mathbf{m} \cdot \hat{\mathbf{n}} \qquad \mathbf{x} \in \Gamma, \end{aligned} \tag{5.24}$$

where the operator $\mathcal{G}$ is defined as

$$\mathcal{G}[u(\mathbf{x})] = -\int_{\Gamma} u(\mathbf{y}) \frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial \hat{\mathbf{n}}(\mathbf{y})} \, \mathrm{d}\mathbf{y} + \gamma(\mathbf{x}) u(\mathbf{x}). \tag{5.25}$$

Note that u has purely Neumann boundary conditions, hence it is only defined up to a constant. However, as mentioned before, the addition of a constant is not important so we can fix u to zero at some point to resolve this issue.

So if $\mathbf{m}$ is some fixed function, we can apply the FEM to calculate u, then use $\mathcal{G}$ (assuming we have some discrete version of it) to calculate Dirichlet boundary conditions on ϕ and, finally, use the FEM again to calculate ϕ. However, in the context of an implicit time integration scheme applied to the LLG equation the value of $\mathbf{m}$ is not fixed, and entire problem should be solved simultaneously to find a consistent set of $\mathbf{m}$, u and ϕ. Efficient methods for the solution of such a simultaneous system, or methods which avoid it, are the topic of Chapter 6. For now we continue by showing how the problem can be discretised.

5.2 Discretisation

The bulk equations (5.23) and the Neumann boundary condition on u in (5.24) are identical to those discussed in Section 4.2 and will be solved using the finite element method. As such the weak residual form, discretisation and Jacobians are identical and all that remains to be discretised is the operator $\mathcal{G}$. We let

$$\phi = \sum_i \phi_i \varphi_i(\mathbf{x}), \qquad u = \sum_j u_j \varphi_j(\mathbf{x}), \tag{5.26}$$

where φ_i are the same polynomial basis functions as used in the finite element method of Chapter 4.

Substituting (5.26) into (5.22) we have

$$\sum_i \phi_i \varphi_i(\mathbf{x}) = -\int_\Gamma \sum_j u_j \varphi_j(\mathbf{y}) \frac{\partial G(\mathbf{x},\mathbf{y})}{\partial \hat{\mathbf{n}}(\mathbf{y})}\, \mathrm{d}\mathbf{y} + \gamma(\mathbf{x}) \sum_j u_j \varphi_j(\mathbf{x}). \tag{5.27}$$

Since the integrands are continuous functions on Γ, we can exchange the order of the sum and the integration, leaving

$$\sum_i \phi_i \varphi_i(\mathbf{x}) = -\sum_j u_j \left[\int_\Gamma \varphi_j(\mathbf{y}) \frac{\partial G(\mathbf{x},\mathbf{y})}{\partial \hat{\mathbf{n}}(\mathbf{y})}\, \mathrm{d}\mathbf{y} \right] + \gamma(\mathbf{x}) \sum_j u_j \varphi_j(\mathbf{x}). \tag{5.28}$$

At this point we could apply the Galerkin method (as described in Section 4.1): by multiplying (5.28) by a test function and integrating. This would result in a system of linear equations $\mathrm{M}\tilde{\phi} = \mathrm{G}'\tilde{u}$ (where G' is the Galerkin discrete form of $\mathcal{G}$) which could be solved for $\tilde{\phi}$. However this would require calculating a double integral for each entry in G'. Instead we apply a standard collocation method. Collocation methods enforce pointwise relationships in contrast to the Galerkin method which enforces relationships in a weighted integral form. The collocation method is the conventional choice of discretisation approach for the BEM operator in micromagnetics.

To get the value of ϕ at node k using the collocation method we choose $\mathbf{x} = \mathbf{x}_k$ in (5.28). Then using the property $\varphi_i(\mathbf{x}_k) = \delta_{ik}$ and replacing k by i we have

$$\phi_i = -\sum_j u_j \left[\int_{\Gamma_j} \varphi_j(\mathbf{y}) \frac{\partial G(\mathbf{x}_i,\mathbf{y})}{\partial \hat{\mathbf{n}}(\mathbf{y})}\, \mathrm{d}\mathbf{y} \right] + \gamma(\mathbf{x}_i) u_i, \tag{5.29}$$

where Γ_j is the region of Γ where $\varphi_j \neq 0$, *i.e.* the elements which contain node j. Notice that the expression inside the square brackets is independent of all potentials: it depends only on the geometry and so can be pre-calculated and stored.

The equation (5.29) gives the value of ϕ at a boundary node in terms of a sum of geometric factors multiplied by the values of u at each boundary node. In other words, this is a multiplication by a dense matrix G giving ϕ at all the boundary nodes in terms of u at all boundary nodes:

$$\tilde{\phi} = \mathrm{G}\tilde{u}, \tag{5.30}$$

where

$$\mathrm{G}_{ij} = -\int_{\Gamma_j} \varphi_j(\mathbf{y}) \frac{\partial G(\mathbf{x}_i, \mathbf{y})}{\partial \hat{\mathbf{n}}(\mathbf{y})} \, \mathrm{d}\mathbf{y} + \gamma(\mathbf{x}_i)\delta_{ij}. \tag{5.31}$$

We now convert the normal derivative of the Green's function into a more tangible form. In 3D

$$\frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial \hat{\mathbf{n}}} = \frac{-1}{4\pi} \frac{\partial}{\partial \hat{\mathbf{n}}} \frac{1}{|\mathbf{x} - \mathbf{y}|} = \frac{-1}{4\pi} \hat{\mathbf{n}} \cdot \nabla \Big(\frac{1}{|\mathbf{x} - \mathbf{y}|} \Big). \tag{5.32}$$

Converting to spherical coordinates with the origin at $\mathbf{x}$ ($r = |\mathbf{y} - \mathbf{x}|$, $\hat{\mathbf{r}} = \frac{\mathbf{y}-\mathbf{x}}{r}$) we have[4]

$$\frac{\partial G(r)}{\partial \hat{\mathbf{n}}} = \frac{-1}{4\pi} \hat{\mathbf{n}} \cdot \hat{\mathbf{r}} \frac{\partial}{\partial r} \Big(\frac{1}{r} \Big) = \frac{+1}{4\pi} \frac{\hat{\mathbf{n}} \cdot \hat{\mathbf{r}}}{r^2}, \tag{5.33}$$

$$\frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial \hat{\mathbf{n}}} = \frac{\hat{\mathbf{n}} \cdot (\mathbf{y} - \mathbf{x})}{4\pi \, |\mathbf{y} - \mathbf{x}|^3}. \tag{5.34}$$

Similarly, in 2D we find

$$\frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial \hat{\mathbf{n}}} = \frac{-1}{2\pi} \frac{\partial}{\partial \hat{\mathbf{n}}} (\ln |\mathbf{x} - \mathbf{y}|) = \frac{\hat{\mathbf{n}} \cdot (\mathbf{y} - \mathbf{x})}{2\pi \, |\mathbf{y} - \mathbf{x}|^2}. \tag{5.35}$$

So the boundary element matrix in $d = 2, 3$ dimensions has the entries

$$\mathrm{G}_{ij} = \frac{-1}{2^{(d-1)}\pi} I_{ij} + \gamma(\mathbf{x}_i)\delta_{ji}, \tag{5.36}$$

where

$$I_{ij} = \int_{\Gamma_j} \varphi_j(\mathbf{y}) \frac{\hat{\mathbf{n}}(\mathbf{y}) \cdot (\mathbf{y} - \mathbf{x}_i)}{|\mathbf{y} - \mathbf{x}_i|^d} \, \mathrm{d}\mathbf{y}. \tag{5.37}$$

5.3 Evaluation of the discrete boundary operator

In this section we discuss methods of calculating the entries of the discrete boundary element matrix, G_{ij}. Firstly we discuss the apparent singularity in the integral (5.37), then we give two methods for evaluating these integrals: an analytical method and a numerical method.

[4] In spherical polar coordinates $\nabla = \hat{\mathbf{r}} \frac{\partial}{\partial r} + \hat{\boldsymbol{\phi}} \frac{1}{r} \frac{\partial}{\partial \phi} + \hat{\boldsymbol{\theta}} \frac{1}{r \sin\theta} \frac{\partial}{\partial \theta}$. Obviously $\frac{1}{r}$ has no angular dependence so only the derivative with respect to r is non-zero.

5.3.1 Apparent singularity of the main integral

One factor that should be considered before attempting to evaluate the integral I_{ij} from (5.37) is the apparent singularity when integrating over an element containing the source point $\mathbf{x}_i$. However it is easy to show that there is no singularity when the element is flat. In this case, if $\mathbf{x}_j$ is within the element being integrated over, then we have a factor of

$$\hat{\mathbf{n}}(\mathbf{y}) \cdot (\mathbf{y} - \mathbf{x}_i) \equiv 0, \tag{5.38}$$

in the integrand. For all points $\mathbf{y}$ in the element this term is zero by the definition of $\hat{\mathbf{n}}$, so the integral is zero and the singularity is avoided.

For the case when $\mathbf{x}_i$ is outside the element this is not necessarily true. In particular: elements near sharp corners of a domain may be very close to a node that is not in the plane of the element (*i.e.* such that $\hat{\mathbf{n}}(\mathbf{y}) \cdot (\mathbf{y} - \mathbf{x}_i) \neq 0$), resulting in a *near*-singular integral. This means that care must be taken in order to evaluate the integrals accurately.

5.3.2 Analytical solutions

For linear test/solution basis functions and triangular elements an exact analytical solution for the integral (5.37) was given by Lindholm [68, App. B].[5] Combining equations (2) and (3) from [68] with (5.34) and using our notation for nodal positions, we see that Lindholm's L operator is

$$\begin{aligned} L[U] &= \frac{-1}{4\pi} \int_\Gamma U(\mathbf{y}) \frac{\hat{\mathbf{n}}(\mathbf{y}) \cdot (\mathbf{y} - \mathbf{x})}{|\mathbf{x} - \mathbf{y}|^2} \, \mathrm{d}\mathbf{y}, \\ &= -\mathcal{G}[U] + \gamma(\mathbf{x}) U(\mathbf{x}). \end{aligned} \tag{5.39}$$

Thus $L[\varphi_j]$ integrated over the patch of elements Γ_j with $\mathbf{x} = \mathbf{x}_i$ corresponds exactly to $\frac{1}{4\pi} I_{ij}$, as required in the calculation of $G_{i,j}$.[6]

We now give the analytical solution from Lindholm in the notation used here. Let $A_\triangle$ be the area of the triangle, $\mathbf{z}_{i,\triangle}$ denote the position of the ($i \mod 3$)-th

[5] Note: Lindholm's definition of the 3D Green's function is a factor of -1 different to (5.15) and (5.16).

[6] The minus sign from the different definitions of the Green's function has cancelled with the minus sign from the fact that we are interested in calculating -1 times the operator.

node of triangle $\triangle$, and $\hat{\mathbf{n}}_\triangle$ be the outer unit normal to the plane of the triangle ("outer" with respect to the domain Ω). We write the vector from triangle node k to $\mathbf{x}$ as

$$\mathbf{r}_{k,\triangle}(\mathbf{x}) = \mathbf{z}_{k,\triangle} - \mathbf{x}, \tag{5.40}$$

the vector along an edge of the triangle and

$$\mathbf{s}_{k,\triangle} = \mathbf{z}_{k+1,\triangle} - \mathbf{z}_{k,\triangle}, \tag{5.41}$$

and $\hat{\mathbf{s}}$ for the equivalent unit vector. Then using the third (un-numbered) equation in appendix B of [68], converting to our notation and substituting in values from the rest of the paper where practical we have

$$\begin{aligned}\frac{1}{4\pi} I_{ij} = \sum_{\triangle \in \Gamma_j} \frac{|\mathbf{s}_{k+1,\triangle}|}{8\pi A_\triangle} \Bigg[\Big((\hat{\mathbf{n}}_\triangle \times \hat{\mathbf{s}}_{k+1,\triangle}) \cdot \mathbf{r}_{k+1,\triangle}(\mathbf{x}_i) \Big) \Omega_\triangle(\mathbf{x}_i) \\ - \hat{\mathbf{n}}_\triangle \cdot \mathbf{r}_{1,\triangle}(\mathbf{x}_i) \sum_{l=1}^{3} (\hat{\mathbf{s}}_{k+1,\triangle} \cdot \hat{\mathbf{s}}_{l,\triangle}) P_{l,\triangle}(\mathbf{x}_i) \Bigg],\end{aligned} \tag{5.42}$$

where, as before, Γ_j is the set of triangles that are in contact with the node j. The quantity $P_{l,\triangle}(\mathbf{x})$ is given by

$$P_{l,\triangle}(\mathbf{x}) = \ln \left(\frac{|\mathbf{r}_{l,\triangle}(\mathbf{x})| + |\mathbf{r}_{l+1,\triangle}(\mathbf{x})| + |\mathbf{s}_{l,\triangle}|}{|\mathbf{r}_{l,\triangle}(\mathbf{x})| + |\mathbf{r}_{l+1,\triangle}(\mathbf{x})| - |\mathbf{s}_{l,\triangle}|} \right). \tag{5.43}$$

Finally we need $\Omega_\triangle$, the solid angle subtended by the triangle at $\mathbf{x}_i$:

$$\Omega_\triangle(\mathbf{x}) = \operatorname{sign}(\hat{\mathbf{n}}_\triangle \cdot \mathbf{r}_1) 2 \arccos \left[\frac{|\mathbf{r}_1|\,|\mathbf{r}_2|\,|\mathbf{r}_3| + |\mathbf{r}_1|\,\mathbf{r}_2 \cdot \mathbf{r}_3 + |\mathbf{r}_2|\,\mathbf{r}_1 \cdot \mathbf{r}_2}{\sqrt{2(|\mathbf{r}_2|\,|\mathbf{r}_3| + \mathbf{r}_2 \cdot \mathbf{r}_3)(|\mathbf{r}_3|\,|\mathbf{r}_1| + \mathbf{r}_3 \cdot \mathbf{r}_1)(|\mathbf{r}_1|\,|\mathbf{r}_2| + \mathbf{r}_1 \cdot \mathbf{r}_2)}} \right], \tag{5.44}$$

where we have dropped the $\mathbf{x}$ argument and $\triangle$ index from $\mathbf{r}_{i,\triangle}(\mathbf{x})$ for brevity. The use of $\mathbf{r}_1(\mathbf{x})$ in (5.42) and (5.44) is not a typo: the results are independent of which node in the triangle is used.

A useful open source implementation of this calculation in C is included in magpar [83] and redistributed in Nmag [32].

5.3.3 Adaptive numerical quadrature

The analytical formula given in the previous section is highly accurate but is only applicable to 2D triangular boundary elements, *i.e.* 3D tetrahedral bulk elements. However, it is very useful when testing code to be able to run 2D simulations since the run time can be orders of magnitude smaller. It is also sometimes useful to be able to use structured quadrilateral meshes for simple geometries and for testing purposes. For such meshes an adaptive numerical quadrature able to accurately integrate I_{ij} is appropriate. It is likely that similar analytical results could be derived for each of these cases, but it is simpler and more general to use a numerical approach instead.

As discussed in Section 5.3.1 the integral I_{ij} is non-singular, so no special techniques are needed for its integration (other than what is needed to attain good accuracy). As such an adaptive quadrature method can effectively calculate the integrals. We use a simple adaptive quadrature algorithm as follows:

1 Calculate the integral I_{ij} with a quadrature of order n_1 to obtain I_1;
2 Calculate the integral I_{ij} with a quadrature of order n_2 to obtain I_2;
3 **while** $\left|\frac{I_1-I_2}{I_2}\right| \geq \epsilon_q$ **do**
4 | Set $I_1 = I_2$, $n_2 = 2n_2$;
5 | Calculate the integral I_{ij} with a quadrature of order n_2 to obtain I_2;
6 **end**

The choice to double the quadrature order after each non-convergence in line 3 means that little time is wasted attempting to evaluate near-singular integrals with low order quadratures.

It is possible to optimise this process by using a quadrature scheme which reuses the knots (*i.e.* evaluation points) from the lower order calculations in the more accurate calculations, such as the Clenshaw-Curtis quadrature [98]. However, for simplicity, we have not implemented the reuse of knot values.

The implementation of this algorithm in `oomph-lib` uses standard Gaussian quadrature (as mentioned in Section 4.1.4) with $\epsilon_q = 10^{-8}$, starting orders of $n_1 = 2$, $n_2 = 4$, and a maximum order of 256. These parameters gave good accuracy[7] and sufficiently fast computation time such that the time spent computing these

[7]Errors of below ϵ_q when compared with the analytical formula for triangular elements.

integrals is not a major component of the overall computation time. As such we did not experiment with varying these values.

5.4 Conclusions

In this chapter we have derived a hybrid FEM/BEM approach to calculating the magnetostatic field generated by a magnetic body. This approach is beneficial in that it enforces the boundary condition on the potential at infinity without requiring calculations on an infinite domain. However, it involves a dense matrix of size proportional to the number of boundary nodes, and the process of evaluating the required integrals is more complex that those given by a FEM discretisation.

We have also described general methods for the evaluation of the resulting integrals using numerical quadrature and a less general method for a common case using an analytical formula. There is a large scope for improving the efficiency of the adaptive numerical quadrature described here. However, such optimisations were not investigated because the evaluation of the boundary matrix is not a performance critical component of the coupled LLG-magnetostatics problem since it only needs to be evaluated once for a given mesh.

While we have followed the convention in computational micromagnetics of using a collocation approach to discretise the BEM operator, a Galerkin approach may be more appropriate. In particular, we note that the authors of HLib use a Galerkin approach [14].

Chapter 6

Efficient solution of the linear systems

When we apply implicit time integration schemes, a FEM spatial discretisation and the Newton-Raphson method for linearisation (as discussed in Section 3.4 and Chapter 4) to the LLG equation we are left with the problem of solving a sequence of large sparse linear systems to obtain the approximate solution. With the addition of FEM/BEM for calculation of magnetostatic fields (as discussed in Chapter 5), the linear systems to be solved become larger and more complex. In this this chapter we discuss solution strategies for these systems.

In Section 6.1 we first discuss the solution of the systems required in the calculation of the magnetostatic field assuming the magnetisation is known. We then discuss methods of solving the linear systems for the LLG equation assuming the magnetostatic field is known. This is non-trivial because of the inclusion of the dense BEM matrix.

In Section 6.2 we describe two efficient methods for the solution of the LLG equation with magnetostatics: a "monolithic" (*i.e.* fully coupled) approach and a semi-implicit approach. The semi-implicit approach breaks the problem into a sequence of easily solved linear systems but at the cost of modifying the time integration scheme. The monolithic approach uses efficient iterative methods to solve the full linear system and preserves all properties of the time integration scheme, such as stability and geometric integration properties. In particular the energy conservation property of IMR requires the use of a monolithic approach

(see Section 3.4.9). Except for the simplest preconditioner, (6.11), the preconditioners for the monolithic approach are novel.

Finally, in Section 6.3, we present some numerical experiments demonstrating the effectiveness of the proposed linear solvers for the systems resulting from both the semi-implicit and monolithic approaches.

6.1 Solution of decoupled systems

In this section we consider the solution of the magnetostatic and LLG problems in isolation (*i.e.* the LLG problem is solved assuming that the magnetostatic field is a known function and vice-versa). Such methods are important building blocks for the solution of the coupled system.

Firstly we consider the two linear systems solved in the magnetostatic field calculations: these are both Poisson systems (see (5.23) and (4.62)), which are well studied. One efficent and robust method for solving Poisson systems on unstructured grids is to use the method of conjugate gradients (CG) with algebraic multigrid (AMG) as a preconditioner [50].

Note that the matrices involved in the Poisson systems do not depend on ϕ, u or $\mathbf{m}$ (*i.e.* the Poisson problems are linear). This means that when solving these parts of the problem the Newton-Raphson iteration is not required (or if used it will converge in a single iteration assuming that the linear solve is sufficiently accurate). Also the Poisson matrices only depend on the geometry and so can be computed once and stored for reuse throughout the duration of the simulation (and similarly for the coarsening information required by the AMG preconditioner).

Now we focus on the sequence of linear systems resulting from solving the LLG equation with the Newton-Raphson method, (4.83). There do not appear to be any solvers which are efficient, robust and scale optimally with matrix size in the literature. We try two solvers in our implementation, unfortunately neither are expected to be efficient for large matrix sizes and large time steps.

The first method used is a direct solve by LU decomposition. As discussed in Section 3.6.1 this method is extremely robust but is very slow for large matrices, particularly for problems in three spatial dimensions.

The second method is to use a Krylov solver preconditioned by an incomplete LU decomposition (ILU), inspired by [95]. This method is expected to be efficient for medium sized problems and/or small time steps, especially in 2D or thin film problems where the FEM nodes are less coupled. However the effectiveness is strongly dependent on the diagonal dominance of the matrix. As the volume of the elements decreases or the time step size increases the matrix becomes less diagonally dominant and the effectiveness of the preconditioner will decrease. Hence this method is not expected to scale optimally to fine meshes or to be very robust with respect to varying parameters. This method is used by both Nmag [30] and magpar.[1]

6.2 Solution of the coupled LLG-magnetostatics system

We now consider the solution of the combined LLG and FEM/BEM magnetostatics system. Unlike typical FEM computations, these linear systems are not entirely sparse: the BEM matrix, G, (derived in Section 5.2, (5.31)) appears as a dense block in the Jacobian. In this this section we describe two approaches to efficiently solve the resulting non-linear system despite this issue.

In Section 6.2.2 we describe the first approach which involves replacing our implicit time integration scheme with an ad-hoc semi-implicit version which handles the magnetostatic calculation explicitly. This means that only decoupled solves for u, ϕ and $\mathbf{m}$ are required at each time step, and the dense block, G, only appears as a matrix-vector multiply. However it also means that we are no longer using the time integration methods discussed in Section 3.4.2, instead we are using some new scheme.

The second approach, described in Section 6.2.3, is to construct a linear solver which can efficiently handle the dense block. We present a novel solver which consists of a Krylov solver combined with a preconditioner and an efficient representation of the BEM block. This approach has the benefit that all of the properties of the original time integration schemes carry over to the coupled problem. In particular when using this approach the IMR retains the energy property

[1] See `src/llg/precond.c` line 271 in the magpar version 0.9 source code.

described in Section 3.4.9, which is the most interesting of its geometrical integration properties.[2] Additionally it may be useful for stochastic problems (*i.e.* when thermal fields are involved, see Section 2.3.2), since in this case only a few time integration schemes are known to converge to the correct solution. The implicit midpoint rule is one of these schemes, but any semi-implicit modification is likely to remove this property [25].

Before describing these approaches in more detail, we briefly discuss the implementation of the monolithic coupling between the LLG and magnetostatic problems, and the structure of the resulting Jacobian.

6.2.1 Residuals and Jacobian for the monolithically coupled system

As mentioned in Section 5.2 the pair of potentials used in the FEM/BEM method is similar to that of the simplified magnetostatic potential discussed in Chapter 4. There are two differences: the first is that there is no coupling from the auxiliary potential u to the LLG equation, and hence no corresponding block in the full Jacobian.

The second difference is that there is an additional coupling between the boundary values of u and the boundary values of ϕ. This coupling between the BEM part (a collocation method) and the FEM part (a Galerkin method) is implemented using the following residual vector:

$$s_\Gamma(\tilde{\phi}_\Gamma, \tilde{u}_\Gamma) = \mathrm{G}\tilde{u}_\Gamma - \tilde{\phi}_\Gamma, \tag{6.1}$$

where $\tilde{x}_\Gamma$ denotes the vector of values of x at the boundary nodes. This residual is included as normal in the non-linear solve and simply enforces the condition $\tilde{\phi}_\Gamma = \mathrm{G}\tilde{u}_\Gamma$ from (5.30). Note that we cannot enforce the condition in the standard way for Dirichlet boundary conditions (by pinning the values at the boundary nodes to the appropriate values) because the condition is obtained as part of the solve.

[2] A number of integration schemes can obtain length conservation: Cayley transform methods [66], semi-analytical methods [100] and various types of semi-implicit midpoint rule methods [93, 87, 74] (including the one described in Section 6.2.2).

A natural consequence of (6.1) is that the BEM matrix is a block of the Jacobian:

$$\frac{\partial s_\Gamma}{\partial \tilde{u}_\Gamma} = \mathrm{G}, \tag{6.2}$$

where we have again used the notation that $\frac{\partial \mathbf{a}}{\partial \mathbf{b}}$ is the Jacobian of the derivatives of $\mathbf{a}$ with respect to $\mathbf{b}$, as in (4.2). Also note that the diagonal Jacobian block for the boundary values of ϕ is simply the negative of the identity matrix

$$\frac{\partial s_\Gamma}{\partial \tilde{\phi}_\Gamma} = -\mathrm{I}. \tag{6.3}$$

Combining the above with the Jacobian matrix as derived in Section 4.4.3, the complete Jacobian is

$$\mathrm{J} = \begin{pmatrix} \mathrm{F} & \mathrm{P} & 0 \\ \begin{matrix} \mathrm{Q} \\ 0 \end{matrix} & \begin{matrix} \mathrm{A}_\phi & \mathrm{A}_{\phi\Gamma} \\ 0 & -\mathrm{I} \end{matrix} & \begin{matrix} 0 & 0 \\ 0 & \mathrm{G} \end{matrix} \\ \mathrm{Q} & 0 & \mathrm{A}_u \end{pmatrix}, \tag{6.4}$$

where $\mathrm{A}_{\phi\Gamma} = \frac{\partial \phi_h}{\partial \tilde{\phi}_\Gamma}$ is the Jacobian of the bulk ϕ values with respect to the boundary values. The order of blocks is $\mathbf{m}$, ϕ, u. The blocks corresponding to derivatives of the ϕ residual are broken into sub-blocks for the bulk and boundary values due to the BEM coupling. The overall Jacobian is a square matrix with number of rows $N_r = 5N$, where N is the number of nodes. The LLG block, F, is of size $3N \times 3N$, the Poisson block, A_u, is of size $N \times N$. The smaller Poisson block, A_ϕ, is of size $N_\Omega \times N_\Omega$, where N_Ω is the number of bulk nodes. The identity and BEM matrices are of size $N_\Gamma \times N_\Gamma$, where N_Γ is the number of boundary nodes. If the singularity in the pure-Neumann Poisson problem for u is handled by pinning a single value of u (as mentioned in Section 4.2.1) then the corresponding matrices are one row/column smaller.

For brevity we write the Jacobian, (6.4), as

$$\mathrm{J} = \begin{pmatrix} \mathrm{F} & \mathrm{P} & 0 \\ \mathrm{Q}' & \mathrm{A}'_\phi & \mathrm{G}' \\ \mathrm{Q} & 0 & \mathrm{A}_u \end{pmatrix}. \tag{6.5}$$

It is worth noting that a slightly different Jacobian struture could be obtained by modifying the derivation in Section 5.1.3. Instead of using $\phi = u + v$ to eliminate v, we could use $\mathbf{h}_{\mathrm{ms}} = -\nabla(u + v)$ and eliminate ϕ. This would result in both of the potentials having a P block, but only one of them having a Q block. It would also reduce the diagonal dominance of G. We have not experimented with this alternative formulation.

The linear system to be solved at each Newton step is (see Section 3.5.2)

$$\mathrm{J}\boldsymbol{\delta} = -\mathbf{r}, \tag{6.6}$$

where $\mathbf{r}$ is the vector of the current discrete Newton residuals for $\mathbf{m}$, ϕ and u; and we need to find the Newton update $\boldsymbol{\delta} = [\boldsymbol{\delta}\tilde{\mathbf{m}}, \boldsymbol{\delta}\tilde{\phi}, \boldsymbol{\delta}\tilde{u}]^T$.

6.2.2 The semi-implicit (decoupled) approach

One approach, which avoids solving the complete non-linear system, is to break the monolithic system into three coupled but simpler problems. This can be achieved by using implicit calculations for the LLG parts as normal, but treating the magnetostatic calculations explicitly. The resulting scheme then only requires independent calculations of the magnetostatic field and the magnetisation as described in Section 6.1. Since the magnetostatic fields are typically much weaker than the exchange field we hope that the stability of the scheme will not be greatly reduced.

An outline of the algorithm to compute the step from time t_n to t_{n+1} is as follows:

1 Extrapolate the magnetostatic potential (using ϕ_n, ϕ_{n-1}) to time t_{n+1}, call the result $\hat{\phi}_{n+1}$;
2 Use $\hat{\phi}_{n+1}$ to calculate $\mathbf{m}_{n+1}$ using an implicit time integration scheme (a non-linear solve of the F block);
3 Calculate u_{n+1} using $\mathbf{m}_{n+1}$ (a Poisson solve);
4 Use boundary values of u_{n+1} to calculate the boundary values of ϕ_{n+1} (a matrix-vector multiply with the BEM matrix);
5 Calculate ϕ_{n+1} everywhere using these Dirichlet boundary conditions and $\mathbf{m}_{n+1}$ (a Poisson solve);

A similar method without the extrapolation step been used previously, see *e.g.* [86].

Note that we use an extrapolation of the magnetostatic potential, ϕ, to t_{n+1} rather than computing it using a step of an explicit time integration scheme because there is no time derivative in the equations for ϕ.

For step 1 of the algorithm we require a second order accurate extrapolation in order to ensure that the local truncation error of the overall time integration scheme remains second order. A simple second order extrapolation formula (based on Lagrange interpolation [59, p. 312]) is

$$f(t_{n+1}) = \frac{t_{n+1} - t_n}{t_{n-1} - t_n} f(t_{n-1}) + \frac{t_{n+1} - t_{n-1}}{t_n - t_{n-1}} f(t_n), \tag{6.7}$$

replacing differences in time with the appropriate time steps gives:

$$\hat{\phi}_{n+1} = \frac{-\Delta_{n+1}}{\Delta_n} \phi_{n-1} + \frac{\Delta_{n+1} + \Delta_n}{\Delta_n} \phi_n. \tag{6.8}$$

Note that if we are using the implicit midpoint rule we need to extrapolate ϕ to the midpoint rather than t_{n+1}, hence in this case we replace Δ_{n+1} by $\Delta_{n+1}/2$ in the above equation.

Unfortunately this method requires two initial values for the magnetostatic potential values, and is not self starting. The additional value can be generated by directly calculating the potential if the magnetisation values at two initial times are known. Otherwise the first can be calculated from the initial condition and the potential at a second time can be calculated by taking one step of a monolithic method or a (much smaller) time step using a first order extrapolation formula.

The linear and non-linear systems resulting from the application of this algorithm are exactly those discussed in Section 6.1.

6.2.3 The monolithic approach

The alternative to the semi-implicit approach described above is to find an efficient way to solve the full system (6.4) and (6.6), this is referred to as a fully coupled or monolithic approach. The major difficulty is that the system contains a dense block, meaning that any operations (*e.g.* LU decomposition, multiplication) on J will be significantly slower than for a sparse Jacobian.[3] We use a combination of techniques to reduce this negative effect of the BEM block.

The first and most important technique is the use of a Krylov solver, this has two benefits. Firstly, as mentioned in Section 3.6.2, a Krylov solver (with an effective preconditioner) is typically significantly faster and more memory efficient than a direct solver. Secondly, using a Krylov solver rather than a direct solve or a multigrid-based solver means that all that is required of J is the computation of matrix-vector products. This allows us to split J into separate matrices as

$$\begin{aligned} \mathrm{J} &= \begin{pmatrix} \mathrm{F} & \mathrm{P} & 0 \\ \mathrm{Q}' & \mathrm{A}'_\phi & 0 \\ \mathrm{Q} & 0 & \mathrm{A}_u \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & \mathrm{G}' \\ 0 & 0 & 0 \end{pmatrix}, \\ &= \mathrm{J}_S + \mathrm{J}_D. \end{aligned} \tag{6.9}$$

The matrix-vector product can be computed as

$$\mathrm{J}\mathbf{x} = \mathrm{J}_S\mathbf{x} + \mathrm{J}_D\mathbf{x}, \tag{6.10}$$

i.e. the dense and sparse parts can be stored and used completely independently.

This leads us to the second component of our method: the use of a hierarchical matrix format for the BEM block [14, 37, 60]. This format uses ideas similar to the fast multipole method to use computationally cheap (in both time and memory) but less accurate approximations to the matrix where possible, without compromising overall accuracy. It reduces the computational cost for a

[3] Dense matrix-vector multiplication takes $\mathrm{O}(N_r^2)$ operations (since all N_r^2 matrix entries are used), while LU decomposition takes $\mathrm{O}(N_r^3)$ operations [54, p. 223].

matrix-vector product with the dense block to $\mathrm{O}(N_\Gamma \log N_\Gamma)$. Since the number of boundary nodes N_Γ is typically less than the total number of nodes in the problem this can allow the cost of matrix-vector products of J to be optimal (*i.e.* $\mathrm{O}(N_r)$) depending on the geometry. It also reduces the memory requirements for the block to $\mathrm{O}(N_\Gamma \log N_\Gamma)$. This introduces a small additional error, but it has been shown that the size of the error can be made significantly less than the FEM approximation error by choosing appropriate parameters [60, p. 77]. The use of such formats has recently become fairly common in the application of the hybrid FEM/BEM method to micromagnetics problems.

The third and final component in our efficient solver is a cheap and effective preconditioner for the linear system. Some approaches to constructing such a preconditioner will be discussed in the following section.

Preconditioning strategies

In this section we give a number preconditioning approaches for the monolithic system with the Jacobian (6.4). The aim is to construct a preconditioner which gives an approximately constant number of Krylov solver iterations independently of all problem and discretisation parameters but which is also quick to set up and does not consume a large quantity of memory.

The first preconditioner is very simple: we use a direct solve (using LU decomposition) of the sparse part of J, *i.e.*

$$\mathcal{P}_1 = \mathrm{J}_S = \begin{pmatrix} \mathrm{F} & \mathrm{P} & \\ \mathrm{Q}' & \mathrm{A}'_\phi & \\ \mathrm{Q} & & \mathrm{A}_u \end{pmatrix}. \tag{6.11}$$

This preconditioner, while much more efficient than a direct solve including the dense block, still suffers from all the usual problems associated with a direct solver (*i.e.* expensive in both memory and computation time when N is large). However, we are interested in its properties as a test of the effectiveness of dropping the G' block from the preconditioner.

A number of other FEM/BEM based models use an approach similar to preconditioner $\mathcal{P}_1$ [95], except that the preconditioner is applied using an inner Krylov solver rather than an LU decomposition. This makes the computational cost

feasible for realistic problems, but has two issues: The first is that technically an inner Krylov solve cannot be used as a preconditioner because it does not apply the same operation at each step of the outer solve (since they iterate to convergence rather than using a fixed number of steps) [81]. This issue can be resolved by using an outer Krylov solver known as Flexible GMRES which accounts for this. The second issue is that a full inner Krylov solve is run at *every iteration* of the outer Krylov solve, which is computationally expensive compared to more standard preconditioning approaches (but still cheaper than a direct solve for sufficiently large problems). However we are only interested in $\mathcal{P}_1$ as a test of the effect of dropping the dense matrix on the iteration counts, hence we do not pursue this further.

Our second and third preconditioners are more ambitious: motivated again by the fact that the magnetostatic field is usually significantly weaker than the exchange effective field we drop LLG-magnetostatics coupling blocks in order to make the preconditioner block triangular. It can then be applied by inverting only the diagonal blocks (exactly or approximately) and using block back/forward substitution. The two preconditioners, resulting from dropping different LLG-magnetostatics blocks, are:

$$\mathcal{P}_2 = \begin{pmatrix} \mathrm{F} & & \\ \mathrm{Q}' & \mathrm{A}'_\phi & \\ \mathrm{Q} & & \mathrm{A}_u \end{pmatrix}, \qquad \mathcal{P}_3 = \begin{pmatrix} \mathrm{F} & \mathrm{P} & \\ & \mathrm{A}'_\phi & \\ \mathrm{Q} & & \mathrm{A}_u \end{pmatrix}, \tag{6.12}$$

where the blocks in $\mathcal{P}_3$ can be reordered to give a block triangular matrix but we continue to write them in this order for consistency. However, inverting $\mathcal{P}_2$ or $\mathcal{P}_3$ directly would still require an expensive, in terms of both setup time and memory, direct solve of the three diagonal blocks F, A'_ϕ and A_u. Ideally we want to approximate these inverses by some cheap iterative process instead.

Based on the fact that AMG is known to be an optimal preconditioner for Krylov solves of Poisson problems, we apply an AMG approximation for the A blocks. Using this approximation we define two "semi-inexact" preconditioners (*i.e.* preconditioners where some of the blocks are approximated using iterative methods)

$$\bar{\mathcal{P}}_2 = \begin{pmatrix} \mathrm{F} & & \\ \mathrm{Q}' & \widetilde{\mathrm{A}'_\phi} & \\ \mathrm{Q} & & \widetilde{\mathrm{A}_u} \end{pmatrix}, \qquad \bar{\mathcal{P}}_3 = \begin{pmatrix} \mathrm{F} & \mathrm{P} & \\ & \widetilde{\mathrm{A}'_\phi} & \\ \mathrm{Q} & & \widetilde{\mathrm{A}_u} \end{pmatrix}, \tag{6.13}$$

where we use $\widetilde{x}$ to denote that the block is approximated iteratively and

$$\widetilde{\mathrm{A}'_\phi} = \begin{pmatrix} \widetilde{\mathrm{A}_\phi} & \mathrm{A}_{\phi\Gamma} \\ 0 & -\mathrm{I} \end{pmatrix}, \tag{6.14}$$

i.e. only the Poisson block of A'_ϕ is approximated by AMG, the rest can be applied by block back/forward substitution.

As a final fully-inexact (*i.e.* using only cheap approximations with iterative methods) preconditioner we try approximating the F block using ILU. However we should expect poor performance for large matrix sizes and time steps, similar to the case of pure-LLG solves preconditioned with ILU as discussed in Section 6.1. We define two fully-inexact preconditioners using this approximation in addition to the AMG approximation for the Poisson blocks discussed above:

$$\widetilde{\mathcal{P}_2} = \begin{pmatrix} \widetilde{\mathrm{F}} & & \\ \mathrm{Q}' & \widetilde{\mathrm{A}'_\phi} & \\ \mathrm{Q} & & \widetilde{\mathrm{A}_u} \end{pmatrix}, \qquad \widetilde{\mathcal{P}_3} = \begin{pmatrix} \widetilde{\mathrm{F}} & \mathrm{P} & \\ & \widetilde{\mathrm{A}'_\phi} & \\ \mathrm{Q} & & \widetilde{\mathrm{A}_u} \end{pmatrix}. \tag{6.15}$$

6.3 Numerical experiments

We now test the performance and robustness of the linear solvers described in this chapter. A test of the accuracy and stability of the monolithic and semi-implicit approaches (using the solvers developed here) will be performed in Chapter 8.

6.3.1 Problem specification

In order to perform these experiments we need a representative example problem. We do not use the µmag standard problem #4 because the hybrid FEM/BEM method for magnetostatic calculations is particularly ill-suited for such thin film problems (as discussed in Section 3.3.2). Hence the performance on the standard problem would be irrelevant for typical problems using this method and the computational cost would be unnecessarily large.

Since the majority of practical FEM micromagnetics calculations are in three dimensions our test case should also be 3D. It also must have sharp corners or edges, in order to include any effect of near singular integrals in the FEM/BEM

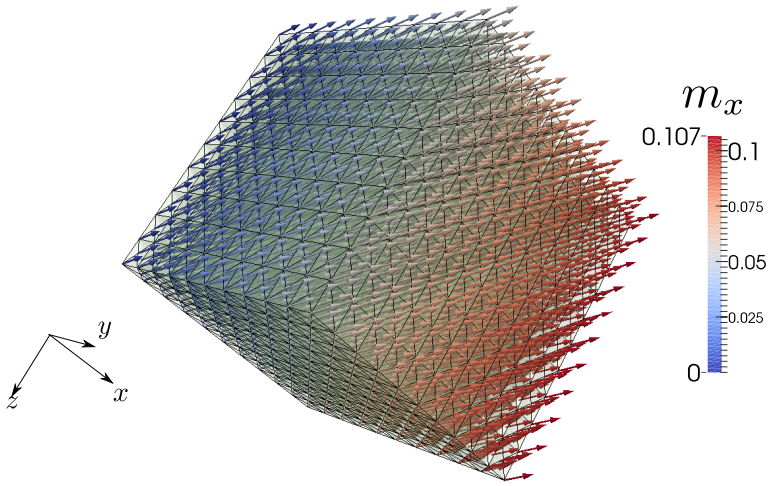

Figure 6.1: The test problem used for the linear solvers in the state at time $t = 0$. The arrows represent the magnetisation direction at each node of the finite element mesh. Colour represents the magnitude of m_x. The semi-transparent pale green grid shows the finite element mesh.

method. Based on these considerations we choose to test the solvers on an $L \times L \times L$ cube with $L = 1$ exchange length.

When testing solvers it is convenient to chose the initial condition such that non-trivial dynamics occur immediately and relevant results can be obtained with only a single time step. As such the initial magnetisation is chosen to be

$$\mathbf{m}_0(x, y, z) = \begin{pmatrix} \sin(2\pi x/50) + \sin(2\pi y/50) \\ \cos(2\pi x/50) + \cos(2\pi y/50) \\ 1.0 - m_y - m_z \end{pmatrix}. \tag{6.16}$$

Note that a wavelength of 50 was chosen to ensure that even meshes with very low refinement are able to resolve the initial state well. The problem geometry and initial conditions are illustrated in Figure 6.1.

We use a range of damping and anisotropy parameters: $\alpha = 0, 0.01, 1$, $\mathcal{K}_1 = 0, 0.1$. We use zero applied field to avoid inducing accidental symmetries.

6.3.2 Implementation details

Time steps of sizes $\Delta_n = 0.01, 0.1, 0.5, 1.0$ are chosen, and meshes with $N = 71, 791, 5631, 42461$ nodes (note that the number of rows/columns in the Jacobian is roughly 5 times the number of nodes). We use both the nodal quadrature discussed in Section 4.5.1 and a standard Gaussian quadrature. Since the only effect of changing the time integration methods on the linear systems is a small change to the constant in the time derivative terms we only run experiments using IMR.

The solvers are run both with and without the use of a hierarchical matrix representation for the dense G block. We use the HLib library [13] with patches from Nmag [32] allowing a collocation BEM approach. For compatibility with this library we use a tetrahedral mesh. We use the HCA II algorithm with parameters based on those used by Knittel [60]: $\epsilon_{ACA} = 10^{-5}$, minimum leaf matrix size $n_{\min} = 30$, admissibility criterion $\eta = 0.25$, polynomial interpolation order $p = 4$, and numerical quadrature order $q = 3$. We also use adaptive recompression with $\epsilon = 10^{-3}$.

The CG and GMRES solvers used are `oomph-lib`'s built-in implementations with relative convergence tolerances of 10^{-8}. GMRES with no restarts and left preconditioning is used for to ensure reliable convergence and reliable residual norms. LU decomposition is implemented using the SuperLU package [67]. The AMG implementation used as a preconditioner for Poisson matrices/blocks is Hypre's BoomerAMG [53]. One V(1,1) cycle with Gauss-Seidel smoothing, CLJP coarsening and a connection strength threshold of 0.7 is used. The ILU preconditioner used for LLG matrices/blocks is Hypre's Euclid with one level of fill in and no drop tolerance (ILU without fill-in was also tried but was extremely ineffective).

All experiments are run on a single core (*i.e.* no parallelism is used) of a desktop computer with an Intel Core i7-3820 processor running at 3.6GHz and 16GB of RAM.

6.3.3 Results

Before presenting the results we note that, unless otherwise specified, all figures in this section show data points for all values of Δ_n, N, α and $\mathcal{K}_1$; both the nodal and Gaussian quadrature schemes; and both the dense and hierarchical BEM block

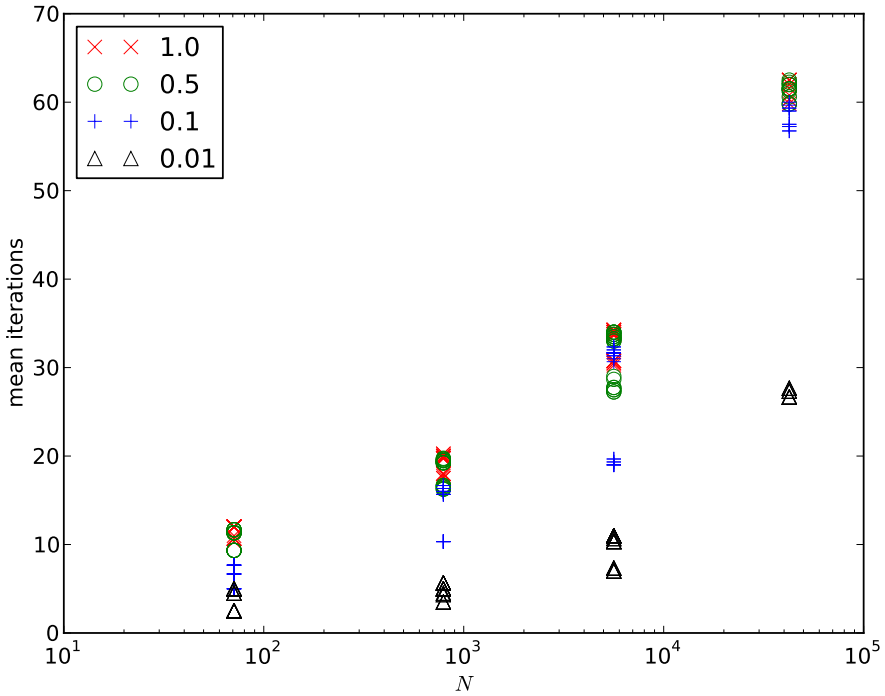

Figure 6.2: Mean (over a single Newton solve) of the GMRES iterations to converge against problem size for the decoupled LLG preconditioned by ILU(1). The legend indicates the time step size. Data points for all values of α and $\mathcal{K}_1$; both types of quadrature; and both formats of the BEM block are shown but are not distinguished.

formats. However, where there appear to be no interesting differences caused by a parameter we do not distinguish between data points with different values of that parameter (*i.e.* we use the same symbol and colour). Also note that occasionally data points overlap, giving the appearance of fewer points.

Figure 6.2 shows the mean (over the linear solves required within a single Newton solve) number of GMRES iterations required to solve the LLG-only system, (4.83), using ILU(1) preconditioning (*i.e.* with magnetostatics handled by the semi-implicit approach). As is expected for a general-purpose preconditioner it is effective for small matrix sizes, but becomes less effective for large problems and as the time step size increases. The parameters α and $\mathcal{K}_1$, along with the BEM block format and quadrature type do not appear to have a major impact on the effectiveness of the ILU(1) preconditioner in this example. The mean preconditioner setup times this preconditioner are shown in Figure 6.3. As with the iteration counts, the setup times become significantly larger as N grows.

Next we consider the monolithic system (6.4) solved using preconditioner $\mathcal{P}_1$,

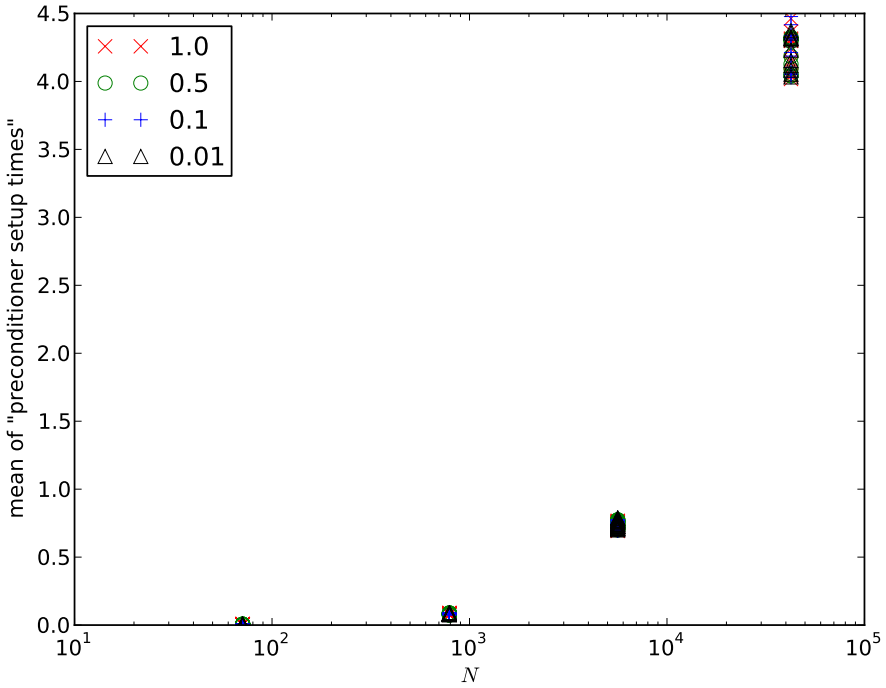

Figure 6.3: Mean (over a single Newton solve) of the time in seconds to set up an ILU(1) preconditioner against problem size for the decoupled LLG block. The legend indicates the time step size. Data points for all values of α and $\mathcal{K}_1$; both types of quadrature; and both formats of the BEM block are shown but are not distinguished.

recall that this uses a direct solve of the sparse part of the Jacobian. The iteration counts are shown in Figure 6.4. Note that they are roughly independent of the number of nodes and vary only slightly with time step and the various other parameters. However the preconditioner setup times, shown in Figure 6.5, rapidly increase with increasing matrix size as would be expected from a direct solve. Also, due to the direct solve, the memory usage of this preconditioner is large. Note that some data for the largest N are missing, this is because the memory required was more than the 16GB available.

Now we consider the partially-inexact preconditioners $\bar{\mathcal{P}}_2$ and $\bar{\mathcal{P}}_3$, which were designed to reduce the setup time and memory issues of $\mathcal{P}_1$ by inverting part of the preconditioner approximately using iterative methods. The iteration counts are shown in Figure 6.6, we see that for both preconditioners the number of iterations increases only slightly as the number of nodes increases and that the various other parameters have little impact on their effectiveness. Also note that the iteration counts are not much larger than those of $\mathcal{P}_1$. The preconditioner

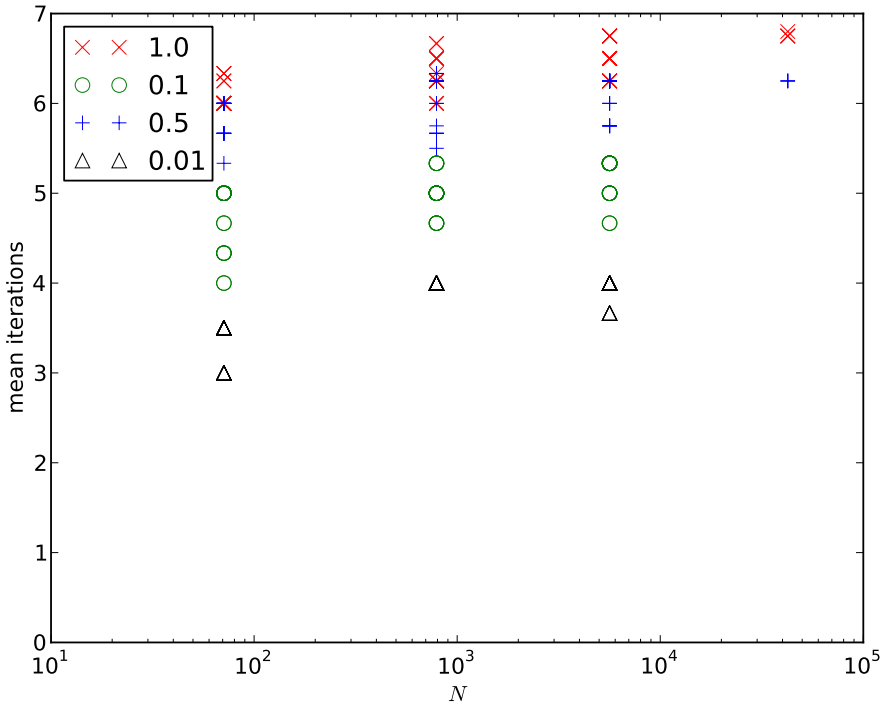

Figure 6.4: Mean (over a single Newton solve) of the GMRES iterations to converge against problem size for the monolithic system preconditioned by $\mathcal{P}_1$ (inverted by LU decomposition). The legend indicates the time step size. Data points for all values of α and $\mathcal{K}_1$; both types of quadrature; and both formats of the BEM block are shown but are not distinguished. Some data points for the largest N are missing due the LU factors requiring more than 16GB of memory.

setup times are shown in Figure 6.7. They are significantly smaller than those in Figure 6.5 but still grow unacceptably large due to the direct solve of the F block. Interestingly the use of nodal integration decreases the time required for the setup of the preconditioner. This is likely due to the "mass-lumping" effect discussed in Section 4.5.1. However it does not decrease the time sufficiently for the preconditioner to be viable for practical usage on problems with number of nodes $N \gtrsim 10^4$. Also of note is that, unlike $\mathcal{P}_1$, the required LU decomposition fits within the 16GB of available memory for all parameters.

Finally we show iteration counts for the fully-inexact preconditioners $\widetilde{\mathcal{P}_2}$ and $\widetilde{\mathcal{P}_3}$ where the F block is approximated using ILU(1). We cannot expect N-independent results here, since even the simpler case of the solution of the F block alone with GMRES preconditioned by ILU(1) does not display such behaviour. The iteration counts shown in Figure 6.8, are as expected: low at first but ineffective for large number of nodes N. In particular, for the largest number of

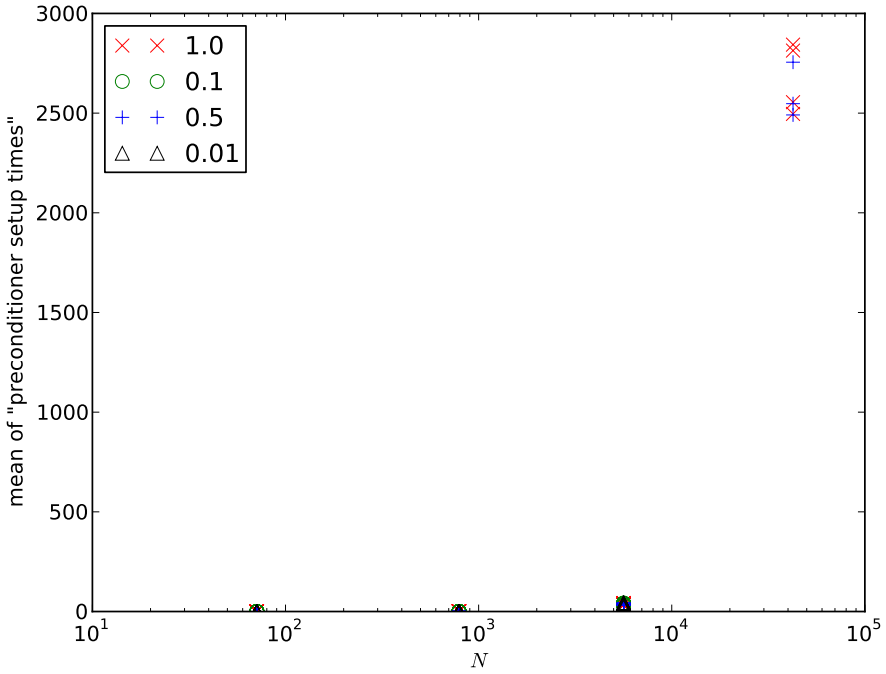

Figure 6.5: Mean (over a single Newton solve) of the time in seconds to set up $\mathcal{P}_1$ (inverted by LU decomposition) against problem size for the monolithic system. The legend indicates the time step size. Data points for all values of α and $\mathcal{K}_1$; both types of quadrature; and both formats of the BEM block are shown but are not distinguished. Some data points for the largest N are missing due the LU factors requiring more than 16GB of memory.

nodes GMRES did not converge within 400 iterations for any of the parameter sets. Also of note is the much larger variation of the iteration counts with the problem parameters even for the same problem size. However the preconditioner setup times, as shown in Figure 6.9 are much better than when using the LU decomposition of the F block.

6.4 Outlook

We now compare the *expected* performance of the monolithic and semi-implicit approaches with the assumption that a "sufficiently good" preconditioner for the F block can be found and discuss further improvements that could be made.

First we examine the relative cost of the assembly of the Jacobian matrices and Newton residuals. As mentioned above the Jacobian matrices corresponding to linear equations are only dependent on the geometry and so do not need to

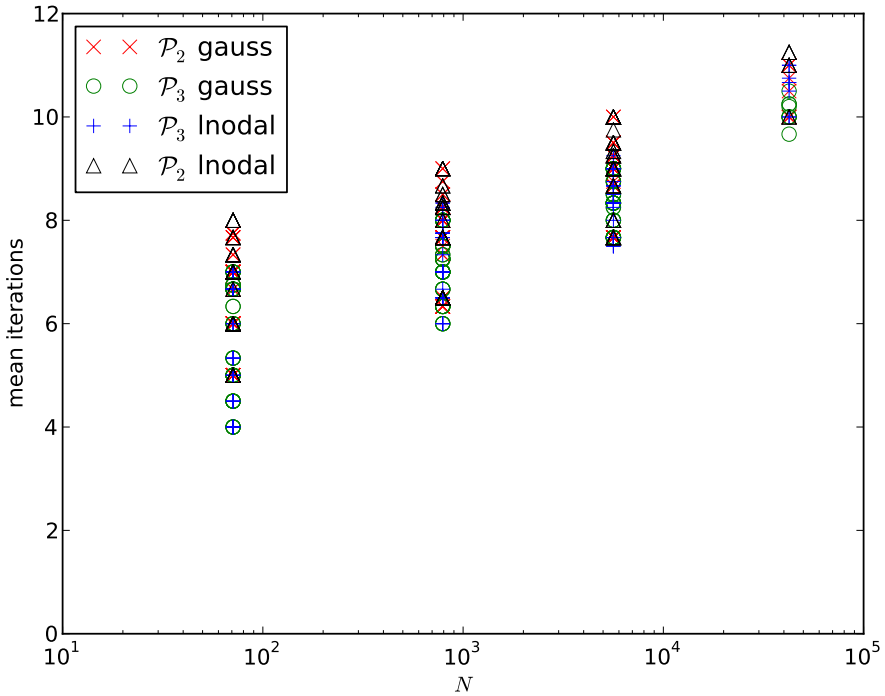

Figure 6.6: Mean (over a single Newton solve) of the GMRES iterations to converge against problem size for the monolithic system using the partially-inexact preconditioners $\bar{\mathcal{P}}_2$, $\bar{\mathcal{P}}_3$ with $\Delta_n = 0.1$. The legend indicates the preconditioner and quadrature scheme used. Data points for all values of α, $\mathcal{K}_1$, and for both formats of the BEM block are shown but are not distinguished.

be recomputed at each Newton step. So the Poisson Jacobians and the LLG-Poisson coupling blocks (Q and P from (6.4)) do not need to be recomputed. This means that for both methods only assembly of the F block is required, and the computation time is identical.

As an aside: the mass matrix blocks on the diagonal of F (M of (4.83)) are also linear. Additionally the skew symmetric structure of (4.83) can be exploited so that the calculation of K_x is reused for $-\mathrm{K}_x$, and similarly for K_y and K_z. Applying these additional optimisations reduces the Jacobian calculations to the assembly of three $N \times N$ Jacobian blocks and the magnetocrystalline anisotropy block (which will typically be either a single $N \times N$ block or empty, see Section 4.4.2).

In practice the Newton residual assembly time is extremely small compared to the solve and Jacobian assembly times ($\sim 99\%$ of the time in the non-linear solver is spent in the Jacobian assembly and linear solves). So the computation time difference due to assembling additional ϕ and u residual components in Newton steps after the first can be ignored.

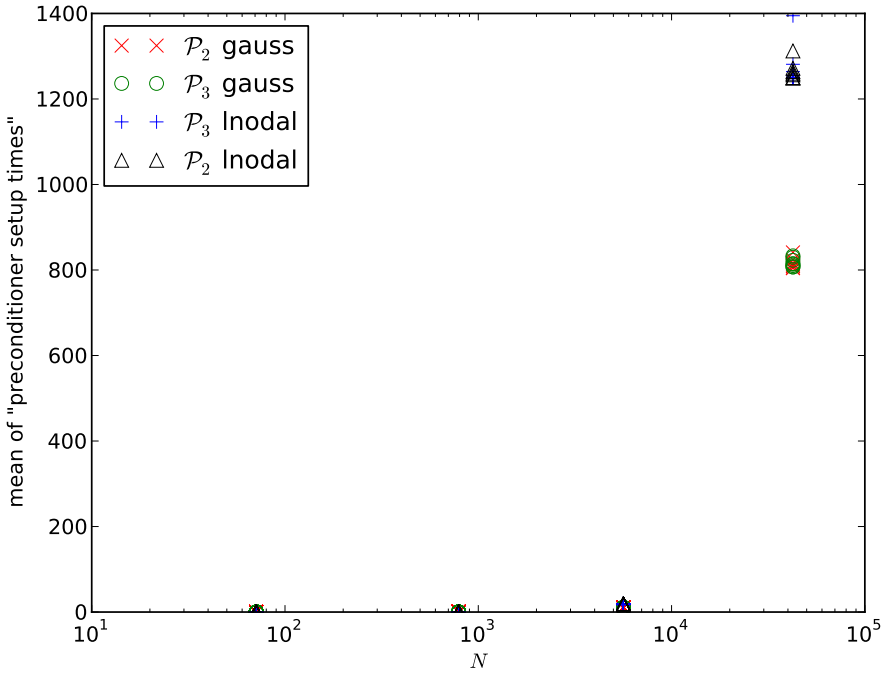

Figure 6.7: Mean (over a single Newton solve) of the time in seconds to set up the partially-inexact preconditioners $\bar{\mathcal{P}}_2$, $\bar{\mathcal{P}}_3$ against problem size for the monolithic system with $\Delta_n = 0.1$. The legend indicates the preconditioner and quadrature scheme used. Data points for all values of α, $\mathcal{K}_1$, and for both formats of the BEM block are shown but are not distinguished.

We now examine the cost per Krylov iteration (or per set of Krylov iterations for the decoupled case). The monolithic approach has more non-zeros in the Jacobian (the P, Q, G blocks), giving approximately $4N$ extra matrix elements, compared to a base of $11N$, hence approximately one third as much time again is taken for each Krylov step. Also our fully coupled method uses GMRES for all blocks rather than GMRES for F and CG for the two Poisson blocks. Since GMRES requires a more computationally expensive orthogonalisation process than CG this will increase the cost of the monolithic approach as compared to the semi-implicit approach. This increase could possibly be reduced by using BiCGStab or a restarted GMRES method instead of GMRES. Finally it is likely that the monolithic method will require more Krylov iterations to converge, but with a good preconditioner for the F block this difference should be fairly small.

Based on all these factors we speculate that, with an effective preconditioner for the LLG block, the computational time for a monolithic time step should be within a factor of 2 of the time for a semi-implicit step.

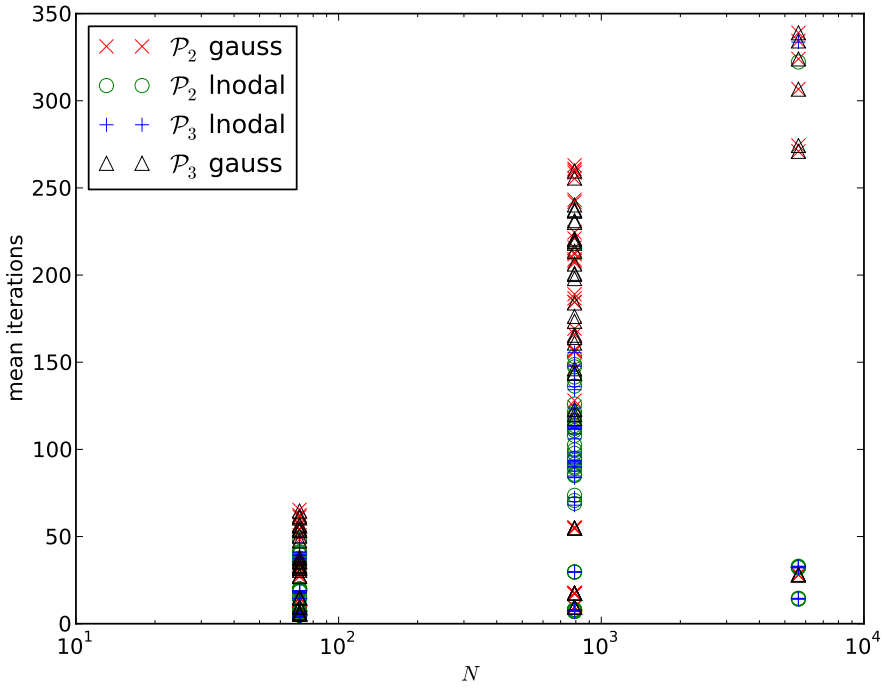

Figure 6.8: Mean (over a single Newton solve) of the GMRES iterations to converge against problem size for the monolithic system using the inexact preconditioners $\widetilde{\mathcal{P}_2}$, $\widetilde{\mathcal{P}_3}$ with $\Delta_n = 0.1$. The legend indicates the preconditioner and quadrature scheme used. Data points for all values of α, $\mathcal{K}_1$, and for both formats of the BEM block are shown but are not distinguished. Some data points are missing for the largest N due to a lack of convergence.

6.5 Conclusions and future work

Monolithic (fully implicit) solvers are required for the energy property of IMR, as well as for other properties of implicit time integration schemes including stability and properties important in stochastic problems. We have demonstrated that a monolithic LLG-magnetostatics solver using GMRES with an effective preconditioner can result in a reasonable computational cost per linear solve. The partially-inexact preconditioners $\bar{\mathcal{P}}_2$ and $\bar{\mathcal{P}}_3$ are able to reduce the iteration count of GMRES extremely effectively and almost independently of the number of nodes, but the set up cost is high for large matrices. However the fully-inexact equivalents, $\widetilde{\mathcal{P}_2}$ and $\widetilde{\mathcal{P}_1}$, which use ILU(1) to approximate the LLG block are ineffective for more than a few thousand nodes (*i.e.* matrices of size $\sim 10,000$). Hence cheap, effective and mesh independent preconditioning methods for the LLG block, F, are still needed for the monolithic block-preconditioned method discussed here to be effective on general large problems.

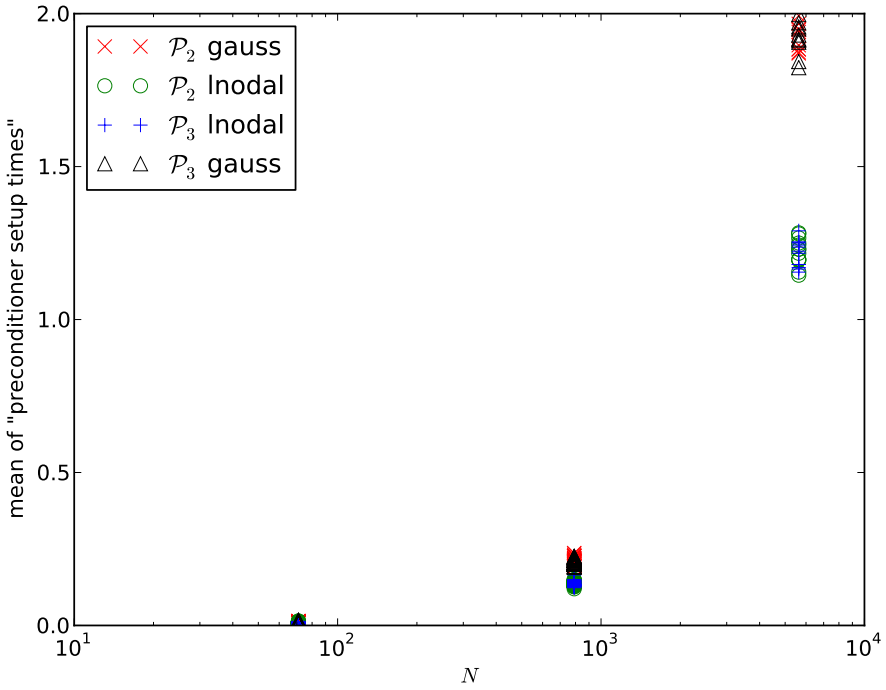

Figure 6.9: Mean (over a single Newton solve) of the time in seconds to set up the inexact preconditioners $\widetilde{\mathcal{P}_2}$, $\widetilde{\mathcal{P}_3}$ against problem size for the monolithic system with $\Delta_n = 0.1$. The legend indicates the preconditioner and quadrature scheme used. Data points for all values of α, $\mathcal{K}_1$, and for both formats of the BEM block are shown but are not distinguished. Some data points are missing for the largest N due to a lack of convergence.

We have also demonstrated solvers for a decoupled LLG block solve, for medium numbers of nodes it appears that ILU(1) is a good preconditioner for this case. However, the time integration properties of the semi-implicit scheme using this solve have yet to be seen.

For granular or bit patterned media a domain-decomposition preconditioner exploiting the low/zero exchange coupling between grains/islands could make for a simple but effective domain decomposition preconditioner. For example, such a preconditioner could be implemented by performing an independent direct solve on the matrix block associated with each grain/island and using this diagonal block solution as the preconditioner. Since the number of nodes in a single grain/island is small and fixed this would remain effective even for very large numbers of grains/islands.

Reordering of the degrees of freedom, scaling, drop tolerance, fill-in level in the ILU preconditioner could be experimented with further to allow the solution of

larger systems or alternative parameters [82, p. 287]. However, ILU preconditioning is unlikely to ever give N-independent or parameter-independent results, so it may not be the best path towards more general efficient preconditioners.

Chapter 7

An adaptive implicit midpoint rule scheme

The implicit midpoint rule (IMR) is a well known integration scheme for initial value ordinary differential equations (ODEs) with favourable stability and accuracy properties (see Section 3.4 or [46, p. 204]). It also has beneficial "geometric integration" properties when applied to the time integration of the Landau-Lifshitz-Gilbert equation (discussed in Sections 3.4.8 and 3.4.9). However it is a non-trivial task to create an efficient adaptive time step selection algorithm for the IMR using typical approaches.

In this chapter we describe a novel adaptive IMR algorithm based on efficient estimates of the local truncation error by an explicit step. We then demonstrate the effectiveness of this algorithm on a number of ODE test problems in Section 7.3 and an ODE form of the LLG in Section 7.4. Experiments on a full PDE LLG problem are performed in Chapter 8.

7.1 The implicit midpoint rule with fixed step size

Let $\mathbf{y}(t)$ be a vector function and denote an approximation to $\mathbf{y}(t)$ at $t = t_n$ by $\mathbf{y}_n$. Let $\Delta_n = t_{n+1} - t_n$ be the nth time step (or just "step"). Then, given a

$$\begin{array}{c|c} \frac{1}{2} & \frac{1}{2} \\ \hline & 1 \end{array}$$

Table 7.1: The Butcher tableau for the implicit midpoint rule.

system of ordinary differential equations (ODEs) of the form

$$\mathbf{y}'(t) = \mathbf{f}(t, \mathbf{y}(t)), \tag{7.1}$$

the implicit midpoint rule (IMR) is

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \Delta_n \mathbf{f}\left(\frac{t_{n+1} + t_n}{2}, \frac{\mathbf{y}_n + \mathbf{y}_{n+1}}{2}\right). \tag{7.2}$$

We introduce shorthand notation for the midpoint values of t and $\mathbf{y}$:

$$\begin{aligned} t_{n+\frac{1}{2}} &= \frac{t_{n+1} + t_n}{2} \quad \left(= t_n + \frac{\Delta_n}{2}\right), \\ \mathbf{y}_{n+\frac{1}{2}} &= \frac{\mathbf{y}_{n+1} + \mathbf{y}_n}{2}. \end{aligned} \tag{7.3}$$

In this notation the IMR is

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \Delta_n \mathbf{f}\left(t_{n+\frac{1}{2}}, \mathbf{y}_{n+\frac{1}{2}}\right). \tag{7.4}$$

The IMR can also be written in the standard form for implicit Runge-Kutta methods as

$$\begin{aligned} k_1 &= \mathbf{f}\left(t_n + \frac{1}{2}\Delta_n, \mathbf{y}_n + \frac{1}{2}k_1\right), \\ \mathbf{y}_{n+1} &= \mathbf{y}_n + 1 \cdot \Delta_n k_1, \end{aligned} \tag{7.5}$$

which is equivalent to the Butcher tableau [46, p. 135] shown in Table 7.1.

Note that, unlike for multistep methods, (7.4) is valid for both constant and variable step sizes because there is no dependence on previous steps.

7.2 Adaptive implicit midpoint rule

Adaptive time step selection algorithms are typically based on approximating the local truncation error (LTE), the error due to a single step of the integration

scheme (see Section 3.4.3). These estimates can be used to select appropriate time steps such that the LTE remains close to a specified value. In this section we first discuss why the standard approaches used to estimate the LTE fail for the IMR, and the approach that we will use. We next derive the formulae required and discuss how a complete adaptive time step selection algorithm can be implemented using the LTE estimates.

7.2.1 Construction of an LTE estimate

Due to the complexity of the LTE of the implicit midpoint rule (as compared with those of TR or BDF2) there are difficulties with the Milne-device approach described in Section 3.4.7. In particular, the LTE of IMR has a term involving the error due to the approximation $\mathbf{y}(t_{n+\frac{1}{2}}) \sim \mathbf{y}_{n+\frac{1}{2}}$ (term II of (3.21)). In order to perform the algebraic rearrangements to obtain the LTE using a Milne-device-like approach we would need this term to appear in the LTE of the predictor. However the term can only appear in the LTE expression for a time integrator which uses the midpoint approximation for $\mathbf{y}$. It turns out that the only such second order time integrator is IMR itself. So other techniques would be needed to approximate this error term in the context of a Milne-device-like method.

An alternative approach, commonly used in Runge-Kutta time integrators, is to repeat the calculation using a higher order method and compare the two answers to directly obtain an LTE estimate [47, p. 165]. However evaluations of the derivative function ($\mathbf{f}$ in (7.1)) are expensive, and the calculation of a single step of a high order Runge-Kutta method requires a number of such evaluations. Hence such approaches usually rely on pairs of Runge-Kutta methods which share most of their derivative evaluation points but have different orders of accuracy. These are known as embedded Runge-Kutta methods, a widely used example is the Dormand–Prince pair (order 4/5) used in MATLAB's `ode45` function [72].

Unfortunately there is no third order method which requires only $\mathbf{f}\left(t_{n+\frac{1}{2}}, \mathbf{y}_{n+\frac{1}{2}}\right)$ and a single other function evaluation.[1] To get around this problem we instead use an explicit multistep method: the explicit version of the third order backwards difference method (eBDF3). This method requires three history values but only

[1] IMR's single function evaluation is positioned such that the second order error terms cancel. Adding one additional evaluation cannot retain the symmetry causing this cancellation and so does not increase the order, at least two additional evaluations are needed.

a single explicit function evaluation in order to compute a 3rd-order-accurate approximation to $\mathbf{y}(t_{n+1})$. Since only one function evaluation is required the computational cost is equivalent to an embedded Runge-Kutta method. With this approach our estimate for the local truncation error is simply

$$T_n = \mathbf{y}_{n+1}^{\text{eBDF3}} - \mathbf{y}_{n+1}^{\text{IMR}} + \mathrm{O}\big(\Delta_n^4\big). \tag{7.6}$$

The eBDF3 method is not widely used because it does not have the basic property essential for time integration [46, p. 365]:

$$\lim_{\Delta_n \to 0} \Big(\max_n ||\mathbf{y}_n - \mathbf{y}(t_n)|| \Big) = 0, \tag{7.7}$$

for large n which is known variously as "stability" [46, p. 378] or "convergence" [54, p. 6]. However, for our usage the IMR history values are used as input for a *single* step of eBDF3, hence the stability is irrelevant.[2] The variable time step version of eBDF3 does not appear in the literature, so we derive the method below in Section 7.2.2 .

Alternatively a 3rd order Adams-Bashforth scheme (AB3) could be used, but this would require three previous derivative values instead of y values making the initialisation of the scheme a little more complex and expensive. Additionally, the calculation of coefficients for the variable step Adams-Bashforth schemes is more complex [46, p. 400]. However, in contrast to eBDF3, AB3 has the basic property (7.7) which could simplify the process of testing an implementation.

7.2.2 The variable step eBDF3 method

The explicit BDF methods are explicit time integration methods derived using the same techniques as the widely used implicit BDF methods. The idea is to write down a divided difference representation of an interpolating polynomial, $\mathbf{p}(t)$, through $\mathbf{y}_{n+1}$ and the appropriate history values of $\mathbf{y}$ (the simpler backward difference representation can be used for the derivation of fixed step methods, hence the name). The derivative of the polynomial is then set equal to the derivative function $\mathbf{f}(t, \mathbf{y})$ at one of the discrete time steps [46, p. 400]. Setting $\mathbf{p}'(t_{n+1}) = \mathbf{f}\big(t_{n+1}, \mathbf{y}_{n+1}\big)$ gives the widely used implicit BDF methods. If we

[2] This could also be thought of as an extrapolation to the value of $\mathbf{y}$ at t_{n+1} using the previous IMR-generated values and a derivative.

instead set $\mathbf{p}'(t_n) = \mathbf{f}(t_n, \mathbf{y}_n)$ we obtain the explicit BDF methods [46, p. 364]. We now derive the first three eBDF methods.

For this purpose we define the Newton divided differences recursively by

$$\begin{aligned} \mathbf{y}[t_{n+1}] &= \mathbf{y}_{n+1}, \\ \mathbf{y}[t_{n+1}, t_n] &= \frac{\mathbf{y}[t_{n+1}] - \mathbf{y}[t_n]}{t_{n+1} - t_n}, \\ \mathbf{y}[t_{n+1}, t_n, t_{n-1}] &= \frac{\mathbf{y}[t_{n+1}, t_n] - \mathbf{y}[t_n, t_{n-1}]}{t_{n+1} - t_{n-1}}, \\ &\vdots \end{aligned} \tag{7.8}$$

The k-th Lagrange interpolation polynomial [19, p. 124], [46, p. 400] can be expressed using these as

$$\begin{aligned} \mathbf{p}_k(t) &= \mathbf{y}_{n+1} + \sum_{j=1}^{k} \mathbf{y}[t_{n+1}, \ldots, t_{n+1-j}] \prod_{i=0}^{j-1} (t - t_{n+1-i}), \\ &= \mathbf{y}_{n+1} + \mathbf{y}[t_{n+1}, t_n](t - t_{n+1}) + \mathbf{y}[t_{n+1}, t_n, t_{n-1}](t - t_{n+1})(t - t_n) \\ &\quad + \mathbf{y}[t_{n+1}, t_n, t_{n-1}, t_{n-2}](t - t_{n+1})(t - t_n)(t - t_{n-1}) + \ldots \\ &\quad + \mathbf{y}[t_{n+1}, \ldots, t_{n+1-k}](t - t_{n+1}) \cdots (t - t_{n-k}). \end{aligned} \tag{7.9}$$

Differentiating (7.9) with respect to t, recalling that divided differences are constants and applying the chain rule to the derivative of the product term we obtain

$$\begin{aligned} \mathbf{p}'_k(t) &= \mathbf{y}[t_{n+1}, t_n] + \mathbf{y}[t_{n+1}, t_n, t_{n-1}]((t - t_n) + (t - t_{n+1})) \\ &\quad + \mathbf{y}[t_{n+1}, t_n, t_{n-1}, t_{n-2}]\big((t - t_n)(t - t_{n-1}) + (t - t_{n+1})(t - t_{n-1}) \\ &\qquad + (t - t_{n+1})(t - t_n)\big) \\ &\quad + \ldots \\ &\quad + \mathbf{y}[t_{n+1}, \ldots, t_{n+1-k}]\big((t - t_n) \cdots (t - t_{n-k}) \\ &\qquad + \ldots + (t - t_{n+1}) \cdots (t - t_{n-k+1})\big), \\ &= \sum_{j=1}^{k} \mathbf{y}[t_{n+1}, \ldots, t_{n+1-j}] \sum_{l=0}^{j-1} \prod_{i=0, i\neq l}^{j-1} (t - t_{n+1-i}). \end{aligned} \tag{7.10}$$

Setting $\mathbf{f}(t_n, \mathbf{y}_n) = \mathbf{p}'_k(t_n)$ results in

$$\mathbf{f}(t_n, \mathbf{y}_n) = \sum_{j=1}^{k} \mathbf{y}[t_{n+1}, \ldots, t_{n+1-j}] \sum_{l=0}^{j-1} \prod_{i=0, i\neq l}^{j-1} (t_n - t_{n+1-i}). \tag{7.11}$$

This can be greatly simplified by noticing that the product $\prod_{i=0,i\neq l}^{j-1}(t_n - t_{n+1-i})$ is zero whenever it contains $i = 1$, *i.e.* whenever $j \geq 2$ and $l \neq 1$. When $j = 1$ the only non-zero terms in the sum of products are for $l = 0$ and $i = 0$, so we only need to consider the case of $l = 1$. This gives

$$\begin{aligned}
\mathbf{f}(t_n, \mathbf{y}_n) &= \sum_{j=1}^{k} \mathbf{y}[t_{n+1}, \ldots, t_{n+1-j}] \prod_{i=0, i\neq 1}^{j-1} (t_n - t_{n+1-i}), \\
&= \mathbf{y}[t_{n+1}, t_n] + \sum_{j=2}^{k} \mathbf{y}[t_{n+1}, \ldots, t_{n+1-j}] \prod_{i=0, i\neq 1}^{j-1} (t_n - t_{n+1-i}), \\
&= \mathbf{y}[t_{n+1}, t_n] - \Delta_n \sum_{j=2}^{k} \mathbf{y}[t_{n+1}, \ldots, t_{n+1-j}] \prod_{i=2}^{j-1} (t_n - t_{n+1-i}).
\end{aligned} \tag{7.12}$$

For $k = 1$ equation (7.12) reduces to

$$\begin{aligned}
\mathbf{f}(t_n, \mathbf{y}_n) = \mathbf{y}[t_{n+1}, t_n] &= \frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{\Delta_n}, \\
\mathbf{y}_{n+1} &= \mathbf{y}_n + \Delta_n f(t_n, \mathbf{y}_n),
\end{aligned} \tag{7.13}$$

which is the forward Euler method.

For $k = 2$ we have

$$\begin{aligned}
\mathbf{f}(t_n, \mathbf{y}_n) &= \mathbf{y}[t_{n+1}, t_n] - \Delta_n \mathbf{y}[t_{n+1}, t_n, t_{n-1}], \\
&= \frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{\Delta_n} - \frac{\Delta_n}{\Delta_n + \Delta_{n-1}} \left(\frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{\Delta_n} - \frac{\mathbf{y}_n - \mathbf{y}_{n-1}}{\Delta_{n-1}} \right).
\end{aligned} \tag{7.14}$$

After some algebra this gives

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \left(1 + \frac{\Delta_n}{\Delta_{n-1}} \right) \Delta_n \mathbf{f}(t_n, \mathbf{y}_n) - \frac{\Delta_n^2}{\Delta_{n-1}^2} (\mathbf{y}_n - \mathbf{y}_{n-1}), \tag{7.15}$$

which is the variable step equivalent of the leapfrog method (explicit midpoint rule) [44, p. 715].

For larger values of k the complexity of the algebra grows enormously. Unfortunately we are interested in the case $k = 3$ which is:

$$\begin{aligned}\mathbf{f}(t_n, \mathbf{y}_n) &= \mathbf{y}[t_{n+1}, t_n] - \Delta_n \mathbf{y}[t_{n+1}, t_n, t_{n-1}] - \Delta_n \mathbf{y}[t_{n+1}, t_n, t_{n-1}, t_{n-2}], \\ &= \frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{\Delta_n} - \frac{\Delta_n}{\Delta_n + \Delta_{n-1}} \left(\frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{\Delta_n} - \frac{\mathbf{y}_n - \mathbf{y}_{n-1}}{\Delta_{n-1}} \right) \\ &\quad - \frac{\Delta_n}{\Delta_n + \Delta_{n-1} + \Delta_{n-2}} \left[\frac{\frac{\mathbf{y}_{n+1} - \mathbf{y}_n}{\Delta_n} - \frac{\mathbf{y}_n - \mathbf{y}_{n-1}}{\Delta_{n-1}}}{\Delta_n + \Delta_{n-1}} - \frac{\frac{\mathbf{y}_n - \mathbf{y}_{n-1}}{\Delta_{n-1}} - \frac{\mathbf{y}_{n-1} - \mathbf{y}_{n-2}}{\Delta_{n-2}}}{\Delta_{n-1} + \Delta_{n-2}} \right].\end{aligned} \tag{7.16}$$

Solving (7.16) for $\mathbf{y}_{n+1}$ by hand would be complex and extremely error prone. Instead (7.16) was solved using the `SymPy` symbolic manipulation library [96], the script used is available online [88]. The resulting time integration scheme is:

$$\begin{aligned}\mathbf{y}_{n+1} &= b\mathbf{f}(t_n, \mathbf{y}_n) + c_n \mathbf{y}_n + c_{n-1} \mathbf{y}_{n-1} + c_{n-2} \mathbf{y}_{n-2}, \\ b &= \frac{\Delta_{n+1}}{\Delta_n (\Delta_n + \Delta_{n-1})} (\Delta_{n+1} + \Delta_n)(\Delta_{n+1} + \Delta_n + \Delta_{n-1}), \\ c_n &= -\frac{(2\Delta_{n+1}\Delta_n + \Delta_{n+1}\Delta_{n-1} - \Delta_n^2 - \Delta_n \Delta_{n-1})}{\Delta_n^2 (\Delta_n + \Delta_{n-1})^2} (\Delta_{n+1} + \Delta_n)(\Delta_{n+1} + \Delta_n + \Delta_{n-1}), \\ c_{n-1} &= \frac{\Delta_{n+1}^2}{\Delta_n^2 \Delta_{n-1}} (\Delta_{n+1} + \Delta_n + \Delta_{n-1}), \\ c_{n-2} &= -\frac{\Delta_{n+1}^2 (\Delta_{n+1} + \Delta_n)}{\Delta_{n-1} (\Delta_n + \Delta_{n-1})^2}.\end{aligned} \tag{7.17}$$

For constant time steps the result reduces to: $b = 3\Delta_n$, $c_n = -3/2$, $c_{n-1} = 3$ and $c_{n-2} = -1/2$ which matches the scheme given in [46, p. 364].

It should be noted that the calculation of the coefficients in (7.17) requires ~ 100 operations (assuming no compiler optimisation of the multiplications). Hence it could be disproportionately expensive if the number of y-values is small. However when used in finite-element or finite-difference calculations $\mathbf{y}_n$ is a vector of 10,000+ values. So the cost of (7.17) is trivial by comparison with, for example, a single calculation of $\mathbf{f}(t_n, \mathbf{y}_n)$.

7.2.3 Computing the next step size

We use a standard method [44, p. 268] for computing the next step size from the ratio of the LTE estimate and a target local truncation error ϵ_Δ

$$\Delta_{n+1} = \left(\frac{\epsilon_\Delta}{T_n}\right)^{\frac{1}{\text{order}+1}} \Delta_n. \tag{7.18}$$

For the implicit midpoint rule this is

$$\Delta_{n+1} = \left(\frac{\epsilon_\Delta}{\left|\mathbf{y}_{n+1}^{eBDF3} - \mathbf{y}_{n+1}^{\text{IMR}}\right|}\right)^{\frac{1}{3}} \Delta_n. \tag{7.19}$$

We also impose some additional constraints to handle extreme cases. If the ratio $\frac{\Delta_{n+1}}{\Delta_n}$ is too small, *i.e.* if the step just taken was much too large to attain the desired accuracy, then we reduce the step size by a factor of 2 and repeat the n-th step. We limit $\frac{\Delta_{n+1}}{\Delta_n}$ to 4 to help improve stability.

Note that some alternative strategies try to minimise the number of modifications to the time step size [54, Chap. 6], in particular this strategy is used in the CVODE library [51, Sec. 2.1]. The motivation for this is the use of a modified Newton-Raphson method where the Jacobian is not recalculated at each Newton step. In the case of ODE solvers this is extremely beneficial, because the Jacobian is usually dense and direct methods are used to solve it – by keeping the Jacobian constant the number of solves and Jacobian assemblies are greatly reduced. The downside is a reduction in the convergence speed of the Newton method. Keeping the step size constant for as long as possible is required here because modifications to the time step cause modifications to the Jacobian. However for PDE solvers, and in particular for PDE solvers using iterative methods to solve the linear systems there is little-to-no benefit in using this technique: the reduction in convergence speed is more detrimental than the gain of not assembling and solving the additional sparse linear systems [54, p. 128].

7.2.4 The complete algorithm

Assuming that we have values for $\mathbf{y}_n$, $\mathbf{y}_{n-1}$, $\mathbf{y}_{n-2}$ (these can be obtained *e.g.* by doing a few steps of fixed step IMR) then an outline of the final adaptive IMR

algorithm is as follows:

1 Set initial conditions and time;
2 **while** $t \leq t_{\max}$ **do**
3 | Set $t := t + \Delta_n$;
4 | Calculate $\mathbf{y}_{n+1}^{\text{eBDF3}}$ from $\mathbf{y}_n$, $\mathbf{y}_{n-1}$, $\mathbf{y}_{n-2}$ using eBDF3, (7.17);
5 | Calculate $\mathbf{y}_{n+1}$ from $\mathbf{y}_n$ using IMR, (7.2), along with some linear/non-linear solver as appropriate;
6 | Calculate Δ_{n+1} from $\mathbf{y}_{n+1}^{\text{eBDF3}}$, $\mathbf{y}_{n+1}$ and Δ_n using (7.19);
7 **end**

Note that we have excluded the modifications for extremely high and low LTE estimates from this description for clarity. Steps 4 and 5 of the algorithm are interchangeable, but in some cases the value $\mathbf{y}_{n+1}^{\text{eBDF3}}$ may be useful as an initial guess for the calculation of $\mathbf{y}_{n+1}$.

7.2.5 Application to Implicit ODEs

We sometimes wish to solve an ODE where $\mathbf{y}'(t)$ is only given implicitly as

$$\mathbf{f}(t, \mathbf{y}(t), \mathbf{y}'(t)) = 0, \tag{7.20}$$

for example in the Gilbert form of the Landau-Lifshitz-Gilbert equation.[3]

However the eBDF3 predictor used in the adaptive scheme requires a derivative evaluation, which for derivatives which can *only* be defined implicitly requires a non-linear solve. This is expensive, so for such cases the adaptive method described here is not efficient. Note that the LLG can be defined explicitly as well as implicitly, so this limitation does not apply to micromagnetics problems (we simply use the Landau-Lifshitz form for the required derivative evaluations).

7.3 Numerical experiments with simple ODEs

In this section we show the results of some experiments with the proposed adaptive IMR algorithm applied to some simple ODE examples. These experiments will

[3]This use of "implicit" is unrelated to the notion of implicitness in the time integration scheme.

allow us to check that the adaptive algorithm selects the correct time steps, and that the errors are commensurate with similar schemes. Numerical experiments with the LLG equation are detailed in Section 7.4.

7.3.1 Implementation details

We compare the adaptive IMR developed above with two commonly used adaptive implicit time integrators:

- The adaptive trapezoid rule (TR) with an error estimate constructed using Milne's device and the second order Adams-Bashforth method (AB2) [44, p. 707].
- Adaptive BDF2, with the error estimate constructed using Milne's device and the variable-step leapfrog method (*i.e.* explicit BDF2) [44, p. 715].[4]

All three methods are implemented within the `oomph-lib` framework. To increase relevance to the general non-linear PDE case (*i.e.* the LLG equation) we use exactly the same methods for the solution of the ODEs as used for semi-discretised PDEs. So we use the Newton-Raphson method with the Jacobian inverted by a linear solver. One effect of this is that the baseline numerical error as seen by the time integration scheme is the tolerance of the Newton-Raphson method rather than the machine accuracy.

Unless otherwise specified all examples in this section use an absolute Newton tolerance of $\epsilon_{\mathrm{N}} = 10^{-8}$, an initial time step of $\Delta_0 = 10^{-5}$, and an adaptive integrator tolerance of $\epsilon_{\Delta} = 10^{-4}$. Any initial conditions required are calculated from the exact solutions.

The BDF2 algorithm used here is extremely similar to the method used by the CVODE library for stiff problems when the order is set to 2 [51][5]. The key difference is in the step selection: CVODE keeps the step size fixed where possible as described in Section 7.2.3.

[4]This method is the default adaptive time integration scheme of `oomph-lib` and has been thoroughly tested over the past ~ 10 years.

[5]By default CVODE allows the order of the method to vary. However previous research has shown that for micromagnetic problems it is beneficial to fix the order to 2 [95].

7.3.2 A polynomial example

The first example is an ODE

$$\begin{aligned} f(t,y) &= 2t, \\ y(0) &= 0.5, \end{aligned} \tag{7.21}$$

with the exact solution

$$y(t) = t^2 + 0.5. \tag{7.22}$$

This problem should be integrated exactly by all of the methods since the local truncation error of all the methods is $T_n \propto y'''(t)$ and for this problem $y'''(t) \equiv 0$. The adaptivity algorithm should detect this and increase the time step at the maximum allowed rate.

The results of this experiment are shown in Figure 7.1. As expected, all three methods keep the error extremely small (at a level commensurate with the Newton tolerance, ϵ_{N}) even for extremely large time steps. The step size increases rapidly for all methods, as expected. The computed solution does not look much like a polynomial because the distance between output points is so large (due to the large time steps).

7.3.3 A damped oscillatory example

The next test features a damped oscillatory solution, the ODE is

$$\begin{aligned} f(t,y) &= -\beta e^{-\beta t}\sin(\omega t) + \omega e^{-\beta t}\cos(\omega t), \\ y(0) &= 0. \end{aligned} \tag{7.23}$$

with the solution

$$y(t) = e^{-\beta t}\sin(\omega t). \tag{7.24}$$

The adaptive algorithm should produce time steps that oscillate in time with the truncation error, while at the same time gradually increase as the term $e^{-\beta t}$ goes to zero.

The ODE was solved with parameters $\omega = 2$ and $\beta = 0.5$, the results are shown in Figure 7.2. All three methods have similar errors and time step sizes. The IMR method chooses slightly smaller steps than TR (which explains the slightly lower error magnitudes). BDF2 has the worst errors as expected because the

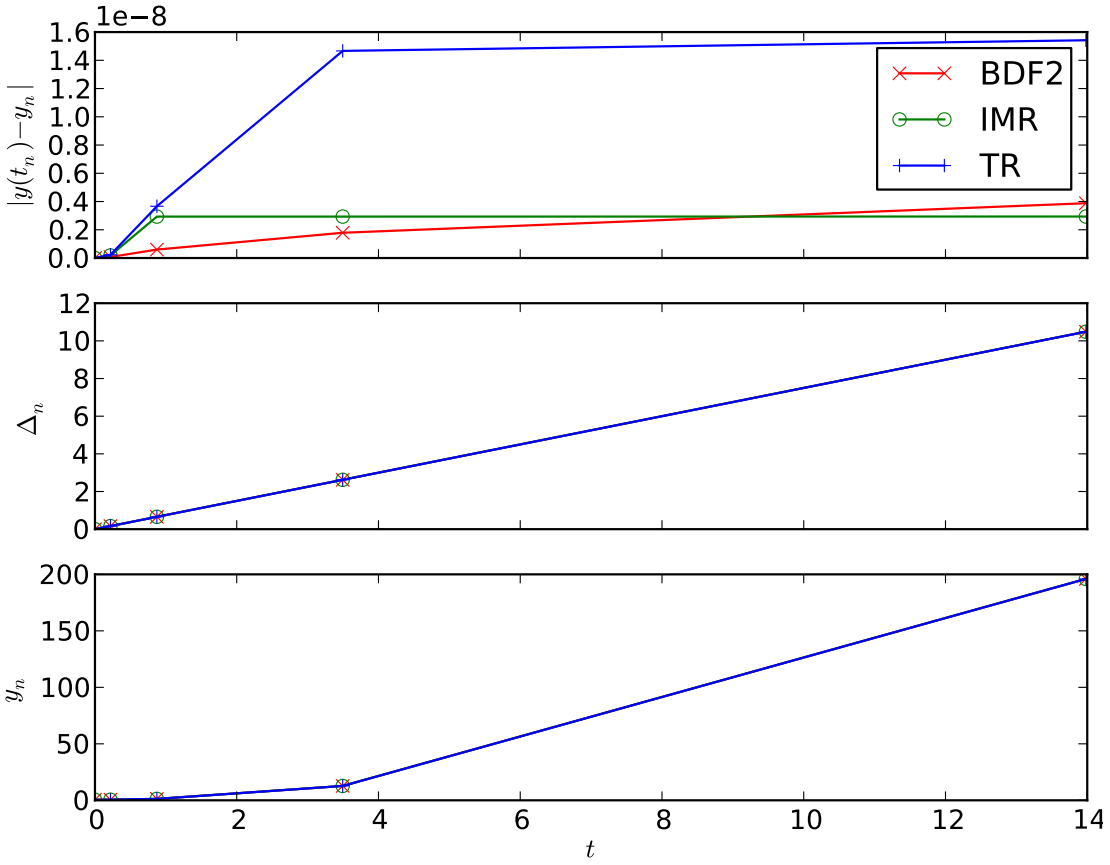

Figure 7.1: Absolute error, time step size and computed solutions for the example ODE with exact solution $y(t) = t^2 + 0.5$.

coefficient of the main LTE term is the largest (see Section 3.4.3).

Note that the plot shows the global error, whereas local truncation error estimates are used for step size selection calculations. Hence the fact that the global error increases above the tolerance, ϵ_Δ, is not surprising.

Interestingly, there is a slight lag on the time step response of the IMR method as time steps become larger. This could be due to the fact that IMR uses data from three previous steps in its LTE estimate, whereas the others only use two previous steps.

In Figure 7.3 we show plots of the decrease in global error norm as the tolerance is decreased. The consistent decrease in the error norms indicates that all of the adaptive methods are able to control the error in the expected manner.

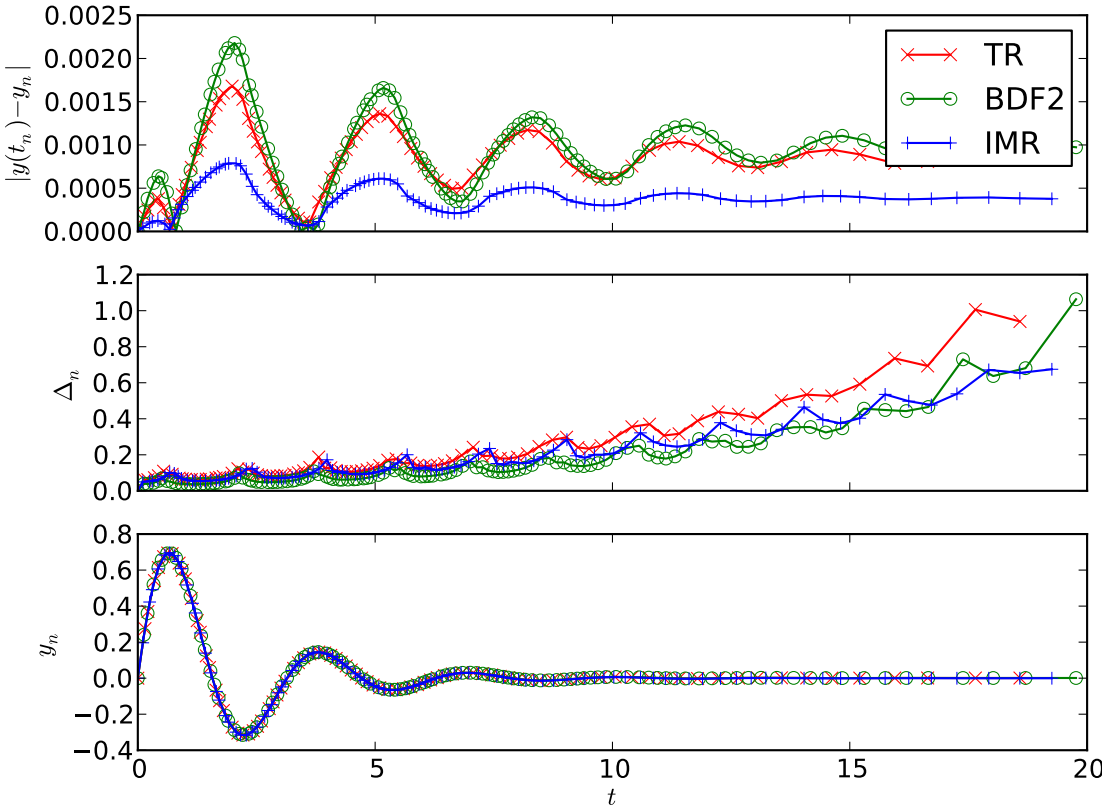

Figure 7.2: Absolute error, time step size and computed solutions for the oscillatory, damped example ODE with exact solution $y(t) = e^{-\beta t}\sin(\omega t)$.

Finally in Figure 7.4 `oomph-lib`'s implementation of the adaptive BDF2 algorithm with VODE's implementation[6]. As noted in Section 7.2.3 VODE prefers to keep time steps constant where possible which results in irregular discontinuous changes in the step size. Despite this difference it can be seen that the amplitude and period of the oscillations in the selected time step sizes is identical, and the overall trend to increasing step size is very similar. `oomph-lib`'s implementation gives slightly lower global errors, probably because of the selection of more appropriate time step sizes.

[6]VODE is an older implementation of CVODE in Fortran rather than C. We use VODE rather than CVODE because it is easily callable from scipy [57]. The newer CVODE has support for iterative linear solvers and parallelism, but uses the same time integration algorithms [51, p. 1].

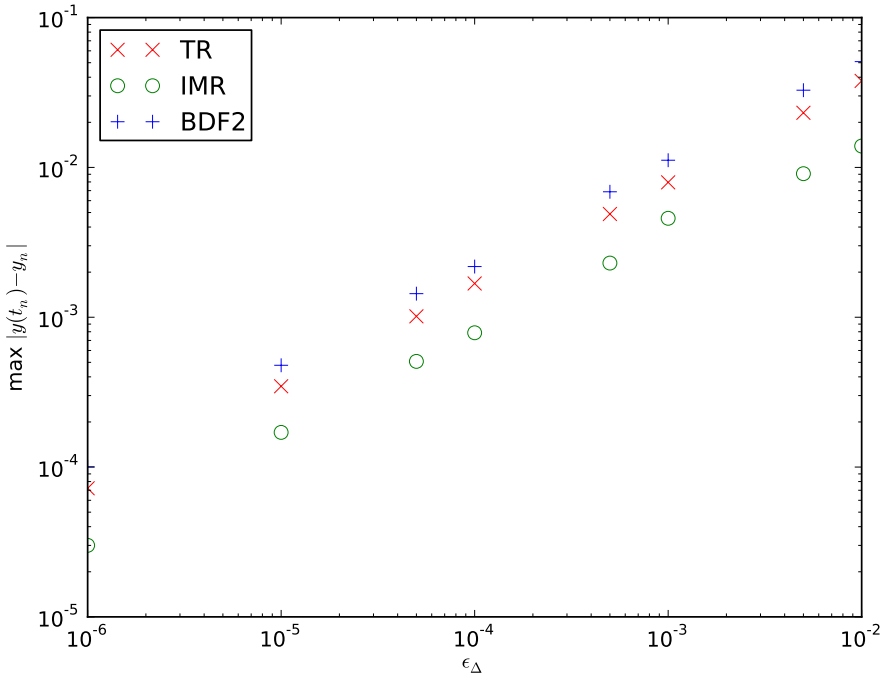

Figure 7.3: Behaviour of the global error norm with varying tolerance for the oscillatory, damped example ODE with exact solution $y(t) = e^{-\beta t}\sin(\omega t)$.

7.3.4 A stiff example

We consider the stiff example ODE used in the definition of A-stability:

$$\begin{aligned} f(t, \mathbf{y}) &= -\lambda \mathbf{y}, \\ \mathbf{y}(0) &= 1, \end{aligned} \tag{7.25}$$

with the exact solution

$$\mathbf{y}(t) = e^{-\lambda t}. \tag{7.26}$$

This example will test that the single step of the eBDF method provides reliable LTE estimates even in cases where the problem is stiff. The time steps should start small but rapidly increase as the exponential decays to zero.

The results of the applying the three methods to problem (7.25) with $\lambda = 100$ and $\mathbf{y}_0 = 1$ are shown in Figure 7.5. The behaviour of the IMR and BDF2 time integrators is essentially the same, indicating that our adaptivity algorithm is effective for stiff problems. The time steps selected by the TR algorithm do not grow as fast as those selected for the other schemes at large t, this is probably due to the build-up of round-off error as noted in *e.g.* [45].

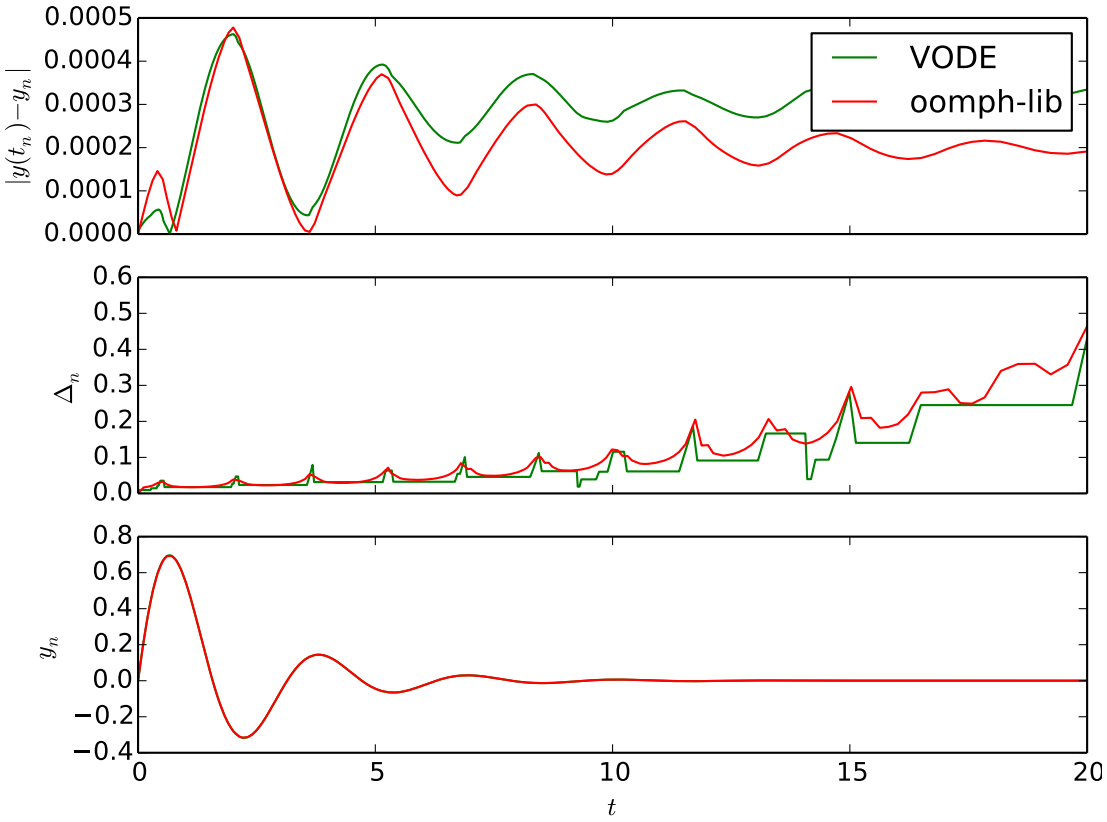

Figure 7.4: Comparison of the `oomph-lib` *and VODE implementations of adaptive BDF2 for the oscillatory, damped example ODE with exact solution* $y(t) = e^{-\beta t}\sin(\omega t)$*. Absolute error, time step size and computed solutions against time are shown.*

7.3.5 Order reduction example

The order reduction effect, discussed in Section 3.4.6, should be automatically detected and adjusted for by a good adaptive time step selection algorithm. So as a final example we study (3.30), the test ODE for order reduction:

$$\begin{aligned} f(t,y) &= -\lambda(y - g(t)) + g'(t), \\ y_0 &= g(0), \end{aligned} \tag{7.27}$$

with exact solution

$$y(t) = g(t). \tag{7.28}$$

For these experiments we chose an oscillatory function, $g(t) = \sin(t)$ and $\lambda = 100$.

Using (3.34), the dominant term of the LTE of IMR for the case when $\Delta_n \gtrsim 1/\lambda$

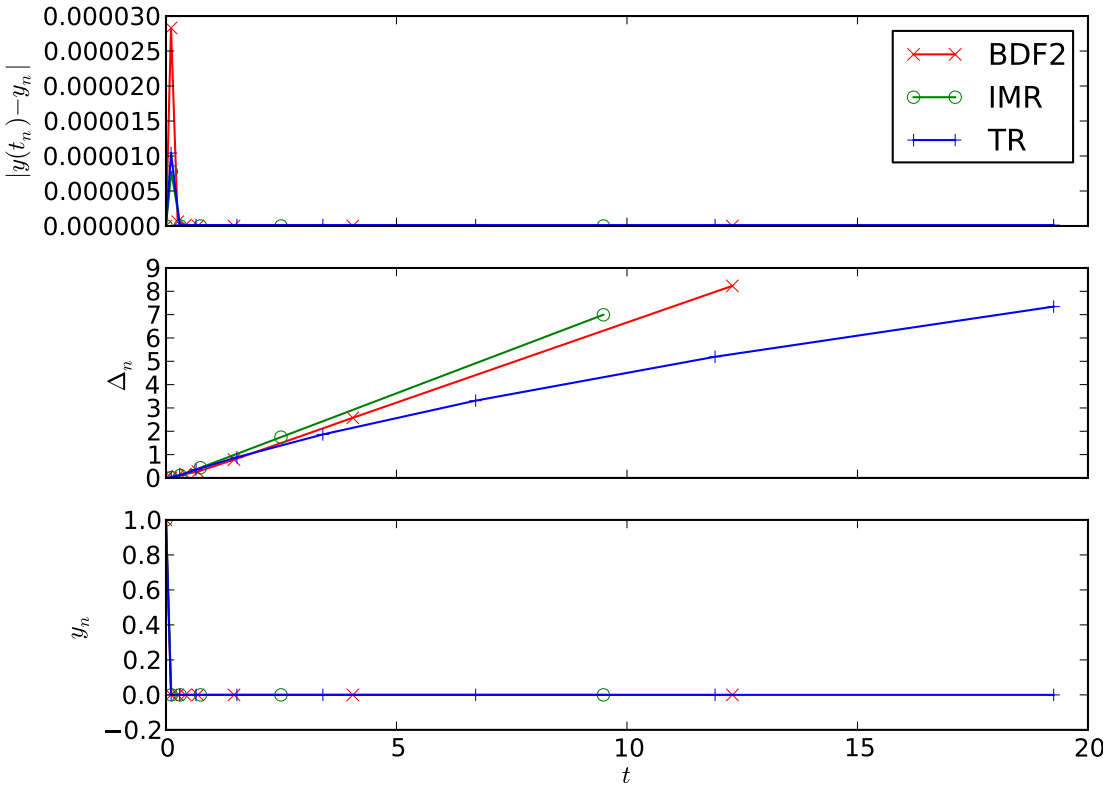

Figure 7.5: Absolute error, step size and computed solutions for the stiff example ODE.

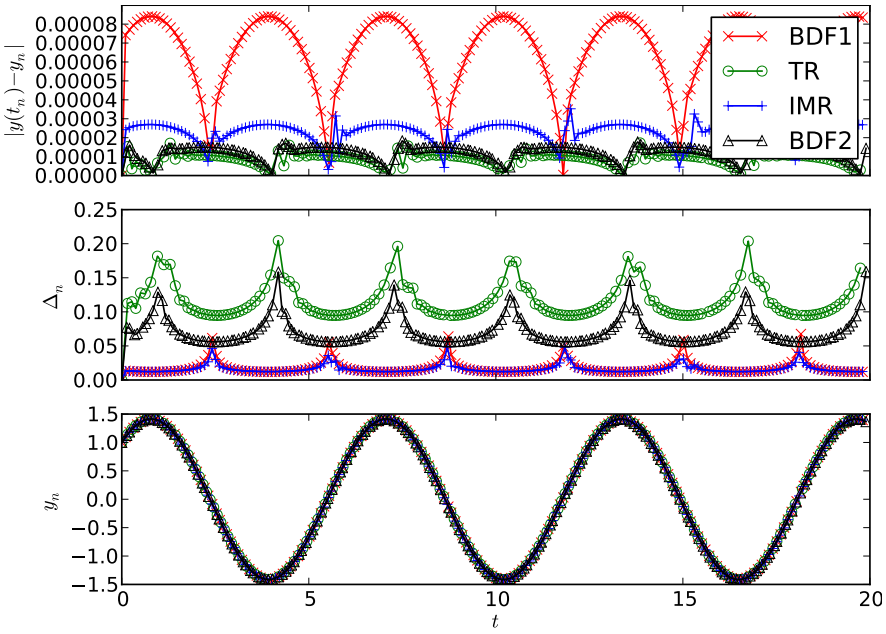

Figure 7.6: Absolute error, step size and computed solutions for the stiff order reduction example ODE given in (3.30).

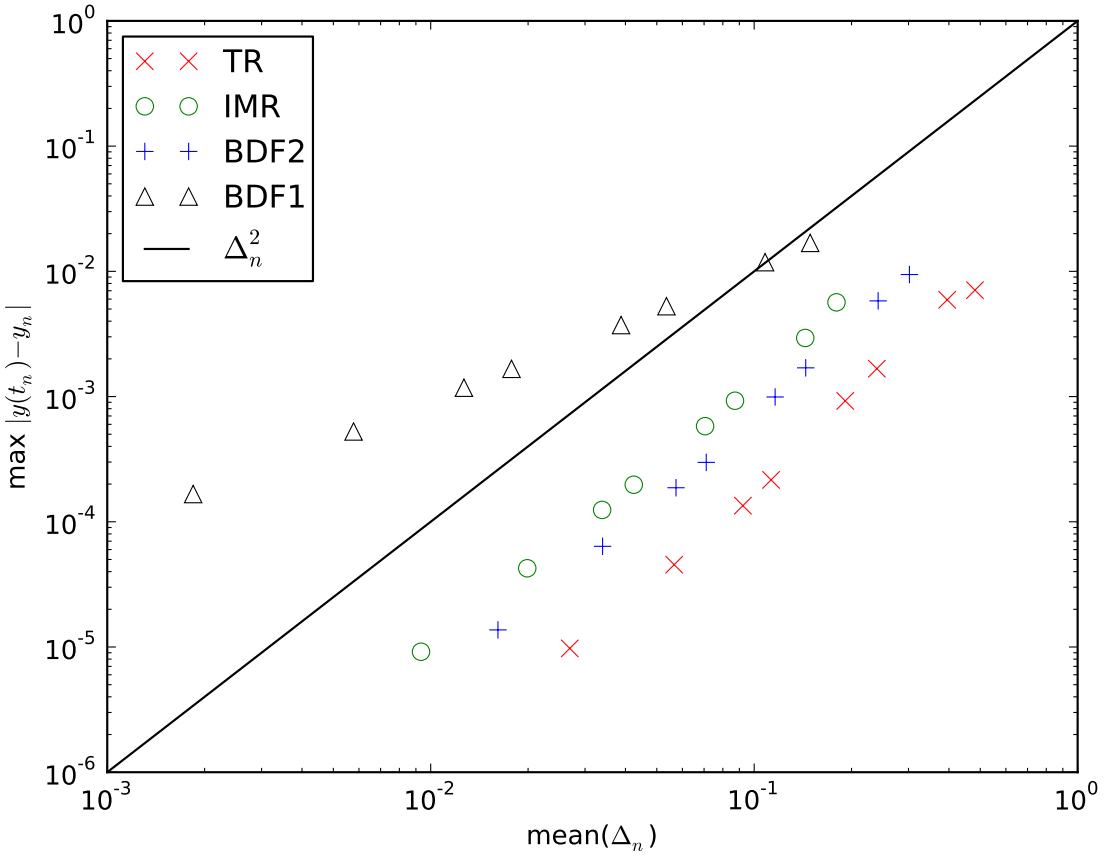

Figure 7.7: Behaviour of the global error norm with varying tolerance vs mean step size for the order reduction example.

is

$$T_n^{\mathrm{IMR}} = \frac{\Delta_n^2}{4}\sin(t_{n+\frac{1}{2}}), \tag{7.29}$$

for smaller Δ_n the $\Delta_n^3\cos(t)$ term will gradually become more important. Note that for TR and BDF2 $T_n \propto \Delta_n^3\cos(t)$, while for BDF1 $T_n \propto \Delta_n^2\sin(t)$. Hence we expect to see the adaptive IMR perform similarly to BDF1, and to select much smaller time steps than TR and BDF2 in order to control the larger errors. To test this we also run the experiment with an adaptive BDF1 (backwards Euler) method using Milne's device (with forward Euler) for adaptivity [44, p. 270].

The results are shown in Figure 7.6. As expected the implicit midpoint rule selects time step sizes very similar to the first order BDF1 method due to order reduction. This allows it to retain similar accuracies to the TR and BDF2 methods at the cost of requiring many more time steps.

The convergence of the solution as the adaptive integrator tolerance, ϵ_Δ, is reduced with fixed $\lambda = 100$ is shown in Figure 7.7. Note that IMR still displays second order convergence (albeit with much larger errors than TR or BDF2). This is because we are not increasing λ as we decrease ϵ_Δ, so we are measuring the normal convergence rather than B-convergence.

7.4 Numerical experiments with an ODE form of the LLG

In this section we present numerical experiments in which we apply the adaptive implicit midpoint rule algorithm to an ODE LLG problem. As in Section 7.3 we compare the performance of the adaptive IMR algorithm developed in this chapter with the adaptive TR and BDF2 algorithms (both with and without renormalisation of the magnetisation after each step). Experiments with PDE LLG problems are postponed until Chapter 8.

7.4.1 Problem definition

To avoid introducing a spatial discretisation we choose an example micromagnetic problem that can be modelled by an ODE: the reversal of a "small" spherical particle under spatially-uniform applied field. With this geometry and with uniform initial magnetisation the magnetostatic field can be analytically shown to be $\mathbf{h}_{\text{ms}} = -\mathbf{m}/3$ throughout the domain [1, p. 112]. This means that the magnetisation remains uniform for spheres small enough that the increase in exchange energy due to any non-uniformity is larger than the corresponding decrease in magnetostatic energy.

In this case the Landau-Lifshitz form of the Landau-Lifshitz-Gilbert equation (2.35) reduces to

$$\begin{aligned} (1+\alpha^2)\frac{\partial \mathbf{m}}{\partial t} &= -\mathbf{m}\times\mathbf{h} - \alpha\mathbf{m}\times[\mathbf{m}\times\mathbf{h}], \\ \mathbf{h} &= \mathbf{h}_{\text{ap}} + \mathcal{K}_1(\mathbf{m}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}}. \end{aligned} \tag{7.30}$$

To simplify the problem even further we choose $\mathcal{K}_1 = 0$ and $\mathbf{h}_{\text{ap}}(t) = [0, 0, -H]$. In this case an exact solution is available for the time taken for $\mathbf{m}$ to reach a

given angle to the z-axis, and for the amount of precession that will take place in that time, more details are given in Appendix A.1 or [70].[7] The exact solution is best expressed in spherical polar notation:

$$\begin{aligned} \theta &= \cos^{-1}(m_z/1), \\ \phi &= \tan^{-1}(m_y/m_x), \end{aligned} \tag{7.31}$$

where θ is the angle between $\mathbf{m}$ and the $+z$-direction. The time for a reversal, τ, from the initial value θ_0 to θ is given by

$$\tau(\theta) = \frac{\alpha^2+1}{H\alpha} \ln\left(\frac{\tan(\theta/2)}{\tan(\theta_0/2)}\right). \tag{7.32}$$

This allows us to calculate a (global temporal) error in the switching time:

$$\mathcal{E}_{\tau n} = |t_n - \tau(\theta_n)|\,. \tag{7.33}$$

Alternatively, (7.32) can be inverted to give θ as a function of time:

$$\theta(t) = 2\tan^{-1}\left[\tan(\theta_0/2)\exp\left(\frac{tH\alpha}{\alpha^2+1}\right)\right]. \tag{7.34}$$

The azimuthal angle at a given θ is [70]

$$\varphi(\theta) = \phi_0 - \frac{1}{\alpha}\ln\left(\frac{\tan(\theta/2)}{\tan(\theta_0/2)}\right). \tag{7.35}$$

After substituting (7.34) into (7.35) and cancelling terms we obtain

$$\varphi(t) = \phi_0 - \frac{tH}{\alpha^2+1}. \tag{7.36}$$

Hence we can also calculate a (global temporal) error in the magnetisation at time t_n

$$\mathcal{E}_{\mathbf{m}n} = |\mathbf{m}_n - \mathbf{m}(t_n)|\,, \tag{7.37}$$

where $\mathbf{m}(t_n)$ is the Cartesian representation of (7.34), (7.36).

[7] Actually the solution for the switching time can be easily extended to cases with $\mathcal{K}_1 \neq 0$ with $\hat{\mathbf{e}} = \hat{\mathbf{z}}$ and/or to any z-aligned ellipsoid of rotation. However it is harder to invert the solution to obtain $\theta(t)$ in these cases.

The appropriate choice of error norm $\mathcal{E}_\tau$ or $\mathcal{E}_\mathbf{m}$ depends on the aim of the calculation. We may simply want to find the switching time of the system, in this case the time error norm, $\mathcal{E}_\tau$, is appropriate (and using time steps such that the precessional behaviour is fully resolved may be excessively expensive). In other cases an accurate representation of the full behaviour of the system may be needed and $\mathcal{E}_\mathbf{m}$ is appropriate.

The only energy term able to vary for this problem is the Zeeman energy (the energy due to the applied field), so the total energy is simply given by

$$E = -\mathbf{m} \cdot \mathbf{h}_{\text{ap}} = H m_z. \tag{7.38}$$

For the main experiment we choose parameters $H = 1.1$ (sufficient to switch the magnetisation) and $\alpha = 0.01$ (a physically relevant value). We set the initial magnetisation to be just off the z-axis in order to avoid the singularity in reversal time when $\theta_0 = 0$ (see (7.32)), more precisely we set $\mathbf{m}_0 = (0.01, 0.0, 1.0)/\sqrt{1.0001}$. The dynamics are simulated for 1000 time units for all experiments, which is sufficient time for the magnetisation to fully switch. This experiment allows us to check the $|\mathbf{m}|$ conservation properties of the adaptive IMR, to compare the accuracy of the solution between different time integrators, and to check again the robustness of the time step selection algorithm.

We also run the same experiment with $\alpha = 0$. This allows us to check the energy conservation properties of adaptive IMR and to observe the what effect the re-normalisation of the magnetisation has on the energy. Note that with $\alpha = 0$ (7.30) becomes simply $\frac{\partial \mathbf{m}}{\partial t} = -\mathbf{m} \times \mathbf{h}_{\text{ap}}$, which is linear in $\mathbf{m}$ and t, hence TR and IMR are identical for this example.

7.4.2 Implementation details

The implementation of the three time integration algorithms without the re-normalisation of $\mathbf{m}$ is exactly as discussed in Section 7.3.1. We also test the performance of the TR and BDF2 methods with an added re-normalisation step to enforce constant $|\mathbf{m}|$.

In Appendix D we compare re-normalisation after every step with two other re-normalisation methods (the self-correcting LLG and re-normalisation only after

the error in the magnetisation length reaches a specified tolerance). We find that there is no reason to prefer these alternative methods. Hence for the rest of this thesis we use the re-normalisation after every step method when re-normalisation is required. It is carried out by simply replacing $\mathbf{m}_{n+1}$ by $\mathbf{m}_{n+1}/\left|\mathbf{m}_{n+1}\right|$ (note that this is done after the selection of the next time step to avoid complicating the step selection process). In the experiments we compare the performance with and without re-normalisation where appropriate.

In the experiments below the Landau-Lifshitz form (2.39) of the LLG is used for ease of implementation. Linearisation is carried out using the Newton-Raphson method. Using the skew operator notation from Section 4.4.2 the Newton-Raphson residual for the Landau-Lifshitz form of the LLG is is

$$\mathbf{r} = (1+\alpha^2)\frac{\partial \mathbf{m}}{\partial t} + \Lambda\left[\mathbf{m}\right]\cdot\mathbf{h} + \alpha\Lambda\left[\mathbf{m}\right]\Lambda\left[\mathbf{m}\right]\cdot\mathbf{h}. \tag{7.39}$$

The corresponding Jacobian can be easily derived using the methods discussed in Section 4.4.2 (note that here $\mathbf{h} \neq \mathbf{h}(\mathbf{m})$) and is given by

$$\mathrm{J} = \frac{\partial \mathbf{r}}{\partial \mathbf{m}} = (1+\alpha^2)J_{ts}\mathrm{I}_3 - \Lambda\left[\mathbf{h}\right] - \alpha\Lambda\left[\mathbf{m}\times\mathbf{h}\right] - \alpha\Lambda\left[\mathbf{m}\right]\Lambda\left[\mathbf{h}\right]. \tag{7.40}$$

The 3×3 Jacobian is calculated analytically and inverted using a direct solve. Unless otherwise specified the Newton tolerance is $\epsilon_{\mathrm{N}} = 10^{-12}$ (to obtain accurate geometric integration properties) and the adaptive integrator tolerance is $\epsilon_{\Delta} = 10^{-4}$. The initial time step is $\Delta_{\mathrm{init}} = 10^{-3}$ which is sufficiently small to allow the adaptive algorithm to naturally increase Δ_n to an appropriate value.

7.4.3 Numerical results

The behaviour of the magnetisation and the time step selection using the adaptive IMR algorithm developed in this chapter is shown in Figure 7.8. The solutions given by adaptive BDF2 and TR (with $|\mathbf{m}|$ re-normalised after each time step) are shown in Figures 7.9 and 7.10 respectively. The overall behaviour of the IMR time step selection algorithm is seen to be working as expected: large steps are chosen at the start when very little switching occurs, the step size decreases as switching occurs and finally grows again as the switching finishes.

The small periodic peaks in the time step size are due to the precession: in

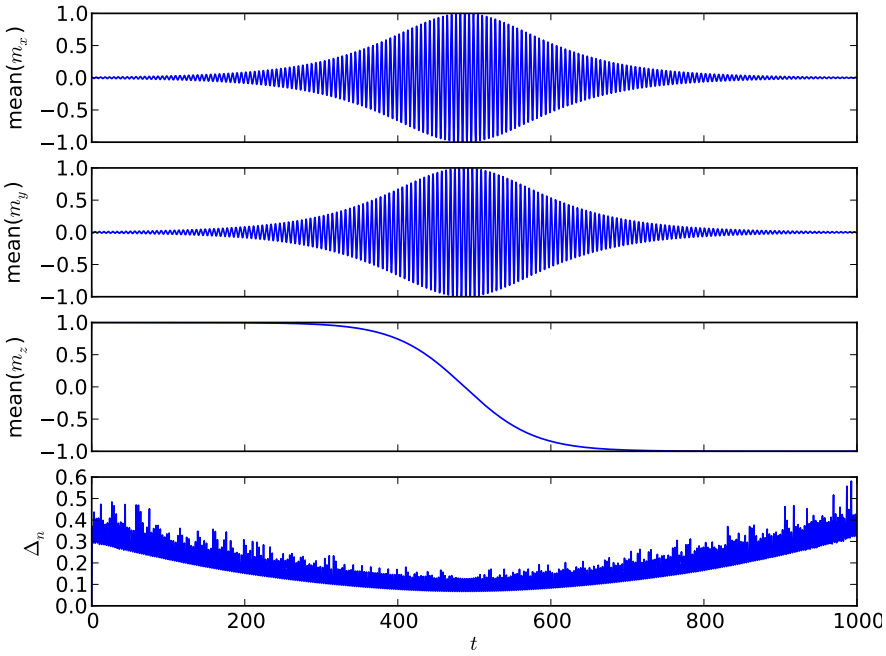

Figure 7.8: Plot of $\mathbf{m}$ *and* Δ_n *over time for the relaxing nano-sphere problem solved by adaptive IMR.*

Cartesian form (but not in the spherical polar coordinate form (7.36)) the precessional part of the solution is written as a sin/cosine term. This causes periodic oscillations in the LTE.

Note, from Figure 7.9, that the switching time obtained by BDF2 with the same error tolerances is very different to that given by IMR or TR. This is due to BDF2's spurious numerical damping (see Section 3.4.5), which initially moves the solution towards the unstable fixed point at $\mathbf{m} = [1, 0, 0]$. The TR solution is qualitatively good. The time steps selected by the three algorithms are similar, TR selects slightly larger steps as would be expected due to its lower local truncation error (see Section 3.4.3).

The behaviour of the magnetisation length over time is shown in Figure 7.11, where we write

$$\mathcal{E}_{|\mathbf{m}|}(t_n) = ||\mathbf{m}_{j,n}| - 1| \,. \tag{7.41}$$

The maximum error when IMR is used is $\mathcal{E}_{|\mathbf{m}|} \approx 10^{-12}$, which is consistent with the Newton tolerance. In contrast the error in the magnetisation length with the TR and BDF2 schemes reaches $\sim 10^{-2}$.

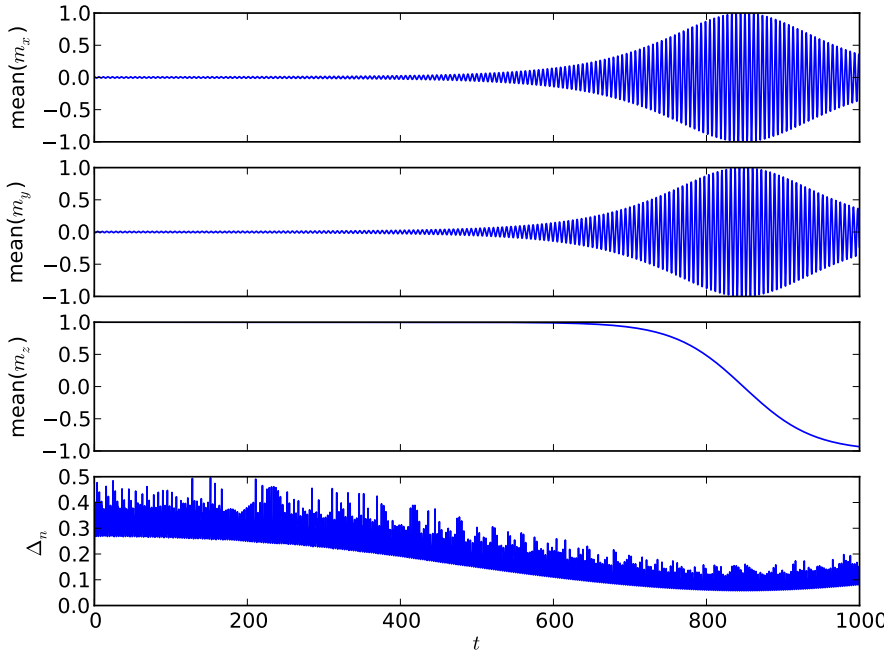

Figure 7.9: Plot of $\mathbf{m}$ *and* Δ_n *over time for the relaxing nano-sphere problem solved by adaptive BDF2.*

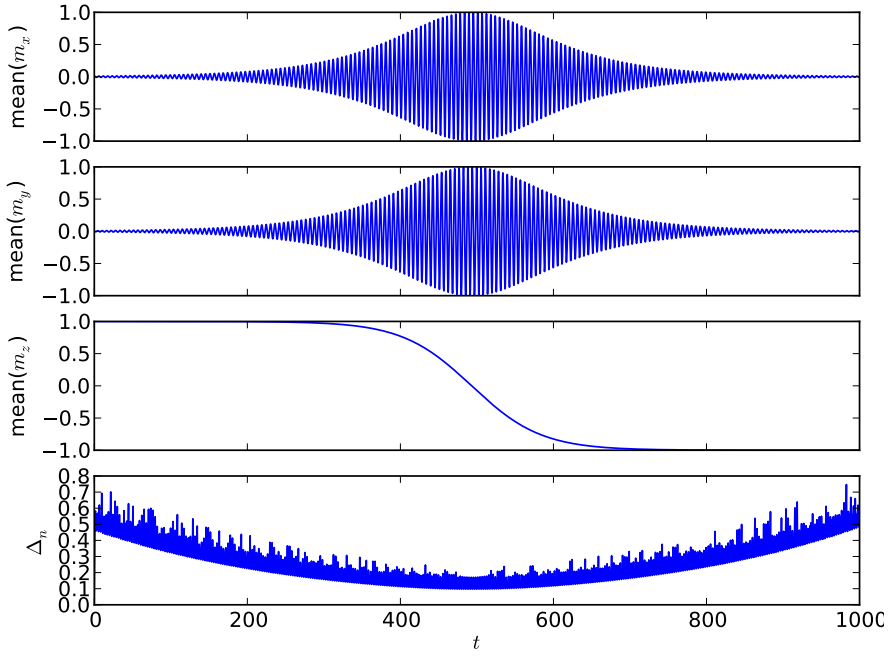

Figure 7.10: Plot of $\mathbf{m}$ *and* Δ_n *over time for the relaxing nano-sphere problem solved by adaptive TR.*

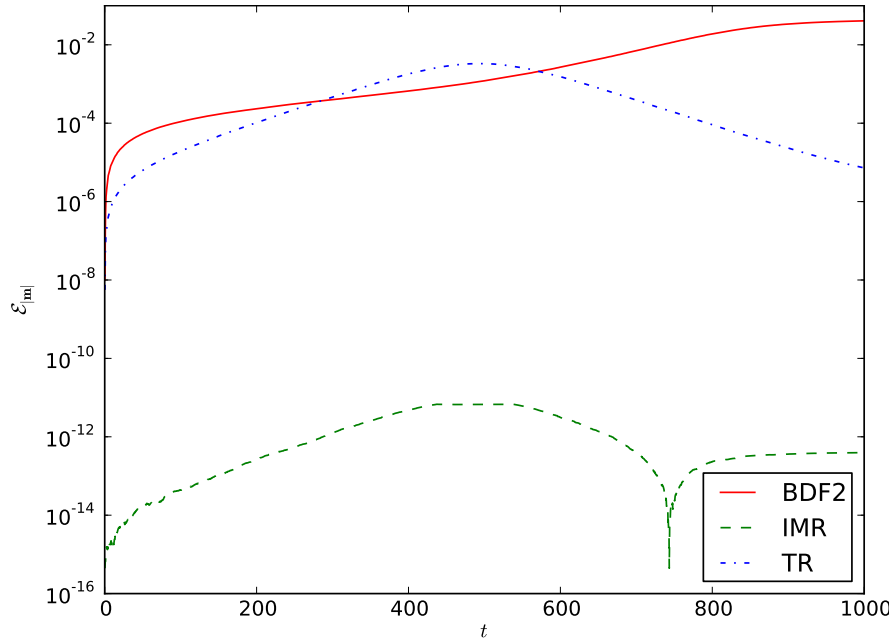

Figure 7.11: Plot of magnetisation length errors, $\mathcal{E}_{|\mathbf{m}|} = ||\mathbf{m}| - 1|$, over time for the relaxing nano-sphere problem solved with each of the three adaptive integrators (without re-normalisation of $\mathbf{m}$).

As mentioned in Section 4.5.2 we would expect to see some effect on the conservation properties when the accuracy of the linearisation method is varied. In our implementation the Newton-Raphson method is used for linearisation, so the relevant measure of accuracy is the Newton tolerance, ϵ_{N}. The obvious experiment to carry out would be to vary the Newton tolerance and examine how the error in $|\mathbf{m}|$ is affected. However the Newton-Raphson method converges extremely quickly meaning that the final residual is often many orders of magnitude smaller than the tolerance, this would hide any correlation between the tolerance and the error. Instead we plot the error against the *actual* converged residual norm (specifically: the mean over all time steps of $||\mathbf{r}||_\infty$ after the Newton method has converged for that time step). In order to generate a variety of converged residual norms we run the experiment with a range of parameters: $\alpha = 1, 0.1, 0.01$, $\epsilon_\Delta = 5\times10^{-3}, 10^{-3}, 5\times10^{-4}, 10^{-4}, 5\times10^{-5}, 10^{-5}$ and $\epsilon_{\mathrm{N}} = 10^{-12}, 10^{-10}, 10^{-8}, 10^{-6}$. A scatter plot showing the error in the length against the mean (over time) of the converged Newton residual norm is shown in Figure 7.12. We see that the magnetisation length conservation behaviour of IMR is well controlled by the Newton tolerance even with large variations in the adaptive time integrator tolerance and the damping parameter.

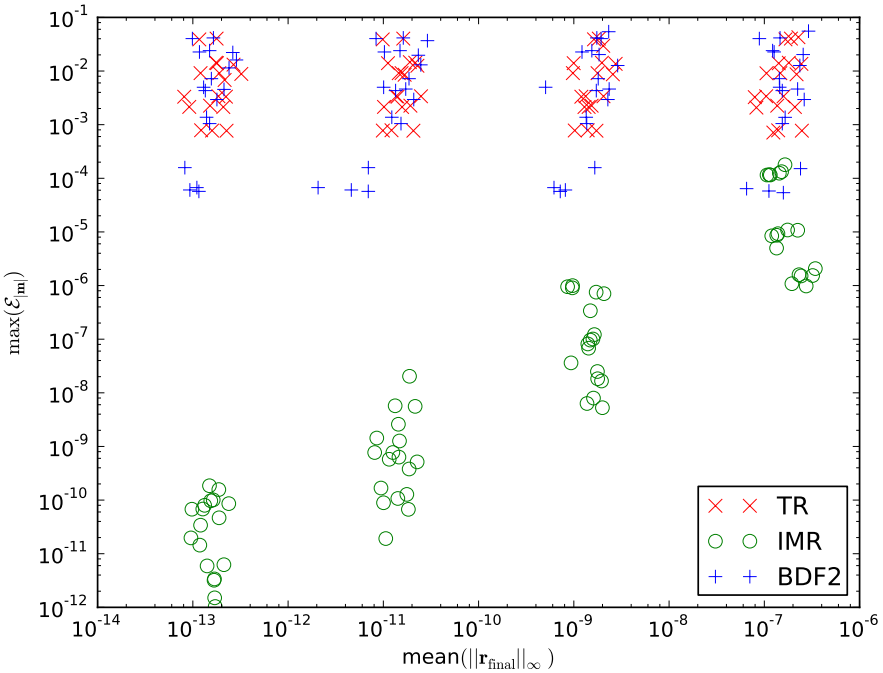

Figure 7.12: Plot of maximum magnetisation length error over time, $\max_n ||\mathbf{m}_n| - 1|$*, against the mean converged Newton residual norm for the relaxing nano-sphere problem with a wide range of* α*,* ϵ_{N} *and* ϵ_Δ *parameters.*

The behaviour of the maximum error in the energy when $\alpha = 0$ as the adaptive step tolerance is reduced is shown in Figure 7.13. The energy error for BDF2 without normalisation is around the Newton tolerance, but when renormalisation is used the error in energy increases drastically. This is due to the injection/removal of energy when the magnetisation length is modified. The energy error when using the TR/IMR algorithm (recall that in this example TR and IMR are identical and renormalisation is not needed for either method due to the geometric integration properties) is below the minimum value that can be calculated using floating point arithmetic. This is not surprising as we expect it to be significantly smaller than the energy error of BDF2, which is $\sim 10^{-14}$ and it is calculated as the difference of two values of O(1) (so it cannot be calculated if it is smaller than $\sim 10^{-16}$).

Finally, we show convergence plots with respect to the two error norms. Figures 7.14 and 7.15 show the convergence with respect to the switching time error, $\mathcal{E}_\tau$, and the global error, $\mathcal{E}_{\mathbf{m}}$, respectively. Adaptive TR and BDF2 algorithms both with and without re-normalisation of the magnetisation after each step are shown, along with the adaptive IMR algorithm developed in this chapter.

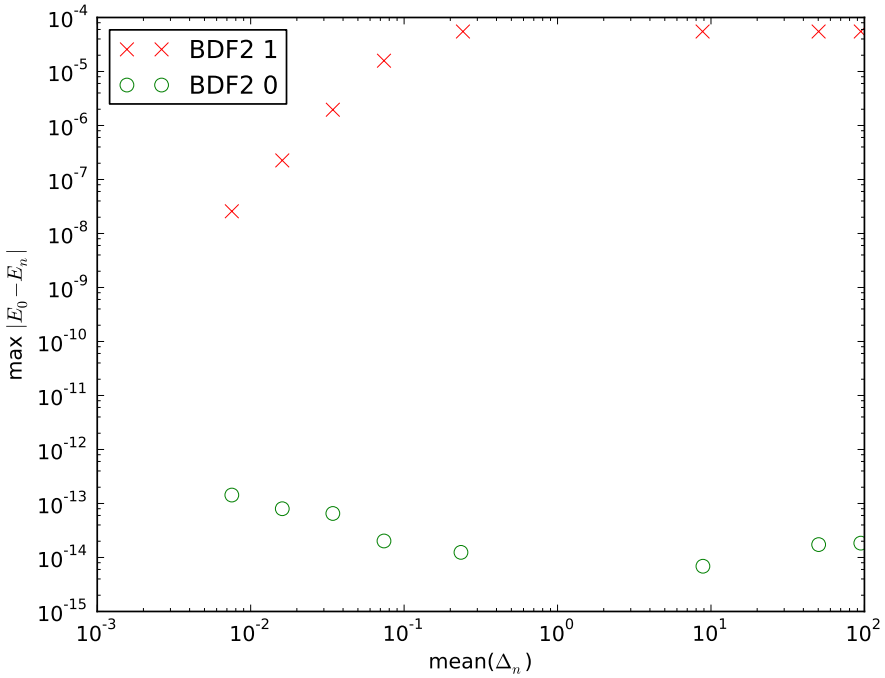

Figure 7.13: Convergence plot of the maximum error in the energy for the relaxing nano-sphere problem with $\alpha = 0$ solved using BDF2 with and without re-normalisation of $\mathbf{m}$. The energy error when using IMR is too small to be calculated. The digit 1 or 0 in the legend indicates re-normalisation or no re-normalisation respectively.

In Figure 7.14 the convergence (in $\mathcal{E}_\tau$) rates for TR and IMR are flat consistent with the asymptotic predictions, in contrast BDF2 does not attain this asymptotic convergence state until a comparatively small time step. The maximum errors are around 1000 time units, a relative error of $\sim 100\%$. The errors of TR and IMR are extremely similar for a given mean time step size. Once the asymptotic convergence regime is reached BDF2 has roughly three times the error of the other methods. Note that the re-normalisation of $\mathbf{m}$ in BDF2 and TR has only a minimal effect on their global error. This indicates that length conservation property of IMR is unlikely to be providing a major benefit to the global error in this simple example.

In Figure 7.15 all convergence (in $\mathcal{E}_\mathbf{m}$) rates show a kink around mean$(\Delta_n) = 0.1$. This is because $|\mathbf{m}| \approx 1$ is fixed so the error norm $\mathcal{E}_\mathbf{m}$ cannot be worse than the anti-parallel case.[8] Above the kink the azimuthal angle is inaccurate and the error

[8] A more effective error norm for such large error cases would be to use polar coordinates and track the total azimuthal angle passed through. However this is non-trivial to implement in general and some rescaling would be required to prevent the φ error from dominating.

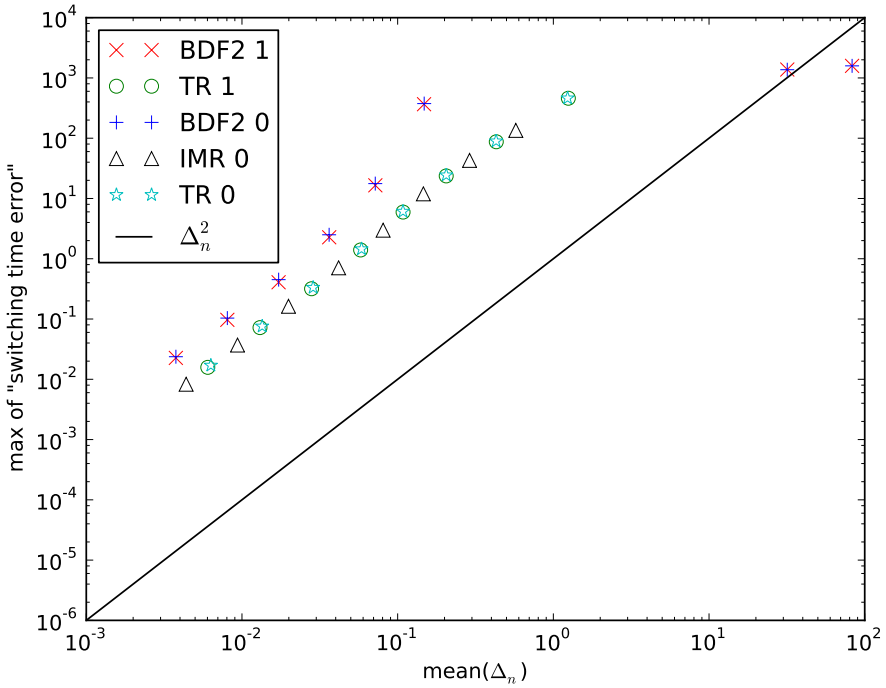

Figure 7.14: Convergence plot of the switching time error, (7.33), against average time step for each method. The digit 1 or 0 in the legend indicates re-normalisation or no re-normalisation respectively.

norms are meaningless, below the kink the usual convergence behaviour can be seen. Comparing the error norms different methods we see the same relationships as in Figure 7.14: BDF2 has three times the error and re-normalisation has little effect.

Note that TR consistently chooses larger time steps than IMR despite having similar global errors. This may indicate that the LTE of IMR is indeed larger than that of TR (as is expected), but that the geometric integration properties of IMR reduce the build up of global error. Alternatively it may be due to differing accuracies in the LTE estimate.

7.5 Conclusions

We have implemented an efficient algorithm for adaptive time step selection with the implicit midpoint rule. The algorithm has been tested on a number of ODEs and found to follow the correct theoretically predicted step selection behaviour, and to behave similarly to other second-order adaptive time integrators. We have

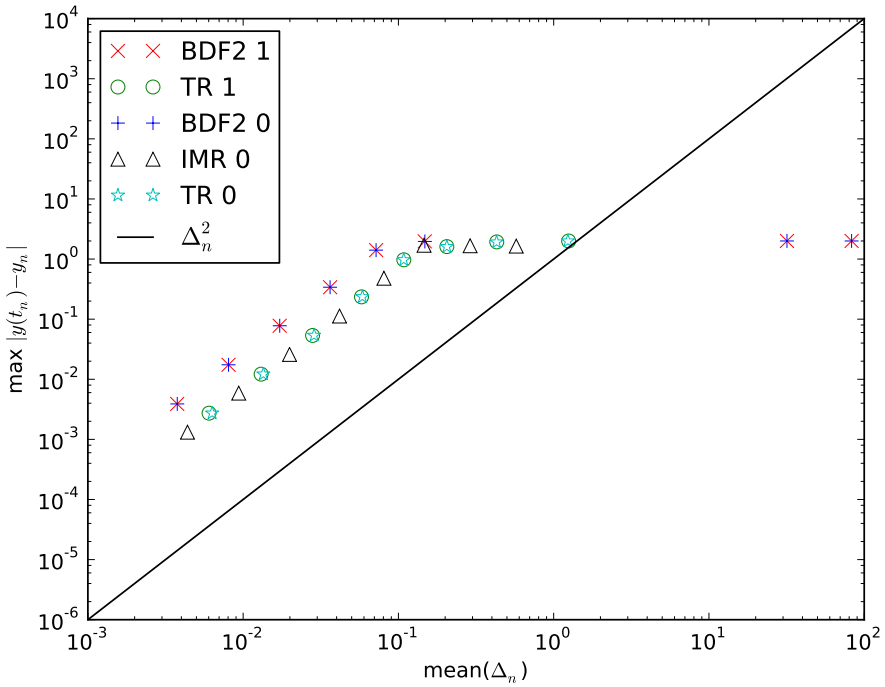

Figure 7.15: Convergence plots of the error in magnetisation, (7.37), against average time step for each method. The digit 1 or 0 indicates re-normalisation or no re-normalisation respectively.

also demonstrated robustness in stiff problems and correct behaviour even when order reduction phenomena are encountered.

When applied to an ODE form of the LLG the adaptive IMR algorithm conserves $|\mathbf{m}|$ as expected for a wide variety of parameters, but the example used here may be too simple for the energy property to be studied properly[9]. It also appears to give smaller global error norms than would be expected from its local truncation error, this may be due to the geometric integration properties. Any effects of these geometric integration properties would be expected to be greatly amplified when solving ODE problems with many interacting spins or PDE problems. This scenario will be tested in Chapter 8 where the adaptive IMR algorithm is combined with the PDE methods discussed in other chapters.

Finally, we have observed some interesting results on the suitability of certain time integration methods to the simulation of the LLG equation. We have found that the use of BDF2 results in significantly larger errors in the calculated switching

[9]Without damping the ODE form of the LLG is linear in $\mathbf{m}$ and t so TR and IMR are equivalent.

time than when the TR or IMR methods are used, particularly when the precessional behaviour is not fully resolved. Assuming that a given level of error is required for an application this means that IMR and TR can take larger time steps (by a factor of $\sqrt{E_{\mathrm{BDF2}}/E_{\mathrm{IMR}}}$), resulting a proportional reduction in the computation time. We have also found that the process of re-normalising the magnetisation length after each time step in BDF2 methods greatly increases the error in the energy.

Chapter 8

Validation, convergence and conservation experiments

In this chapter we apply the various numerical methods constructed throughout this thesis to some micromagnetics problems.

The first case examined is a 2D problem without magnetostatics where there is a known wave-like analytical solution for certain initial conditions. This allows us to test the convergence of the various discretisation approaches when applied to the LLG with the exchange effective field (*i.e.* the PDE form). It also allows us to test the geometric integration properties of the IMR for a simple PDE problem.

The second problem is another simple case, this time with no analytical solution: a non-uniform applied field is used to induce non-uniform dynamics in a 2D problem without magnetostatics. This allows us to test geometric integration properties of the IMR in the absence of an analytical solution.

Finally we show results for the µmag standard problem #4 [73], which is widely used to test dynamic micromagnetic codes. This allows us to verify the complete model and to examine the geometric integration properties when FEM/BEM magnetostatics calculations are included.

8.1 Example with a wave-like solution

8.1.1 Problem definition

In this experiment we solve a micromagnetic problem which has a wave-like exact solution in 2D. The details of this solution are given in Appendix A.2. We solve the LLG without magnetostatics on a two dimensional square domain $\Omega = [0,1] \times [0,1]$ with periodic boundary conditions. We integrate time over $T = [0,5]$. The solution is obtained by setting the initial condition according to (A.4):

$$\begin{aligned} m_x(0) &= \sin(c)\cos(\mathbf{k}\cdot\mathbf{x}), \\ m_y(0) &= \sin(c)\sin(\mathbf{k}\cdot\mathbf{x}), \\ m_z(0) &= \cos(c), \end{aligned} \tag{8.1}$$

and using $\mathbf{h}_{\text{ap}} = \mathbf{0}$. The solution parameters used are $\mathbf{k} = [2\pi, 2\pi]$ (so that the solution is periodic on domains of unit size), $c = 0.1\pi$, and $\alpha = 0.01$. For the energy conservation experiments $\alpha = 0$ is used instead.

This example problem allows us to examine the convergence and geometric integration properties of IMR with the FEM using nodal quadrature. In particular the existence of an exact solution allows us to easily measure the convergence rate for the various methods.

8.1.2 Implementation details

We use the finite element method as discussed in Chapter 4 to spatially discretise the LLG equation. Unless otherwise specified we use a mesh of square elements with 5×2^4 elements along each edge (1681 nodes in total). For the evaluation of the resulting integrals both the nodal quadrature discussed in Section 4.5.1 and standard Gaussian quadrature methods are used.

For time integration the adaptive IMR, TR and BDF2 methods are used with a tolerance of $\epsilon_\Delta = 10^{-5}$ and an initial time step of $\Delta_0 = 10^{-5}$ (significantly smaller than the time step selected by an initial experimental run). Those methods which do not naturally conserve $|\mathbf{m}|$ (the TR and BDF2 methods, and IMR with Gaussian quadrature) are run both with and without re-normalisation of the magnetisation. The re-normalisation process is implemented by setting each

nodal value of the magnetisation to $\mathbf{m}_i = \mathbf{m}_i / |\mathbf{m}_i|$ after each time step (after the selection of the next step size to avoid complicating the adaptivity process).

Linearisation is handled using the Newton-Raphson method with Newton tolerance set to $\epsilon_{\mathrm{N}} = 10^{-12}$ (unless otherwise specified). The resulting linear systems are solved using GMRES with an ILU(1) preconditioner as described in Section 6.1 and Section 6.3.1.

The adaptive IMR integrator, as described in Chapter 7, requires the computation of an explicit time step. For this step we use the Landau-Lifshitz form of the LLG discretised in exactly the same way as described in Chapter 4. The resulting linear system (involving only a mass matrix) is solved using the method of conjugate gradients preconditioned by the lumped mass matrix (*i.e.* the diagonal matrix of the row sums). Note that when nodal quadrature is used the mass matrix is a diagonal matrix and no linear solve is needed (see Section 4.5).

8.1.3 General results

An example snapshot of the solution at time $t = 0.1$ is shown in Figure 8.1. Over time the wave moves in the $[-1, -1]$ direction and is simultaneously damped out towards the $\mathbf{m} = [0, 0, 1]$ state.

The behaviour of the approximate solutions over time at $\mathbf{x} = \mathbf{0}$ is shown in Figure 8.2. Only results from the methods without re-normalisation are shown because they are identical to the equivalent method with re-normalisation (where applicable) at the scale of the plot.

The time steps selected by the various algorithms for this problem are also shown in Figure 8.2. The appearance of a jump in the time step size at $t = 0$ is due to the algorithms rapidly increasing the time step size from the small initial value to a step size appropriate for the given tolerance. After an appropriate step size is reached there is a smooth, gradual increase as the solution is damped out.

8.1.4 Convergence study

Since we have an exact solution for this example we can easily calculate an error norm and plot the convergence as both the time step Δ_n and the element edge length h go to zero. Following the example of Jeong *et al.* [56] we link the

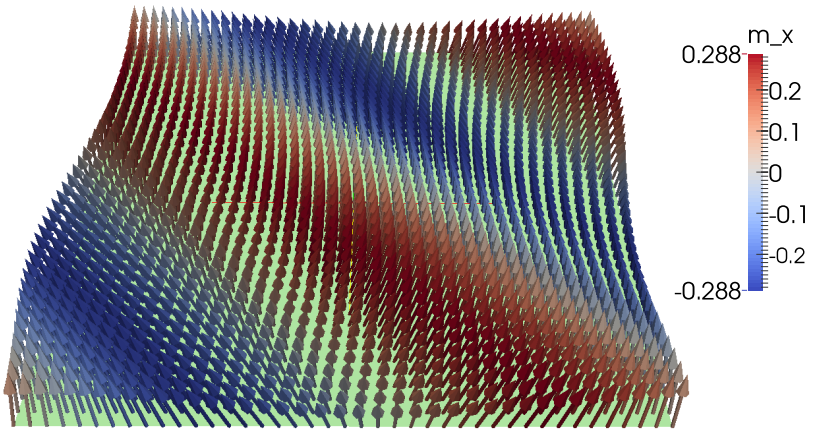

Figure 8.1: Snapshot of the approximate solution for the 2D wave problem at $t = 0.1$ time units obtained using adaptive IMR with nodal integration. Arrows indicate the nodal values of $\mathbf{m}$, colour indicates the value of m_x, and the pale green area is the finite element mesh used. The z-component of the magnetisation is constant over the domain with $m_z = 0.958$.

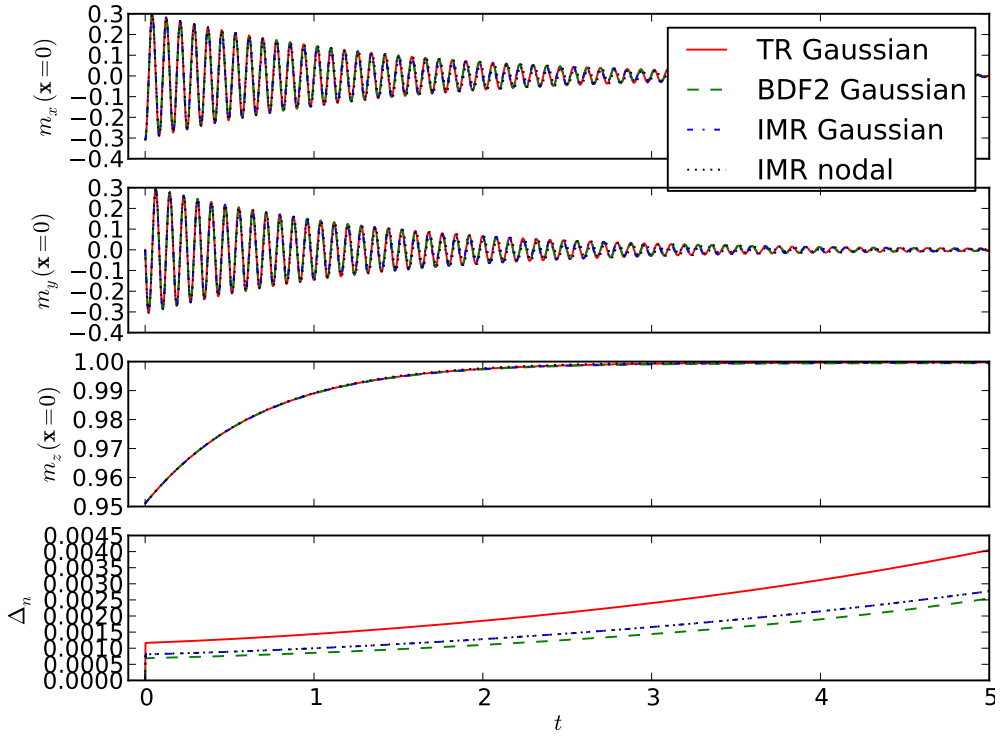

Figure 8.2: The temporal behaviour of the wave solution at $\mathbf{x} = \mathbf{0}$ and the time step selected by the various adaptive integration schemes without renormalisation.

time step size to the spatial discretisation size by $\Delta_n = 0.32h$. It is important to note that, in contrast to explicit time integration schemes, this coupling of the time and space discretisation parameters is *not* required for stability. It is merely more convenient to experiment with a single parameter than with two independent parameters. We choose element edge lengths of $h = 1/(5 \cdot 2^n)$ with $n = 1, 2, 3, 4, 5, 6$.

In these experiments we always use re-normalisation for schemes which are not expected to naturally conserve $|\mathbf{m}|$ because in practical applications such schemes would always use re-normalisation.

As a first convergence experiment we examine the norm of the error after a single time step

$$\mathcal{E}_{\mathbf{m},1} = ||\mathbf{m}(\mathbf{x}_j, t_1) - \mathbf{m}_{j,1}||_2 \,. \tag{8.2}$$

In this case we expect the convergence (in both space and time) to be second order for all methods. The results of this experiment are shown in Figure 8.3, We see that convergence is indeed second order for all methods and that the accuracy of the BDF2 scheme is worse than the other schemes by a roughly constant factor. We also see that the use of nodal quadrature results in an error increase by a small constant factor. This indicates that the geometric integration properties of IMR with nodal quadrature give little or no benefit for a single time step, as would be expected.

We also wish to examine a norm of the error after a large number of time steps, so for the rest of the experiments in this section we examine the integral of the error norm over $T = [0, 5]$. However, reaching the levels of convergence required for $\mathcal{E}_{\mathbf{m}}$ to be meaningful is difficult because the "worst case" for the norm is that the approximated magnetisation is out of phase with the exact solution (*i.e.* the error norm is bounded). A plot of the time integral of $\mathcal{E}_{\mathbf{m}}$ is shown in Figure 8.4. No convergence is seen for IMR and TR until the spatial/temporal refinement level reaches $n = 5$, and no convergence at all is seen for BDF2.

To obtain more meaningful results we should use an alternative error norm which allows a better measure of the error even for approximate solutions which are out of phase with the exact solution. One such error norm measures the deviation of m_z from the exact value

$$\mathcal{E}_{m_z} = \max_j |m_{z,j,k} - m_z(t_n)| \,. \tag{8.3}$$

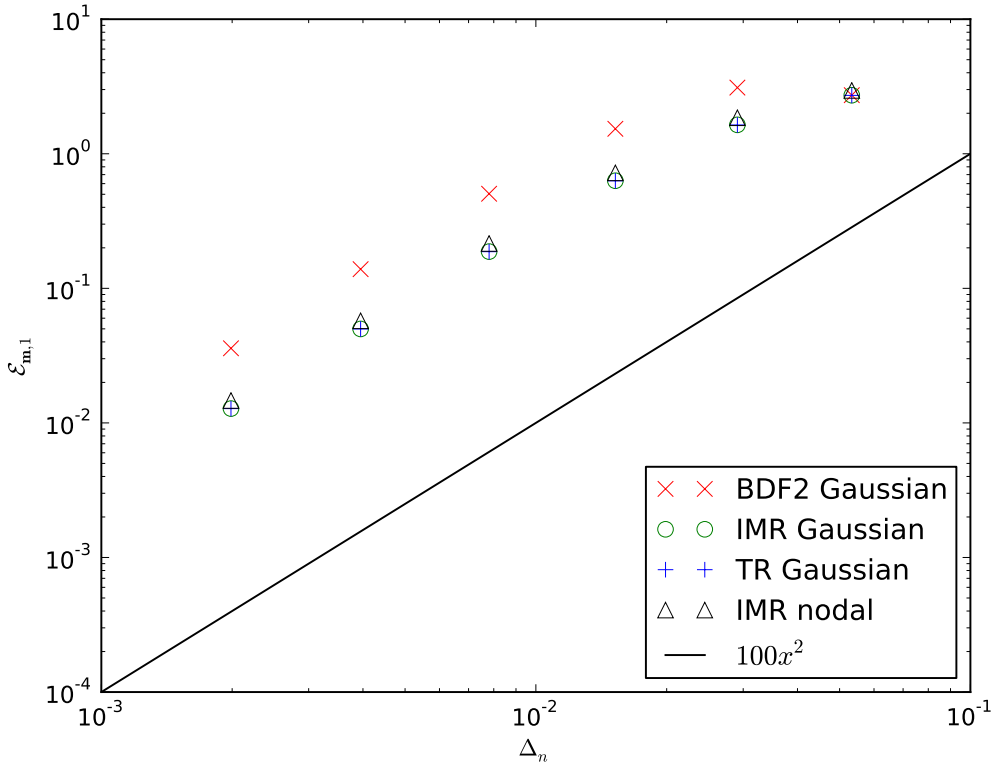

Figure 8.3: Convergence in the error norm $\mathcal{E}_{\mathbf{m}}$ after a single step of time integration for the 2D wave-like problem.

Note that the exact value of m_z is not a function of $\mathbf{x}$ and so it is sufficient to take the maximum value. This error norm measures the level of over/under damping caused by the approximation.

The convergence results for the time integral of the error norm $\mathcal{E}_{m_z}$ are shown in Figure 8.5. We again see second order convergence behaviour for all methods. The BDF2 method requires additional refinement to reach the asymptotic convergence behaviour, and has around an order of magnitude larger error than the other methods once convergence is reached. This difference in error means that the IMR and TR methods are able to take time steps roughly $\sqrt{10} \sim 3$ times larger while still attaining the same global error, resulting in a factor of three speed-up (assuming that the error norm considered here is relevant to the application). Also note that there is no significant difference between IMR with nodal quadrature (which conserves $|\mathbf{m}|$) and IMR with Gaussian quadrature (in which $\mathbf{m}$ is renormalised), indicating that geometric integration offers no significant benefits in this case.

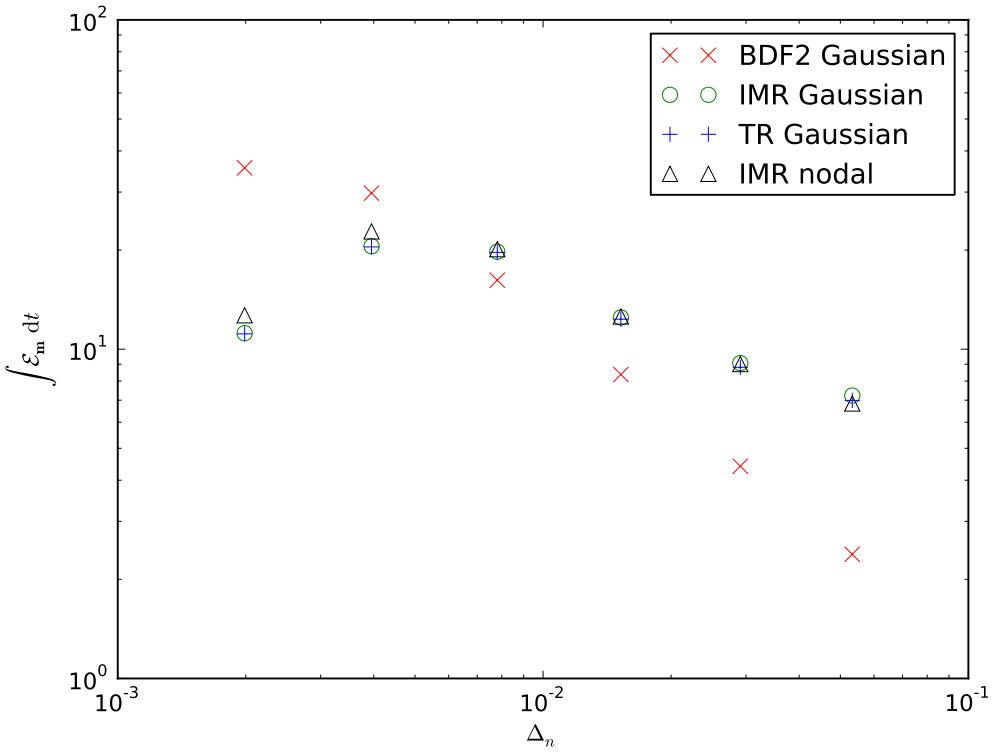

Figure 8.4: Convergence of $\int_T \mathcal{E}_{\mathbf{m}} \, \mathrm{d}t$*, where* $T = [0, 5]$*, for the 2D wave-like problem.*

8.1.5 Geometric integration properties

We now examine the error in nodal magnetisation length:

$$\mathcal{E}_{|\mathbf{m}|}(t_n) = \max_j \left| |\mathbf{m}_{j,n}| - 1 \right|, \tag{8.4}$$

in the approximations generated by each of the time integration schemes (without re-normalisation). Figure 8.6 shows the evolution of $\mathcal{E}_{|\mathbf{m}|}$. When using IMR with nodal quadrature the magnetisation length error remains extremely small, but for other methods the error rapidly grows to around $\mathrm{O}(10^{-4})$ and seems to saturate at this level.

We next examine the energy conservation properties of the various schemes for the wave solution with $\alpha = 0$. The only energy involved is the exchange energy, which can be calculated using (2.48). Note that these integrals can be evaluated exactly by either quadrature scheme because $\nabla \mathbf{m}$ is a constant inside each element. The results are shown in Figure 8.7. Interestingly we see that IMR with Gaussian quadrature and TR both conserve energy in a similar manner to IMR with nodal

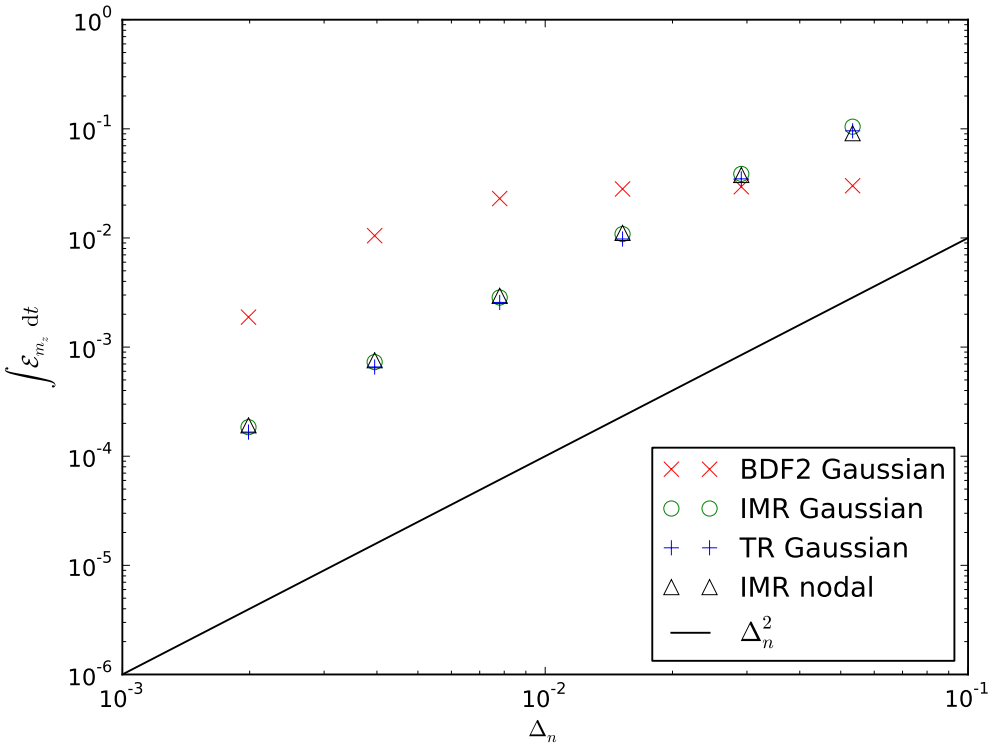

Figure 8.5: Convergence of $\int_T \mathcal{E}_{m_z}\,\mathrm{d}t$, where $T = [0, 5]$, for the 2D wave-like problem.

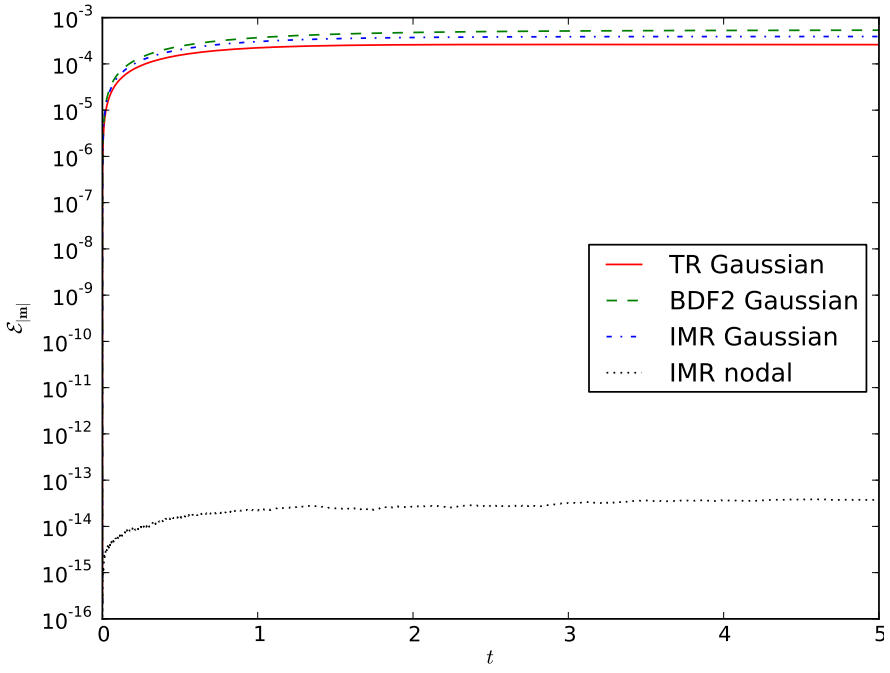

Figure 8.6: Evolution of $\mathcal{E}_{|\mathbf{m}|}$ with various time integrators and quadratures for the 2D wave-like problem.

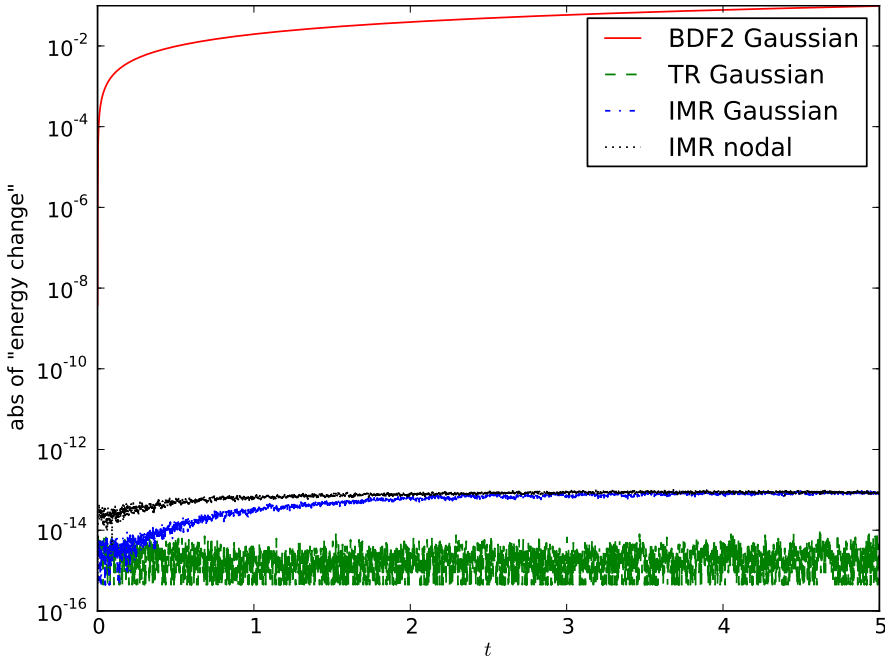

Figure 8.7: Evolution of the error in energy with various time integration schemes and quadratures for the undamped 2D wave-like problem.

quadrature. This is likely due to some special property of the exact solution, and does not appear to be the case in general (see Section 8.2).

8.1.6 Effect of Newton tolerance

In Section 4.5 relationships between the conservation properties and the accuracy of the linearisation method were derived. In our implementation the Newton-Raphson method is used for linearisation, so the relevant measure of accuracy is the Newton tolerance, ϵ_{N}. As was described in Section 7.4.3 we study this effect by plotting the error against the *actual* converged residual norm. In order to generate a variety of converged residual norms we run the experiment with a range of parameters: $\epsilon_{\mathrm{N}} = 10^{-8}, 10^{-9}, 10^{-10}, 10^{-11}, 10^{-12}$; $\epsilon_{\Delta} = 10^{-3}, 10^{-4}, 10^{-5}$; $\alpha = 1, 0.001, 0$; and $h = 0.05, 0.025, 0.0125$. We only integrate in time over the shorter interval $T = [0, 1]$ due to the volume of computations required. The results are shown in Figure 8.8, there is a clear correlation between the magnitudes of the final residuals and the error in $|\mathbf{m}|$. A similar result is seen in Figure 8.9 for the energy conservation property when $\alpha = 0$.

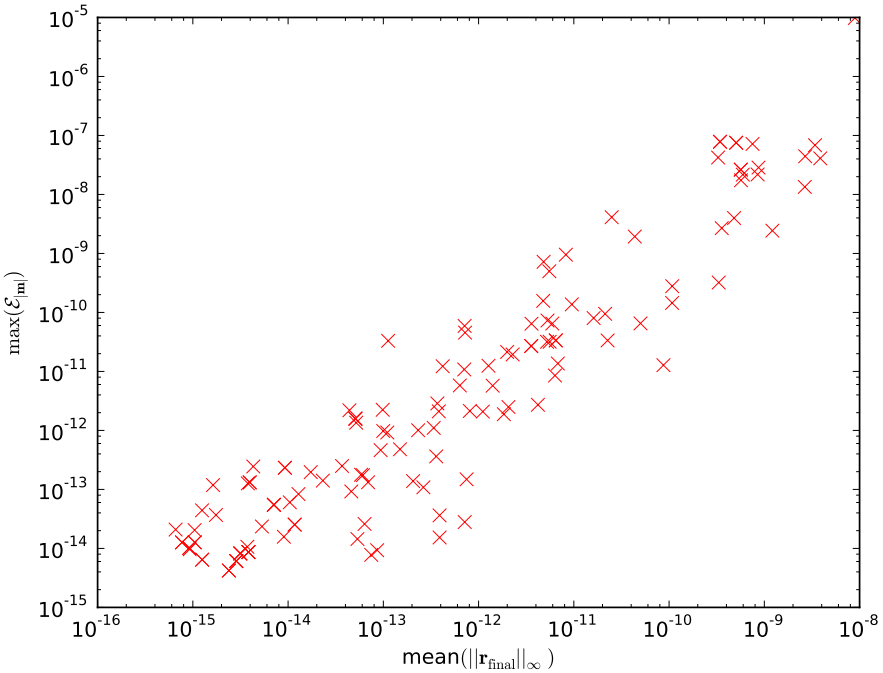

Figure 8.8: Correlation between maximum (over all nodes and all time steps) error of nodal magnetisation lengths and the mean (over time) of the converged Newton residual norm in the 2D wave-like problem solved using adaptive IMR and nodal quadrature.

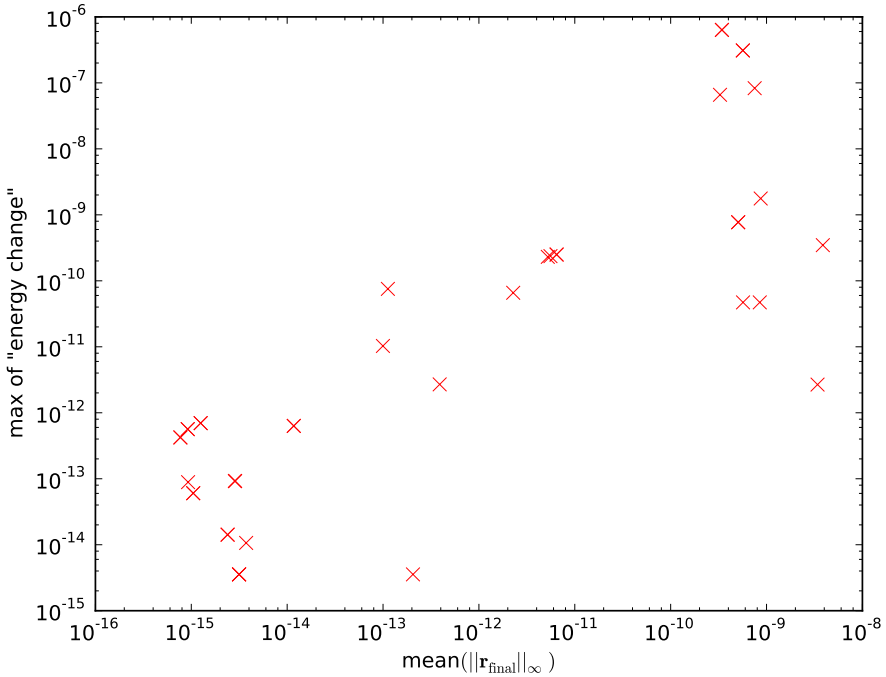

Figure 8.9: Correlation between the maximum (over time) of the error in energy and the mean (over time) of the converged Newton residual norm in the undamped 2D wave-like problem solved using adaptive IMR and nodal quadrature.

8.1.7 Conclusions

We have shown that the methods we tested have the expected convergence and time step selection properties. We also observed that the accuracy of the BDF2 method is lower than the TR or IMR methods by around one order of magnitude. Due to this effect the TR and IMR methods can take larger time steps while attaining the same error, resulting in shorter computation times.

We have also shown that the geometric integration properties of IMR are preserved in a weak form FEM model with a nodal quadrature scheme. As predicted in Section 4.5 these properties are linked to the accuracy of the non-linear solver.

Additionally we observed unexpected geometric integration properties when IMR with Gaussian quadrature or TR are used. It is expected that this is a consequence of the specific test problem rather than the methods themselves, and this conjecture will be tested in the next section.

Since all methods except for BDF2 showed some geometric integration properties the effect of such properties on the overall error cannot be analysed using this example.

8.2 An example with non-uniform applied field

In this section we construct a problem which is non-uniform in space without the involvement of the magnetostatic field by applying a non-uniform applied field. This will allow us to examine whether the anomalous geometric integration properties observed in the previous section are related to the analytical solution.

This experiment also reveals some reduction in the effectiveness of the geometric integration on meshes of triangular elements.

8.2.1 Problem definition

We solve the LLG without magnetostatics on a two dimensional square domain $\Omega = [0, 5] \times [0, 5]$ with Neumann boundary conditions. The problem is integrated in time over $T = [0, 5]$.

We use a uniform initial condition $\mathbf{m} = [1, 1, 0]/\sqrt{2}$ with a non-uniform applied field $\mathbf{h}_{\text{ap}} = [0, 0, h_z]$ where

$$h_z = \begin{cases} 1.1(1 - x) & x < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{8.5}$$

As before we use $\alpha = 0.01$ except for energy conservation experiments where we use $\alpha = 0$.

The dynamics occurring in this problem are quite simple: the magnetisation in the region of the domain $x < 1$ moves towards $\mathbf{m} = [0, 0, 1]$ due to the applied field. Exchange interactions force the rest of the domain to move towards the same field, albeit much more slowly, until eventually the entire domain is in a uniform state.

The implementation details of this example are exactly as described in Section 8.1.2, except that we also use meshes of triangular elements. The structure of the triangular element mesh is equivalent to the square element mesh except that the square elements are divided diagonally into two triangles.

8.2.2 Geometric Integration properties

The evolution of the magnetisation length for this experiment is shown in Figure 8.10, we see that in this case only IMR with nodal quadrature conserves $|\mathbf{m}|$ as expected. The same is true for the energy conservation with zero damping, as shown in Figure 8.11.

8.2.3 Triangular meshes

Finally we show some results when a mesh of triangular elements is used. The evolution of $|\mathbf{m}|$ is shown in Figure 8.12. We see that the error in $|\mathbf{m}|$ when using IMR with nodal quadrature is significantly larger on triangular elements than on square elements. Similarly the error in the energy (with $\alpha = 0$) is significantly larger, as shown in Figure 8.13. Other experiments show this issue for a variety of other cases including 3D problems with tetrahedral elements, but not for the wave-like example above.

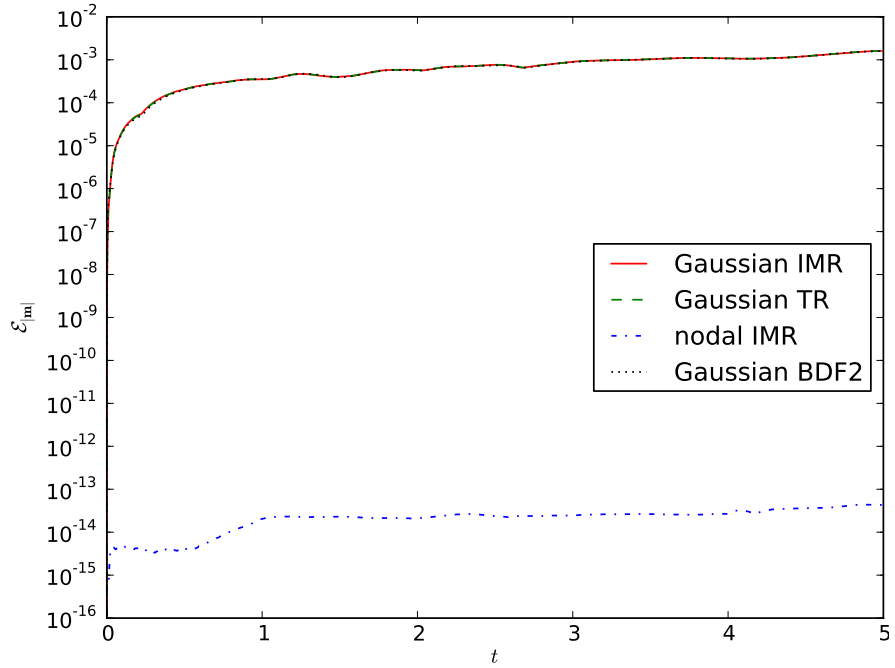

Figure 8.10: Evolution of $\mathcal{E}_{|\mathbf{m}|}$ with various time integration schemes and quadratures for the 2D nonuniform field problem.

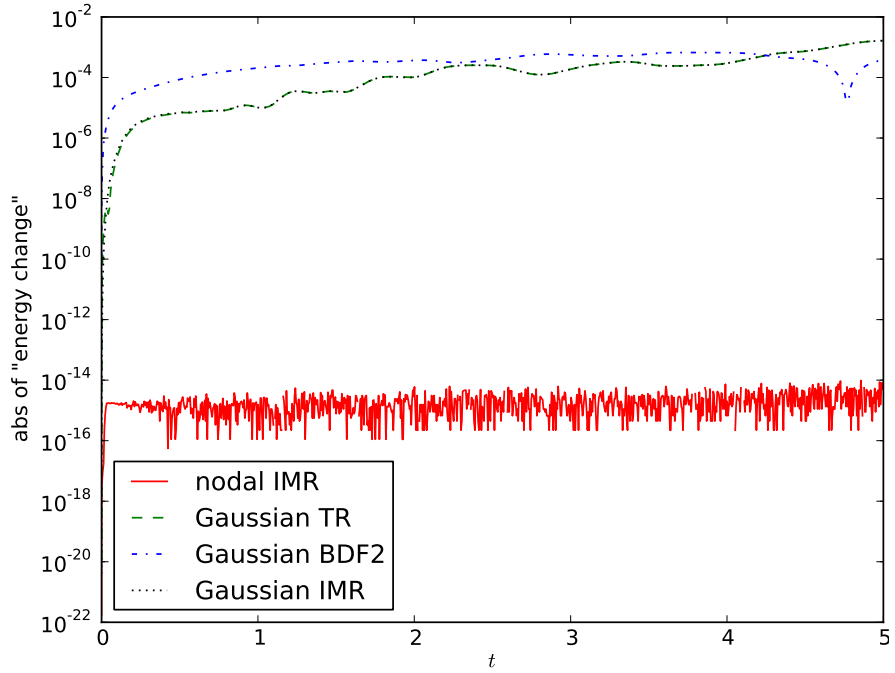

Figure 8.11: Evolution of the error in energy with various time integration schemes and quadratures for the undamped 2D nonuniform field problem.

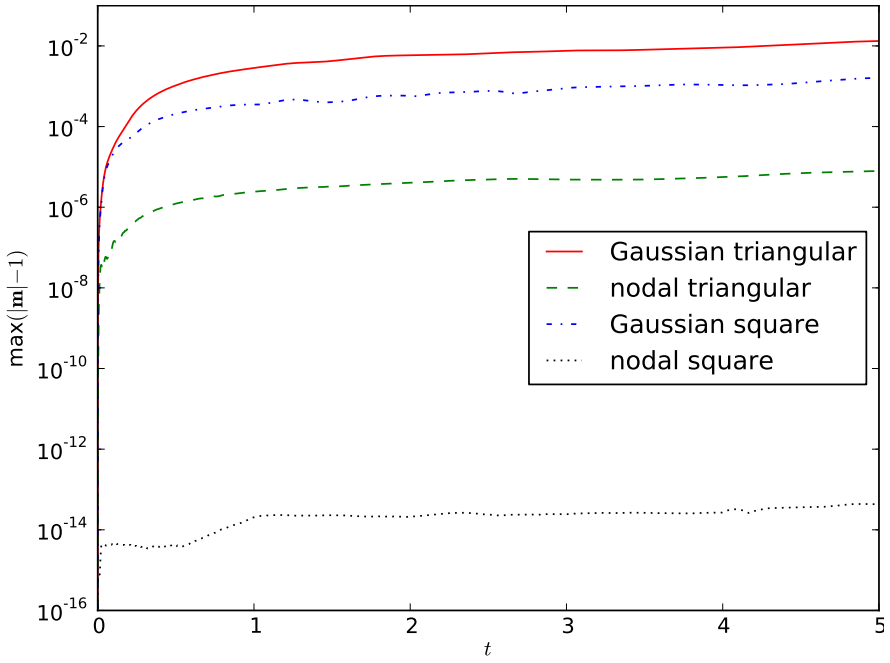

Figure 8.12: Evolution of $\mathcal{E}_{|\mathbf{m}|}$ with IMR, both quadratures, and both element shapes for the 2D nonuniform field problem.

We do not have an explanation for this effect. However it should be noted that the non-conservation effects are small enough that they would not be noticed if the numerical experiments were run with a loose non-linear solver tolerance, such as used in *e.g.* [10].

8.2.4 Conclusions

We have confirmed that the wave-like example is in some sense "too easy" and does not provide a rigorous test of geometric integration properties: IMR with Gaussian quadrature and TR do not conserve energy or $|\mathbf{m}|$ in general.

We have also shown an example problem where our implementation of IMR with nodal quadrature unexpectedly gives significantly worse conservation of energy and $|\mathbf{m}|$ when used with a mesh of triangular elements, as compared to square elements.

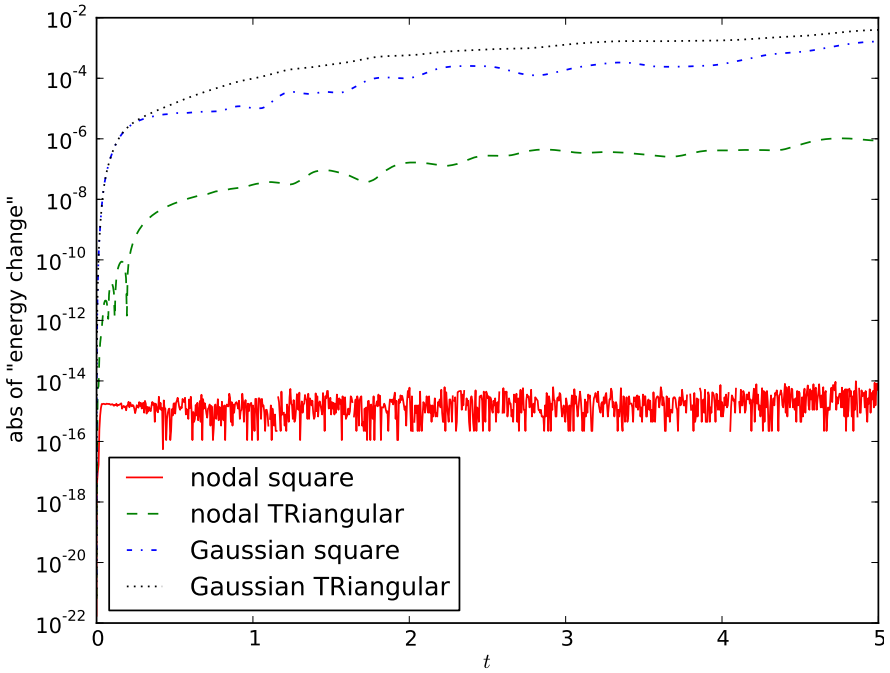

Figure 8.13: Evolution of the error in energy with IMR, both quadratures, and both element shapes for the undamped 2D nonuniform field problem.

8.3 The µmag standard problem #4

The µmag standard problem #4 [73] is the benchmark problem most widely used to test implementations of dynamic micromagnetic models. It involves modelling the reversal of an extremely thin cuboid film of permalloy-like material under two different applied fields. Unfortunately FEM/BEM magnetostatic calcultations are ill-suited to thin film problems: the dense BEM matrix size is proportional to the number of nodes on the boundary and in a thin film every single node is on the boundary. Additionally the key benefit of FEM/BEM (accurate resolution of complex geometries) is not required since the film is a simple cuboid. However, since there are no other widely studied test problems and there are no problems with non-trivial magnetostatic and exchange effects for which an analytical solution is known, we use the standard problem to verify our implementation.

8.3.1 Problem specification

The magnetic domain is a sheet of magnetic material $500 \times 125 \times 3$nm with material parameters

$$\begin{aligned} A &= 1.3 \times 10^{-11}\text{J/m}, \\ M_s &= 8.0 \times 10^{5}\text{A/m}, \\ K_1 &= 0.0, \\ |\gamma_{\text{L}}| &= 2.211 \times 10^{5}\text{m/As}, \\ \alpha &= 0.02. \end{aligned} \tag{8.6}$$

The simulation is run with two different applied fields:

$$\begin{aligned} \mu_0 \mathbf{h}_{\text{ap},1} &= [-24.6, 4.3, 0.0]\ \text{mT}, \\ \mu_0 \mathbf{h}_{\text{ap},2} &= [-35.5, -6.3, 0.0]\ \text{mT}, \end{aligned} \tag{8.7}$$

where $\mu_0 = 4\pi \times 10^{-7}$. The initial condition is the S-state given by slowly relaxing the magnetisation from the state created by a saturating field in the $[1, 1, 1]$ direction.

These magnetic parameters result in a magnetostatic exchange length (and unit length in our simulations) of

$$l_{\text{ex}} = \sqrt{\frac{2A}{\mu_0 M_s^2}} = 5.6858\text{nm}, \tag{8.8}$$

hence the normalised dimensions are approximately $87.94 \times 21.98 \times 0.53$. Our unit time is

$$t_{\text{unit}} = \frac{1}{|\gamma_{\text{L}}| M_s} = 5.653\text{ps}. \tag{8.9}$$

8.3.2 Implementation details

We use the FEM to spatially discretise the LLG equation and the Newton-Raphson method to solve the resulting non-linear systems as described in Chapter 4. The hybrid FEM/BEM method, described in Chapter 5, is used for magnetostatic calculations. For the coupling of the LLG equation with the magnetostatic calculations we use both the monolithic and semi-implicit methods discussed in Chapter 6. The solution of the monolithic linear system is done using the fully

iterative preconditioner $\widetilde{\mathcal{P}_3}$, it turns out that for thin film problems the ILU(1) approximation to the LLG block is effective. The decoupled systems resulting from the semi-implicit method are solved using the methods described in Section 6.1. Both methods use the parameters described in Section 6.3.1.

The TR, BDF2 and IMR adaptive time integration schemes (see Section 7.2 and Section 7.3.1) are used with both the Gaussian and nodal quadrature schemes. For those schemes which do not naturally conserve $|\mathbf{m}|$ both normalised and renormalised versions are tested.

As previously mentioned, the adaptive IMR requires an explicit time step for the estimation of the error. This is implemented as described in Section 8.1.2. No coupling with the FEM/BEM calculations is required for this explicit step because only the value of the magnetostatic potential at the n-th time step is required (which is already known from the IMR calculations).

A structured mesh consisting of cuboid elements is used due to the simple geometry of the problem and the fact that we have observed issues with geometric integration on tetrahedral elements (see Section 8.2.3). In the z direction (out of the thin film plane) a single layer of elements is used at all refinements. This is the standard approach [73], and is expected to give acceptable resolution because the exchange length of the material (the length scale over which the magnetisation can vary strongly) is around twice the thickness of the film. The number of elements along the x and y axes, denoted n_x and n_y, is varied but the ratio is fixed at $n_x = (500/125)n_y$ so that the element edge lengths in each direction are identical. We use $n_x = 75, 100, 125$, which gives edge lengths of 1.17, 0.89 and 0.70 exchange lengths respectively.

The Newton tolerance is set to $\epsilon_{\mathrm{N}} = 10^{-11}$, the adaptive integrator tolerance is $\epsilon_{\Delta} = 10^{-5}$ and the initial time step is $\Delta_0 = 10^{-4}$. We cap the time step at $\Delta_{\max} = 4.5$ to avoid issues with linear and solver convergence with extremely large time steps.

To generate the initial S-state we run the simulation for 300 time units starting from the state $\mathbf{m} = [1, 1, 1]$ with $\alpha = 1.0$ and the applied field

$$\mathbf{h}_{\mathrm{api}}(t) = \begin{cases} 10(1 - \frac{t}{100}) & t < 100, \\ 0 & t \geq 100. \end{cases} \tag{8.10}$$

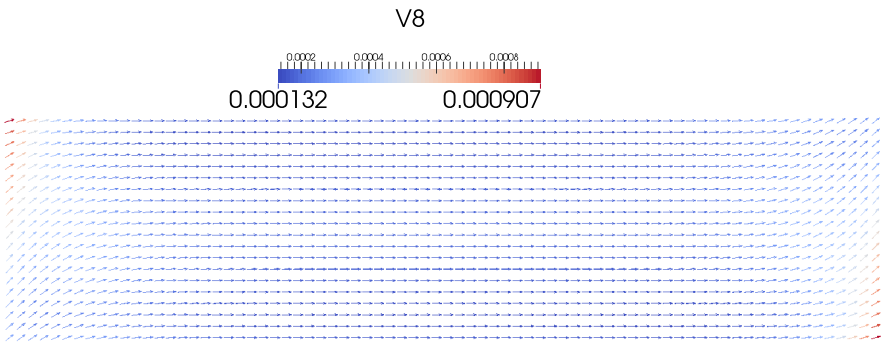

Figure 8.14: Initial S-state as generated by adaptive IMR with nodal quadrature and $n_x = 75$ nodes along the x direction. Colour indicates the z (out of plane) component of magnetisation.

The time integrator history data is then set such that it has been in this state forever and the time step sizes are set to the initial value. The relevant applied field as specified in the problem is set and the simulation is continued. The initial condition is always generated using the same numerical methods as are used in the dynamic part of the simulation.

Note that with $\alpha = 1$ the time scale for the magnetisation to relax is much shorter than with $\alpha \sim 0.01$. Hence the simulated time allowed for relaxation to the initial condition is sufficiently long to be considered "slow" despite the fact that it is much shorter than the simulated time for the dynamics calculations.

We compare our results against those submitted to the µmag website by d'Aquino *et al.* [73]. These results were generated by a fixed time step IMR scheme with a finite difference spatial discretisation monolithically coupled to a fast Fourier transform method for magnetostatic calculations [24].

8.3.3 General results

For this problem we find that without either re-normalisation or geometric integration the time steps selected by the adaptive algorithms become unacceptably small (*i.e.* the error becomes large). Hence these cases were not run until completion and are not included in the results.

In Figure 8.14 we show the initial S-state, as generated by IMR with nodal quadrature.

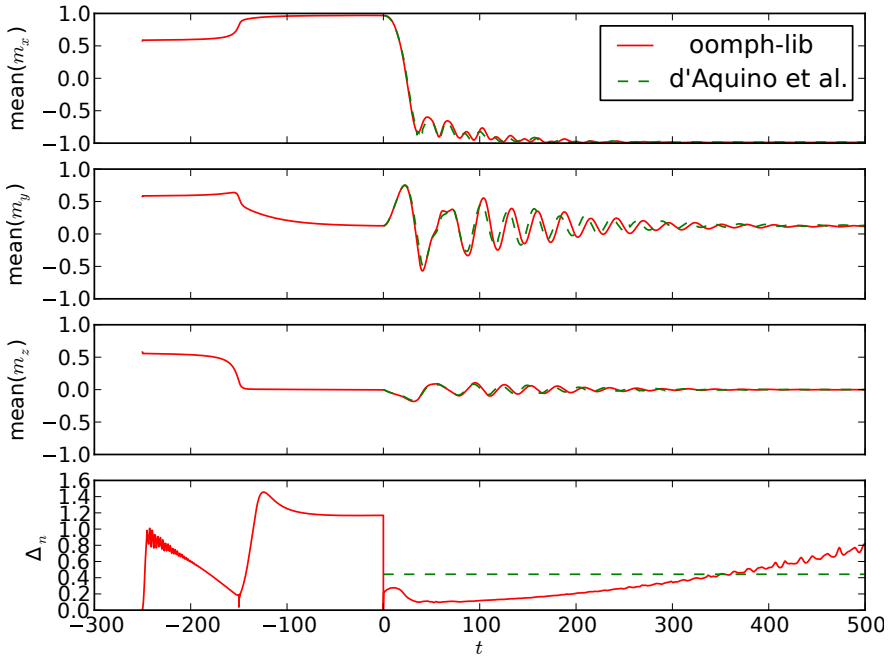

Figure 8.15: Evolution of the mean magnetisation for the μmag problem #4 with field 1 using monolithic IMR with nodal quadrature. For comparison we also show the results submitted to the μmag website by d'Aquino et al. [73].

Next we show the mean magnetisation for the two fields from (8.7) in Figures 8.15 and 8.16 respectively. Our results with smaller numbers of nodes, other time integration schemes, and smaller adaptive time integrator tolerance agree well with the results shown in Figures 8.15 and 8.16.

The results agree reasonably well with the results obtained by d'Aquino except around $t = 100$ when field 2 is used (Figure 8.16). We have also tested the behaviour when using IMR with fixed time steps or alternative methods for generating the initial S-state but these have no effect on the discrepancy. However, it is worth noting that all of the submitted solutions on the μmag website [73] behave differently for this part of the problem! The issues may be due to corner singularities in the magnetostatic field and, if so, could be resolved by the use of higher order solution and test basis functions for the magnetostatic potential [86].

The same plots show the time step selection behaviour. During relaxation (negative time in the figures) the time step has a large value initially, which decreases as the field decreases and the relaxation takes place, then become very large once

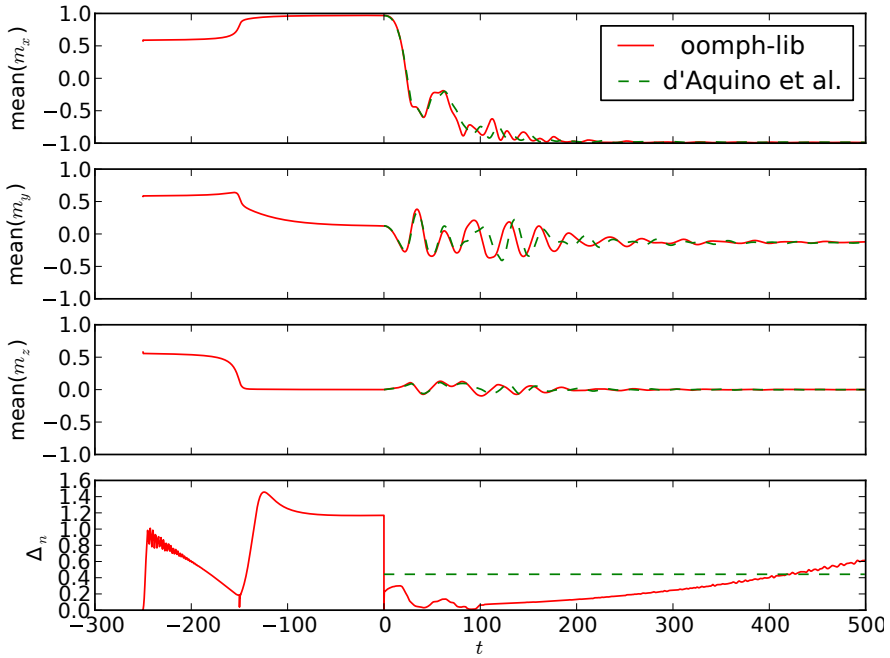

Figure 8.16: Evolution of the mean magnetisation for the μmag problem #4 with field 2 using monolithic IMR with nodal quadrature. For comparison we also show the results submitted to the μmag website by d'Aquino et al. [73].

the field reaches zero. During the dynamics (positive time in the figures): a reasonable large time step is selected initially. It then decreases around the time of the peak in $\mathbf{m}$ and steadily increases as the oscillations are damped out. The behaviour observed for the second field is essentially the same.

We show a comparison of the solution generated using IMR with nodal quadrature and monolithic or decoupled approach in Figure 8.17. The dynamics are very similar, but the monolithic method is able to take larger time steps and has less noise in the size step during the initial relaxation.

In Figure 8.18 we show a comparison of the three time integration schemes (with Gaussian quadrature, monolithic magnetostatics and $n_x = 100$) for field 2. We can see that all three of the schemes give very similar magnetisation values. However, IMR takes significantly smaller time steps during the initial relaxation phase ($t \sim -250$). This is likely due to the order reduction effect (see Section 3.4.6): since the field is initially very large, the Jacobian will contain large terms resulting in a reduction of accuracy for IMR. The adaptive time integration scheme detects and accounts for this effect by reducing the time step size.

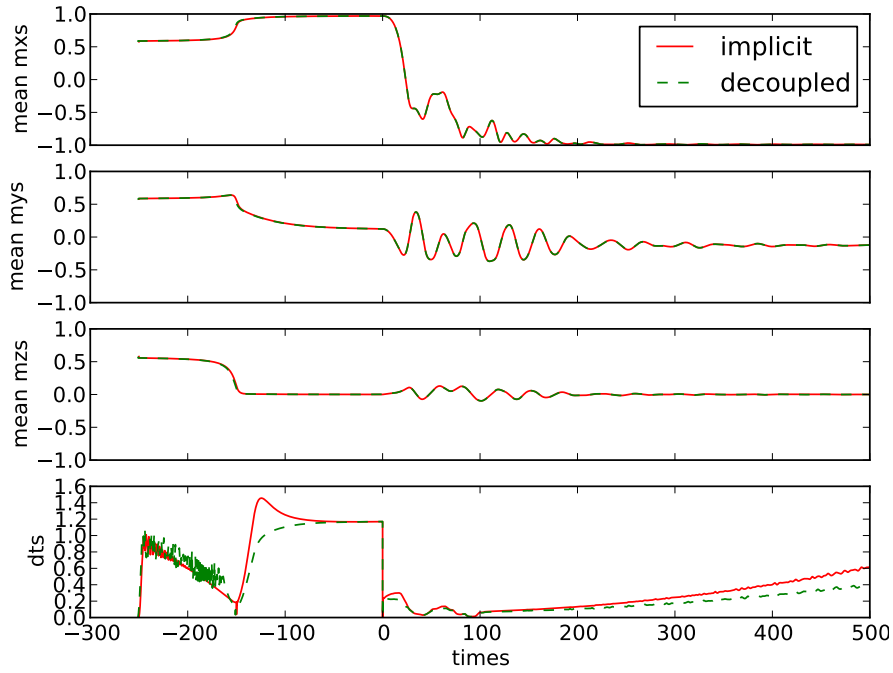

Figure 8.17: Evolution of the mean magnetisation for the μmag problem #4 with field 2 using IMR with nodal quadrature and monolithic (implicit) or semi-implicit (decoupled) coupling to the magnetostatics problem.

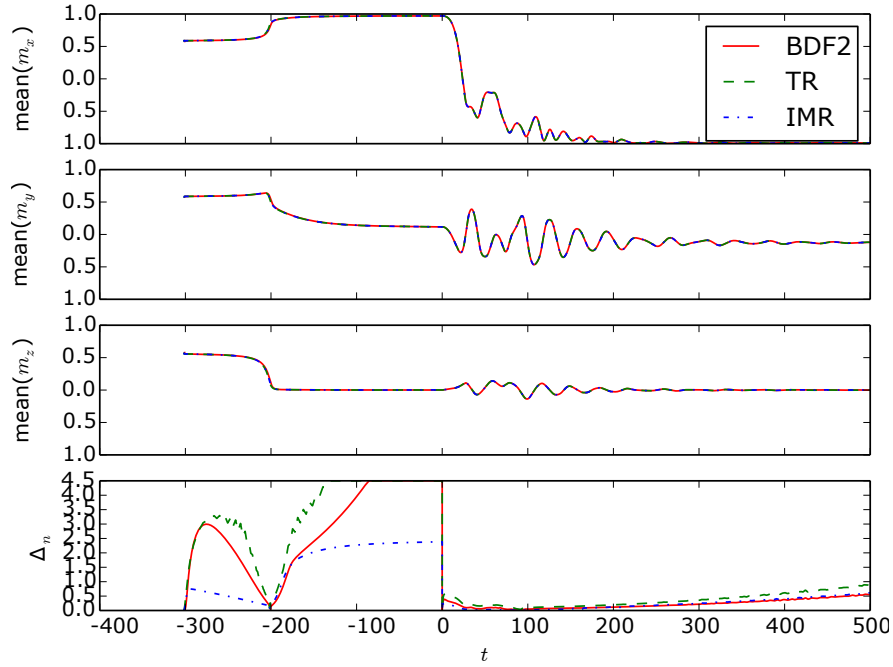

Figure 8.18: Evolution of the mean magnetisation for the μmag problem #4 with field 2 using various time integration schemes, Gaussian quadrature, monolithic coupling, and $n_x = 100$.

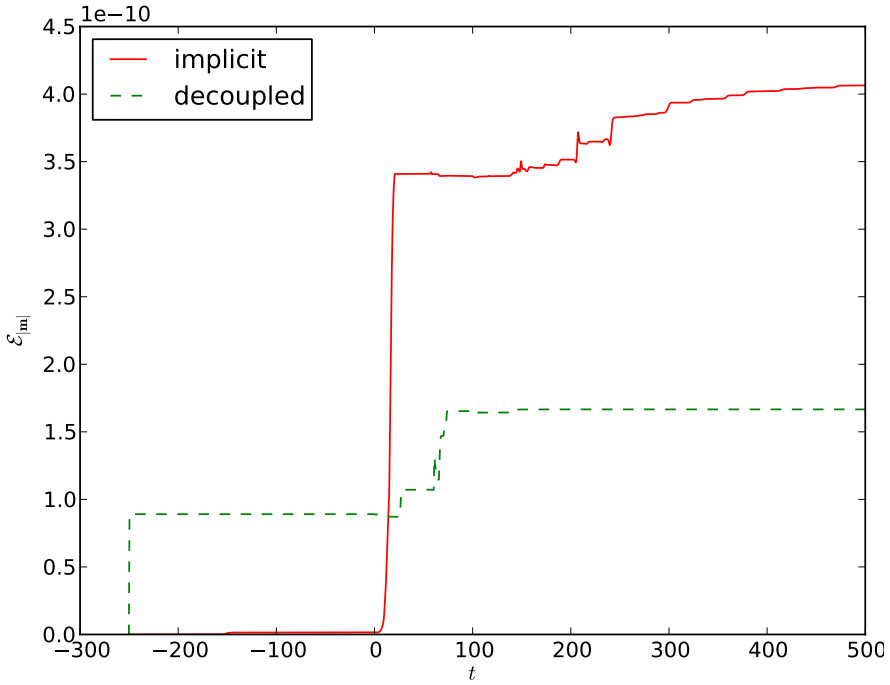

Figure 8.19: Evolution of $\mathcal{E}_{|\mathbf{m}|}$ with IMR, nodal quadrature and both coupling strategies for the μmag problem #4 with field 2.

8.3.4 Geometric integration properties

The evolution of the maximum nodal error in the magnetisation length for the decoupled and implicit methods is shown in Figure 8.19. As expected both coupling approaches conserve $|\mathbf{m}|$ to around the Newton tolerance.

The evolution of the energy of the system when $\alpha = 0$ is shown in Figure 8.20. It can be seen that neither the monolithic or decoupled methods give the desired conservation behaviour. From various other experiments we have drawn the conclusion that this occurs when the magnetostatic calculations are introduced.

We believe that the lack of energy conservation is due to the asymmetry of the discrete BEM operator. This asymmetry be expected to cause issues with energy conservation because the symmetry property (3.39) no longer holds, and the energy property (3.49) cannot be derived. Standard collocation based discretisation approaches, as used in our discretisation of the BEM operator, do not preserve the symmetry of the underlying operators. A symmetric discrete BEM operator could be obtained by applying a *symmetric Galerkin* BEM discretisation [12] [101, p. 75] to the formulation discussed in Chapter 5. Alternatively a standard

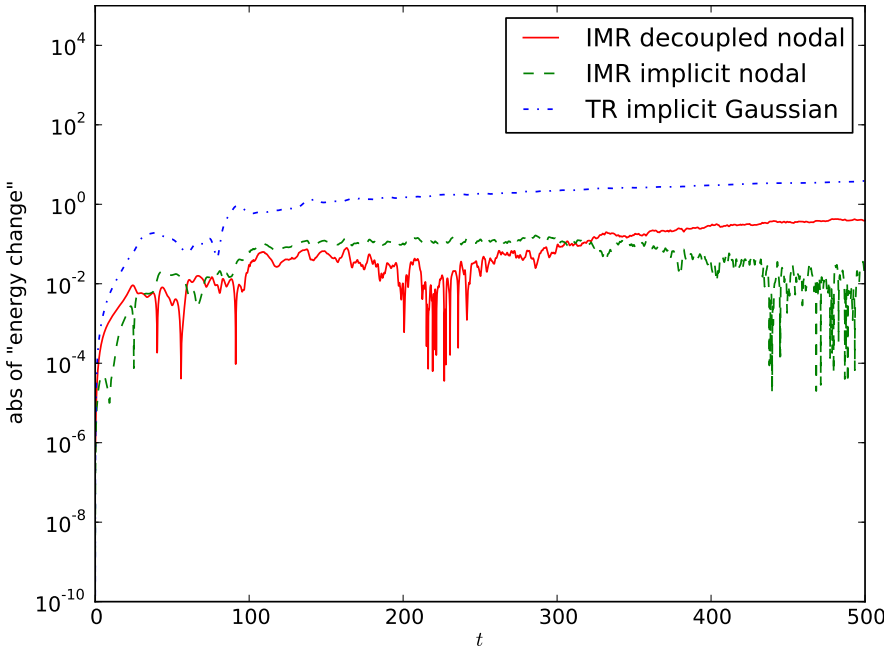

Figure 8.20: Evolution of the error in energy with various time integration schemes, quadratures, and coupling strategies for the undamped μmag problem #4 with field 2. Results for other combinations of methods are very similar.

Galerkin BEM discretisation combined with the single-layer-potential formulation described by García-Cervera and Roma [41] [60, p. 19] would result in a symmetric discrete operator without the requirement to evaluate hyper-singular integrals.

8.3.5 Effect of nodal quadrature scheme on spatial convergence

In this section we compare the results when using nodal quadrature and Gaussian quadrature. We focus on the case of field 2 and the use of monolithic coupling. In Figure 8.21 we show the mean y component of the magnetisation for a range of mesh refinements and integration schemes around the time when the results begin to deviate from each other. Varying the tolerance of the adaptive integrator ($\epsilon_\Delta = 10^{-3}, 10^{-4}, 10^{-5}$) or the time integration scheme has almost no effect on these results. Hence we can assume that the results with Gaussian quadrature and the highest refinement level are the most accurate. Based on this we can see that the use of nodal quadrature results in lower accuracy for the same level of

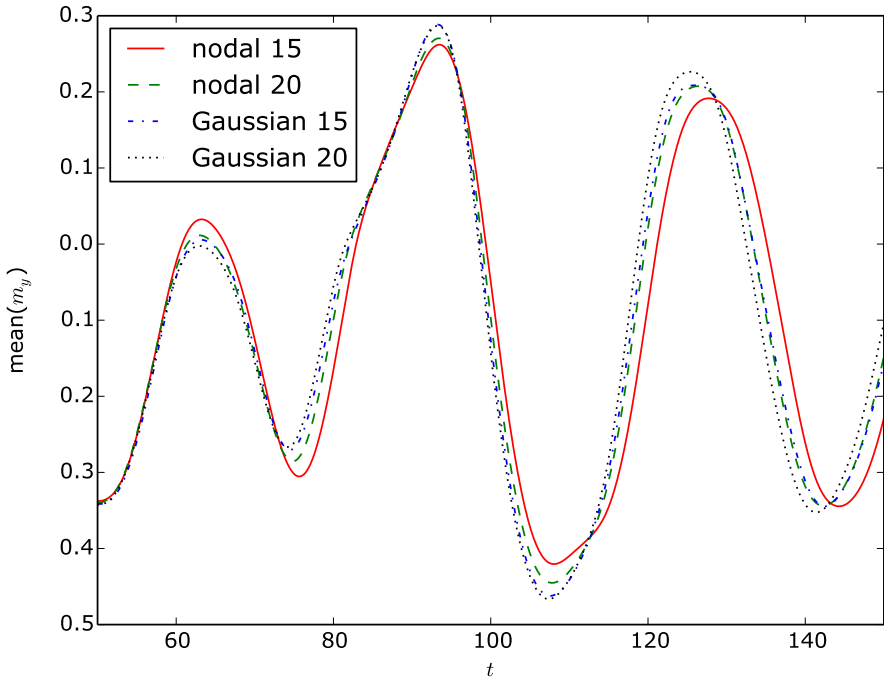

Figure 8.21: Mean y component of the magnetisation for the μmag problem 4 with field 2 solved using IMR, monolithic coupling, both quadrature schemes and varying spatial resolution n_x. The number in the legend indicates $n_x/5$.

spatial refinement.

Therefore we conclude that nodal quadrature can have a negative effect on the overall accuracy of the scheme. However, recall that our implementation of IMR with magnetostatics does not conserve energy (see Section 8.3.4). So these results are not necessarily representative of the accuracy of an energy-conserving implementation.

8.3.6 Effectiveness of the linear and non-linear solver

In this section we examine the effectiveness of the iterative linear and non-linear solvers for the monolithic method introduced in Section 6.2.3 as applied to the μmag standard problem #4.

Before showing the results we describe how the number of iterations required by the linear and non-linear solvers affect the CPU time and memory usage of the overall method. When using implicit time integration schemes almost all of the computation time is spent within the non-linear (*i.e.* Newton-Raphson)

solve in each time step, hence the speed and number of these solves is critical to performance. Within a Newton-Raphson step the linear (Jacobian) solve time typically dominates the time taken,[1] the number of Jacobian solves that are required is equal to the number of Newton-Raphson iterations. Writing these relationships mathematically we have

$$\begin{aligned} T_{\text{total}} &\propto \cdot N_{\text{TimeSteps}} \cdot T_{\text{Newton}}, \\ &\propto N_{\text{TimeSteps}} \cdot N_{\text{NewtonIterations}} \cdot T_{\text{JacobianSolve}}. \end{aligned} \tag{8.11}$$

When using GMRES the linear solve time is proportional to the square of the number of linear solver iterations (see Section 3.6.2), as well as to the total number of nodes N (because the number of non-zero entries in the Jacobian is proportional to the number of nodes). Hence the total CPU time obeys a relationship of the form

$$T_{\text{total}} \propto N_{\text{TimeSteps}} \cdot N_{\text{NewtonIterations}} \cdot N^2_{\text{LinearIterations}} \cdot N. \tag{8.12}$$

In terms of memory usage the relationships are much simpler: neither the number of time steps or Newton iterations affects the memory used. If GMRES is used then additional linear solver iterations increase the memory usage linearly as described in Section 3.6.2. However, in FEM/BEM problems the boundary element matrix typically dominates the memory usage to the extent that this is irrelevant [95].

The mean number of iterations required for the convergence of the linear solver against the number of nodes in the problem is shown in Figure 8.22. Note that while the iteration count increases with the number of nodes, N, it remains reasonable for the problem sizes required for good accuracy in this example. The time taken to set up the preconditioner is displayed in Figure 8.23. Again it grows with the number of nodes but remains reasonable for all cases shown here. Also note that with nodal quadrature the preconditioner is cheaper to set up but less effective. To counteract this effect a higher level of ILU fill-in could be used with nodal quadrature.

Next we look at the number of iterations of the Newton-Raphson method required

[1] When highly effective solvers are available, *i.e.* small two dimensional problems, the Jacobian assembly time can become important. However, our current Jacobian assembly code is not well optimised *e.g.* an order of magnitude speed-up could be easily obtained by caching constant (linear) blocks as described in Section 6.2.1.

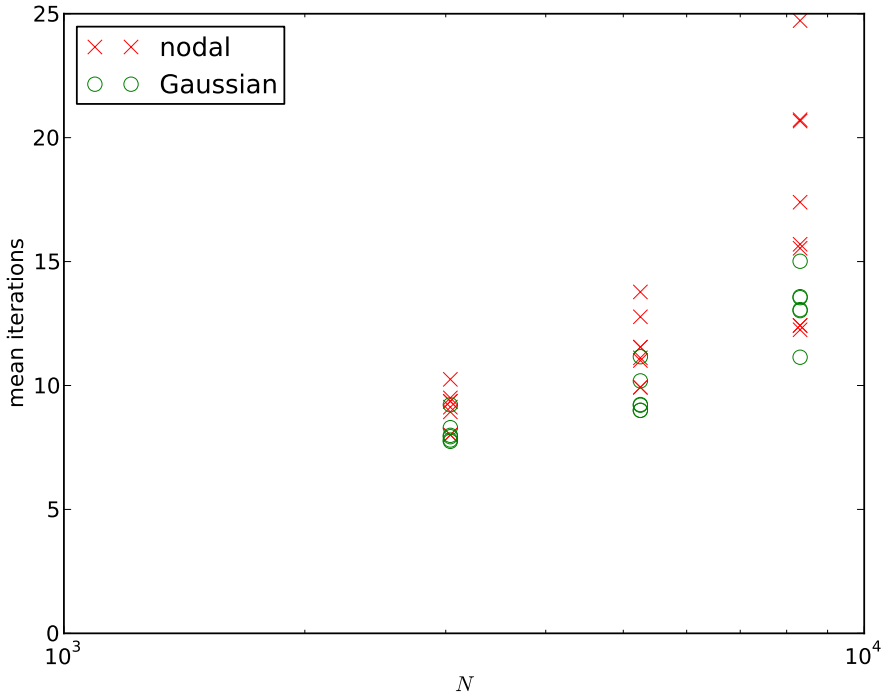

Figure 8.22: GMRES iterations to converge against problem size for the monolithic system using the inexact preconditioner $\widetilde{\mathcal{P}}_3$ with the various time integration schemes, quadratures, and applied fields for the μmag problem #4 (including the calculation of the relaxed initial condition).

to solve the monolithically coupled non-linear system. The mean number of iterations required for convergence is consistently just over 2 (and always less than 3) despite the extremely tight tolerance and the very large time steps (during the calculation of the initial condition and towards the end of the simulation). In particular, we note that the number of iterations required is significantly lower than the ~ 14 quasi-Newton iterations required for the monolithic method used in [24]. The various other example problems covered in this chapter required similar numbers of Newton-Raphson iterations.

We do not explore the total solve times of the various methods for this example. The solve times are dominated by the calculations involving the BEM matrix due to the fact that we have not used a hierarchical matrix representation of the dense block (because HLib is only compatible with triangular elements) and the extremely large number of boundary nodes. Hence any timing results would be meaningless for practical problems in which the FEM/BEM method is appropriate.

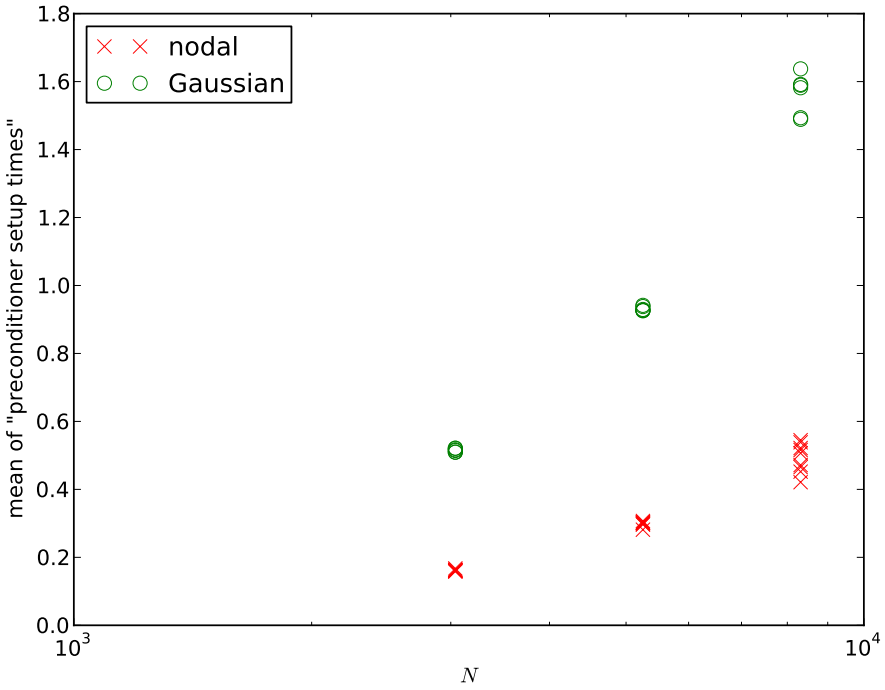

Figure 8.23: Time in seconds to set up the inexact preconditioner $\widetilde{\mathcal{P}_3}$ against problem size for the monolithic system using the inexact preconditioner $\widetilde{\mathcal{P}_3}$ with the various time integration schemes, quadratures, and applied fields for the μmag problem #4 (including the calculation of the relaxed initial condition).

8.3.7 Conclusions

Our methods give results for the μmag problem #4 which converge well and match up reasonably with the benchmark results of other models.

We observe the expected conservation of magnetisation length with IMR and nodal quadrature. However, we do not observe the expected conservation of energy. Asymmetry of the BEM part of the problem, and could be resolved by the use of alternative formulations and/or discretisation approaches.

Finally we have shown that, aside from issues with the lack of a hierarchical matrix implementation for rectangular boundary elements, we can efficiently solve the monolithic non-linear problem resulting from coupling the LLG with FEM/BEM magnetostatics calculations. Application of Newton-Raphson method results in around 2.2 Newton iterations per non-linear solve on average. The use of GMRES preconditioned by $\widetilde{\mathcal{P}_3}$ from Section 6.2.3 results in around 15 Krylov iterations per linear solve, with a reasonable preconditioner setup time.

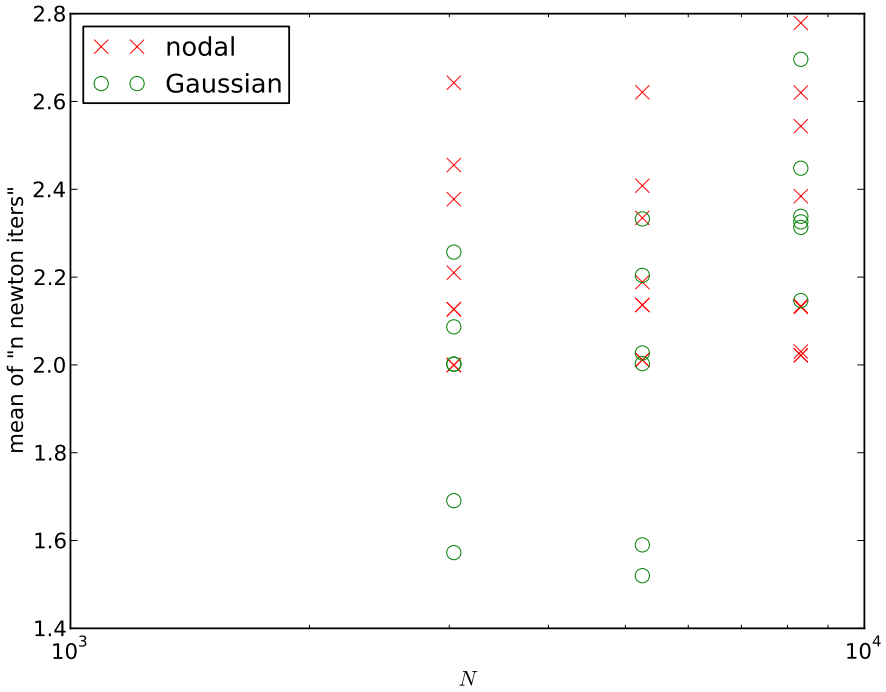

Figure 8.24: Newton-Raphson iterations to converge against problem size for the monolithic system with the various time integration schemes, quadratures, and applied fields for the μmag problem #4 (including the calculation of the relaxed initial condition).

8.4 Conclusions

In this chapter we have demonstrated the effectiveness of the methods described and developed throughout this thesis on micromagnetic problems with spatial variations.

We observed that our adaptive IMR algorithm selects appropriate time steps even for complex problems involving exchange and magnetostatics. We have shown that IMR with nodal quadrature retains its magnetisation length conservation property on a number of problems, but also that it is less effective on certain meshes of triangular elements. Similarly IMR (with nodal quadrature) provides energy conservation for problems without FEM/BEM magnetostatics, but that the energy conservation fails when collocation FEM/BEM magnetostatics calculations are included. This issue is likely due to the asymmetry of the discrete BEM operator and could be avoided by using an alternative formulation and/or discretisation of the BEM problem.

Our methods converge exactly as expected to an exact solution of the LLG with

exchange effective field. Their accuracy on the µmag standard problem #4 with the first applied field also appears to be very good. For the second field the results agree well with a finite difference based method using a similar time integration scheme only up to $t = 100$, but all of the solutions submitted to the µmag website disagree around this time.

Chapter 9

Discretisation induced stiffness in the Landau-Lifshitz-Gilbert equation

For continuum mechanics models of the LLG equation stiffness (the restriction of time step size by stability rather than by the desired accuracy) has long been recognised as an issue, at least for some problems [75].

Stiffness in time-dependent PDEs has at least two possible sources: firstly a problem may be physically stiff due to large differences in the characteristic time scales of different (physical) components of the solution [54, Chap. 4]; secondly the choice of spatial discretisation method may be the cause of stiffness. In particular a fine spatial discretisation often results in a stiff system of ODEs. An intuitive explanation for this is that a finer mesh resolves shorter wavelength modes, even if they have almost no effect on the resulting solution. Shorter wavelength implies higher frequency, *i.e.* shorter characteristic time scales. These modes interact with the solution, and the large variation in time scales causes stiffness in a similar way to physical effects. For a more rigorous discussion of this effect in terms of the eigenvalues of the time integration operator see e.g. [7, Sec 8.2].

As discussed in Section 3.4 time integration methods can be roughly divided into two classes: those which are inefficient when applied to the solution of stiff problems and those which are not. These two classes correspond to *explicit* methods that calculate the value at the next time exclusively in terms of previous values

and *implicit* methods in which a system of equations must be solved. Typically one step of an implicit method requires more computational effort than a step of an explicit method because of this solve. However good implicit methods are unconditionally stable, allowing them to take much larger time steps when applied to stiff problems [54, Chap. 4]. Hence there is a trade-off between time step size and the computation time per step. Since the optimal choice of method depends on the stiffness it is important to understand its origins in micromagnetic simulations.

In this chapter we examine numerically the relationship between stiffness and spatial discretisation size for a simple micromagnetic problem where stiffness due to physical effects can be ruled out. We also explore how stiffness is affected by the use of the FEM/BEM method for magnetostatic calculations. Finally, we study the effects of discretisation on stiffness in methods which combine explicit magnetostatic calculations with implicit exchange field and LLG calculations (*i.e.* semi-implicit methods).

The results in this chapter have also been published in the IEEE Transactions on Magnetics journal [89].

9.1 Problem definition

As our case study we choose a sphere with a radius of one exchange length. The initial magnetisation is $\mathbf{m} = [0.2, 0, 1.0]/|\mathbf{m}|$, the applied field is $\mathbf{h}_{\text{ap}} = [0, 0, -1.1]$. We considered three values for the Gilbert damping constant: $\alpha = 1, 0.1$ and 0.01. With this geometry and uniform magnetisation the magnetostatic field can be analytically shown to be $\mathbf{h}_{\text{ms}} = -\mathbf{m}/3$ throughout the domain (as mentioned in Section 7.4) [1, p. 112], and so, due to energy considerations, the magnetisation remains uniform for spheres of radius $R < 2.082\sqrt{3} \sim 3.606$ exchange lengths [52, p. 211].

The dynamics are therefore very simple: the magnetisation processes around the z-axis while gradually damping towards the applied field (along the negative z-axis). This simplicity means that the problem can also be written as an ODE, allowing for useful comparisons. Additionally exact solutions for the switching time are known [70], allowing quantification of the error.

As mentioned above, the simple geometry of this problem means that the physics can be captured, without any loss of accuracy, by an ODE version of the LLG:

$$\begin{aligned} \frac{\partial \mathbf{m}}{\partial t}(1+\alpha^2) &= -\mathbf{m}\times\mathbf{h} - \alpha\mathbf{m}\times(\mathbf{m}\times\mathbf{h}), \\ \mathbf{h} &= \mathbf{h}_{\text{ap}} - \mathbf{m}/3, \end{aligned} \tag{9.1}$$

where $\mathbf{m} = \mathbf{m}(t)$ and we have used the Landau-Lifshitz form for ease of implementation.

9.2 Implementation details

In these experiments we use the FEM discretisation of the non-dimensional LLG as discussed in Chapter 4. Magnetostatics calculations (when in use) are handled using either the FEM/BEM discretisation as discussed in Chapter 5 or the analytical formula $\mathbf{h}_{\text{ms}} = -\mathbf{m}/3$. We use three time integration schemes. The first is the standard IMR as discussed in Section 3.4.2, with monolithic coupling to the magnetostatic calculations as described in Section 6.2.3. The second scheme is IMR with semi-implicit (decoupled) magnetostatic calculations as discussed in Section 6.2.2.

The third scheme is an explicit method. For explicit time integration we use the mathematically equivalent Landau-Lifshitz form of the LLG, (2.38) because the time derivative can be written explicitly, avoiding a non-linear solve. We use a two stage Runge-Kutta method (RK2), also known as Heun's method [7, p. 58]:

$$\begin{aligned} \mathbf{m}_* &= \mathbf{m}_n + \Delta_n f(t_n, \mathbf{m}_n), \\ \mathbf{m}_{n+1} &= \mathbf{m}_n + \frac{\Delta_n}{2}\left(f(t_n, \mathbf{m}_n) + f(t_{n+1}, \mathbf{m}_*)\right). \end{aligned} \tag{9.2}$$

The time derivatives required in (9.2) are calculated by inverting the finite-element mass matrix using a diagonally preconditioned conjugate gradient solver. The magnetostatic potential ϕ is recalculated using the FEM/BEM method at appropriate time and magnetisation values during each stage of (9.2). The Poisson solves required for the evaluation of the potentials u and ϕ use the methods and parameters discussed in Chapter 6.

Good quality (radius-to-edge ratio > 2) quasi-uniform unstructured tetrahedral

meshes of a sphere were generated using TetGen [90]. Meshes were refined by decreasing the maximum element volume parameter.

All simulations were run for 4 time units ($\approx$ 20ps) with a full reversal taking between 8 and 400 time units depending on the damping. We ran the simulations without magnetostatics, with FEM/BEM magnetostatics, and with the analytical magnetostatic field for each value of the Gilbert damping constant.

To find bounds for the maximum stable step size, $\Delta_{\max}$ we use a simple heuristic algorithm: The computation is repeated with a sequence of decreasing step sizes (halved each time) until a stable solution is observed, this stable step size is denoted Δ_a. A solution is considered to be unstable if the maximum angle between the magnetisation of neighbouring nodes is greater than $\pi/4$. This provides bounds on the maximum stable step of $\Delta_{\max} \in [\Delta_a, 2\Delta_a)$. To tighten these bounds we then use two steps of a standard binary search algorithm: the computation is run with $\Delta_n = 3\Delta_a/2$, if it is successful then $\Delta_{\max} \in [\frac{3\Delta_a}{2}, 2\Delta_a)$ otherwise $\Delta_{\max} \in [\Delta_a, \frac{3\Delta_a}{2})$.

The initial step size, Δ_{init}, is selected such that the temporal error is sufficiently small to obtain good accuracy (so any reduction in step size below this value is wasteful.

To assess physical stiffness and to find a time step size which gives sufficiently accurate results we solved the ODE form of the problem, (9.1), using the RK2 and IMR methods with $\Delta = 0.1$. Using IMR with $\alpha = 0.01$ we found a relative error in the final switching time of 0.3% (absolute error 1.2 time units $\approx$ 6ps), for other values of α the percentage error is even smaller. The relative error in switching time with $\alpha = 0.01$ using RK2 was 2.25%. Since these results are from an ODE calculation the error is only due to the time integration, with no contributions from spatial discretisation. Based on these results we conclude that $\Delta_{\text{init}} = 0.1$ gives a sufficiently small temporal error to be a reasonable time step size. Additionally, since no stability issues were seen for any value of α, we conclude that any lack of stability when integrating the equivalent PDE must arise purely from the spatial discretisation rather than the underlying physics.

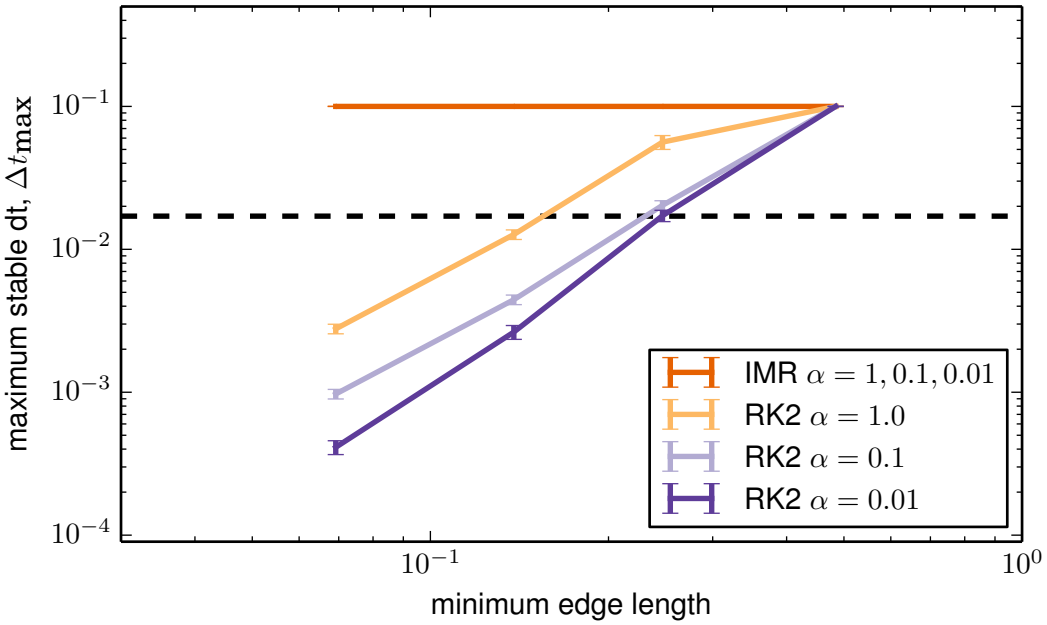

Figure 9.1: Maximum stable time step as a function of discretisation size for LLG without magnetostatics. Data points are $\Delta_{\max}$, the largest stable time step found. Error bars represent the range in which the largest stable time step is contained. The horizontal dashed line shows the time step where RK2 and IMR are equally efficient (IMR is more efficient when the maximum stable step of the RK2 method moves below this line). The maximum time step is limited to 0.1 for accuracy reasons.

9.3 Numerical results

The results of the experiments with $\mathbf{h}_{\text{ms}} = \mathbf{0}$ are shown in Figure 9.1. Results with full FEM/BEM magnetostatics and results with the analytical formula for magnetostatics are shown in Figure 9.2 and 9.3 respectively.

In all cases we find that as the damping constant is reduced by an order of magnitude, stable explicit time step sizes are reduced by approximately a factor of two (*i.e.* an increase in the stiffness).

Comparing Figure 9.1 and 9.2 we see that using FEM/BEM magnetostatics significantly increases the stiffness, requiring roughly an order of magnitude smaller explicit time steps for stability. However, from Figure 9.3 we see that adding the exact field does not induce stiffness, so we can conclude that this effect is due to the numerical methods used and not the magnetostatic field itself. From our

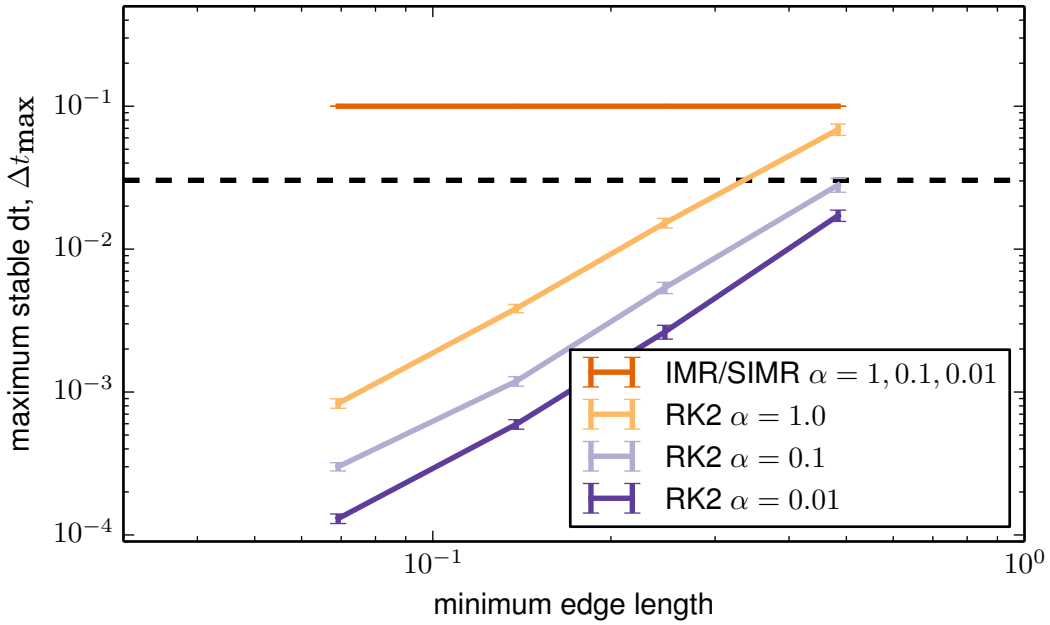

Figure 9.2: Stable time step against discretisation size for LLG with FEM/BEM magnetostatics.

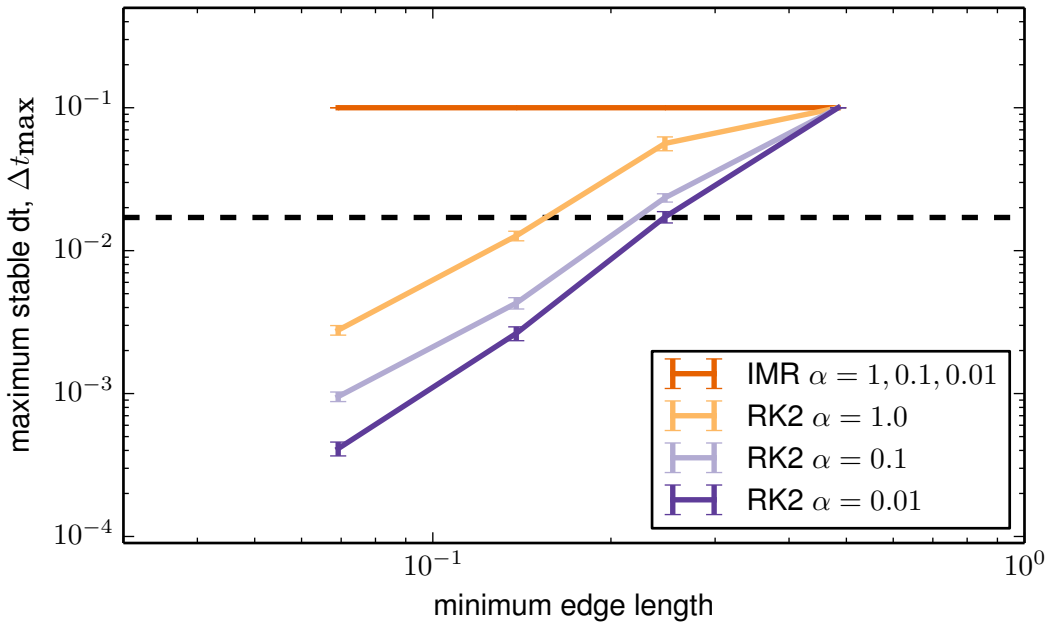

Figure 9.3: Stable time step against discretisation size for LLG with $\mathbf{h}_{ms} = -\mathbf{m}/3$.

data we cannot predict whether other methods of calculating the magnetostatic field, such as multipole methods, will result in similar increases in the number of explicit time steps required. However we expect that the effect is due to the coupling with the additional Poisson problems (which means that the semi-discretised equation is a differential algebraic equation [47, p. 371] rather than an ODE), thus any potential based method is likely to exhibit similar behaviour.

Figure 9.2 shows that using a semi-implicit method with explicit magnetostatic calculations (SIMR) imposes no stability restrictions on the time step due to spatial discretisation at the spatial resolutions relevant for micromagnetic problems for this example.

To further analyse our results we need an estimate of the ratio of computational effort for explicit vs implicit time steps. Without magnetostatics we find that each step of IMR takes on average 5.86 times more computation time than a step of RK2. With magnetostatics each step of SIMR only takes 3.40 times more computation time than a step of RK2 (the difference is due to the cost of solving multiple Poisson problems at each Runge-Kutta stage). Using these ratios we can calculate the stable RK2 time step required to have equivalent computational efficiency to IMR with step size 0.1. This stable step size is marked on Figure 9.1, 9.2 and 9.3 with a dashed line.

Based on these results we say that a problem is "stiff", and that an implicit method will perform significantly better than an explicit one, if the ratio of the desired time step, Δ_{init} to the maximum stable explicit time step, Δ_{max} is greater than 20. A caveat is that both types of model could be further optimised using, for example parallelism, improved preconditioning, mass lumping, hierarchical boundary matrix representations, etc.

Typical advice for the number of elements per exchange length is that an absolute minimum is one, and in order to show that the results are mesh independent the mesh should be refined a few times [33, Sec. 11]. This leads to a reasonable finest mesh with around three elements per exchange length.

We see that with FEM/BEM magnetostatics, realistic damping ($\alpha = 0.01$) and at least three elements per exchange length our case study is stiff. Without FEM/BEM magnetostatics stiffness only occurs if refinement to around five or more elements per exchange length is needed for any part of the domain. Problems that require this level of refinement include resolving the geometry in studies of

granular or patterned media [95] and resolving vortex-core-like structures [4].

On the other hand LLG problems can be only moderately stiff if refinement is only needed up to the level around one element per exchange length, as is often required for simple geometries. This is consistent with the fact that the µmag standard problem #4 is often solved using explicit time integration methods [73].

Similar experiments with the standard 4th order Runge-Kutta method [54, p. 41] and $\alpha = 1.0$ (plots not shown) give essentially the same results as for RK2. As α is reduced the maximum stable step size Δ_{max} is reduced, but not as rapidly as for RK2. However due to the increased computational cost per time step (a factor of two) the onset of stiffness for RK4 (for all α) occurs at roughly the same spatial discretisation as for RK2 with $\alpha = 0.1$.

Finally we point out that the discussion above assumes that the accuracy obtained with $\Delta = \Delta_{\text{init}}$ is sufficient. If a higher temporal accuracy than this is required then smaller time steps are needed regardless of stability, so the limitations imposed by stiffness are proportionally less significant.

9.4 Conclusions

Our results show that the LLG equation without magnetostatics or with analytical magnetostatic field calculations becomes stiff (*i.e.* implicit methods are significantly more efficient) as the number of elements per exchange length decreases below around 5. If FEM/BEM magnetostatic calculations are used stiffness occurs at much coarser discretisations, beginning at around $2-3$ elements per exchange length. In all cases decreasing the damping constant also increases the stiffness.

The results for the ODE version of the problem indicate that the observed stiffness is due to the spatial discretisation and not the physics of the problem. Since more complex physics is unlikely to ever reduce the stiffness, we expect that these results will extend to other, more complex, problems.

We also found that our semi-implicit FEM/BEM method does not suffer from discretisation induced stiffness in this simple test case.

Chapter 10

Conclusions and future work

10.1 Conclusions

In this thesis we have introduced a complete numerical method for the solution of dynamic micromagnetic problems using the finite element method with the Newton-Raphson linearisation, a variety of implicit time integration schemes, hybrid FEM/BEM magnetostatics calculations, and efficient iterative linear solvers. The methods have been validated against an analytical solution for a problem without magnetostatics and against the µmag standard problem #4 with magnetostatics.

In Chapter 7 we have demonstrated a novel and widely applicable adaptive time step selection algorithm for the implicit midpoint rule. We have also shown that the time step selection works well for a wide variety of ODE test cases, as well as for a number of PDE test cases using the LLG equation. Additionally we have shown that the geometrical integration properties of the constant time step IMR extend to the adaptive time step version.

In Sections 6.2, 6.3 and 8.3 we introduced and tested efficient, robust and scalable solution methods for the monolithically coupled LLG-FEM/BEM magnetostatics problem provided that a good preconditioner for the LLG sub-problem is available. Such monolithic couplings are required to obtain the energy property of the implicit midpoint rule and may offer advantages in stochastic integration methods.

In Sections 7.4 and 8.1 we studied the performance of common implicit time

integration schemes on some micromagnetic problems with exact solutions. We found that the overall accuracy of the BDF2 scheme is always poor compared to the TR and IMR schemes, this is probably due to a combination of the larger local truncation error and the spurious numerical damping of BDF2.

Finally in Chapter 9 we studied the effect of spatial discretisation on the relative performance of implicit and explicit time integration schemes (stiffness). We found that stiffness in micromagnetics can arise from the spatial discretisation alone, and that the stiffness increases as the element size is decreased, as expected from standard PDE theory. We also found that the introduction of FEM/BEM magnetostatics calculations increases the stiffness.

10.2 Future work

In the course of our numerical experiments we uncovered some issues with the geometrical integration properties of our complete model. Firstly we found that the conservation properties of IMR with FEM and nodal quadrature was much less effective when applied to meshes of triangular elements, at least in our implementation. This effect is fairly small and is not contradicted by any numerical results in the literature that we are aware of due to the common use of comparatively loose linearisation tolerances. An alternative implementation of IMR with FEM and nodal quadrature should be used with a tight linearisation tolerance in order to find out if this effect is an artefact of our implementation or a real issue.

Secondly, we found that when FEM/BEM magnetostatics calculations discretised by a collocation based approach were included the energy conservation property of IMR was lost. This is probably due to the asymmetry of the discrete BEM operator, which could be corrected by the use of alternative formulations of the method. In particular the use of the García-Cervera-Roma formulation [41] [60, p. 19] with a Galerkin discretisation approach [101, p. 75] should resolve this issue.

Once these issues have been corrected the relative performance of schemes with geometric integration properties, in terms of the accumulation of the temporal error, should be evaluated for realistic problems.

In the area of linear solvers a general, efficient, robust and scalable preconditioner for the Newton-Raphson linearised LLG equation is still required. One approach

to the construction of such a preconditioner could be to exploit the block structure of the Jacobian matrix and to use multigrid-based methods to approximate the Laplacian-like skew-symmetric off-diagonal blocks resulting from the exchange effective field. A less general approach for the case of granular or patterned media could be the use of a domain decomposition preconditioner. In such a method the small matrix block associated with each grain/island would be inverted by a direct solver and the combination used as a block diagonal preconditioner. Due to the weak coupling between grains/islands this would provide a good approximation for the inverse of the entire LLG block. With either of these enhanced LLG preconditioners the effectiveness of the preconditioner discussed in Chapter 6 on extremely large problems could be investigated.

In the time integration of the stochastic LLG only a few time integration schemes are known to converge to the correct solution, one of which is the implicit mid-point rule [25]. Since the use of a semi-implicit magnetostatics coupling modifies the time integration scheme a monolithic coupling scheme is required to maintain this property. The preconditioner developed in Chapter 6 should be tested in this capacity once effective preconditioners for the LLG sub-problem are available.

Appendix A

Analytical solutions of the Landau-Lifshitz-Gilbert equation

When testing numerical methods for differential equations it is very helpful to be able to compare the results with known analytical solutions. This appendix contains some solutions useful for this purpose.

A.1 Ellipsoidial nano-particle

In a 2000 paper [70] Mallinson gives an analytical equation for the time taken for magnetisation to "switch" from one polar angle (angle to the z-axis) to another. He also gives the azimuthal angle (the angle "around the z-axis") rotated through during this switching.

The conditions for the model to apply are:

1. A sufficiently small particle that the exchange field dominates and the magnetisation is uniform.

2. Constant applied field aligned with the z-axis.

3. Uniaxial magnetocrystalline anisotropy: $\mathbf{H}_{\text{ca}} = \frac{2K}{\mu_0 M_s}(\mathbf{M} \cdot \hat{\mathbf{z}})\hat{\mathbf{z}}$.

4. A prolate ellipsoidal particle aligned with the z-axis.

Let θ_1, θ_2 be the initial and final angles between the z-axis and the magnetisation (*i.e.* initial and final polar angles in the spherical polar coordinate system with the field axis as the main axis). Let H_k be the combined anisotropy field strength: $H_k = \frac{2K}{M_s} + M_s(N_\perp - N_\parallel)$ where N is the demagnetisation tensor of the ellipsoid and all other symbols have their usual meanings. Then the time taken to switch from θ_1 to θ_2 is given by

$$\tau = \frac{\alpha^2+1}{|\gamma_{\mathrm{L}}|\alpha}\frac{1}{H^2-H_k^2}\left(H\ln\left[\frac{\tan(\theta_2/2)}{\tan(\theta_1/2)}\right] + H_k\ln\left[\frac{H-H_k\cos\theta_1}{H-H_k\cos\theta_2}\right] + H_k\ln\left[\frac{\sin\theta_2}{\sin\theta_1}\right]\right). \tag{A.1}$$

The azimuthal angle precessed through during this switching is

$$\phi = -\frac{1}{\alpha}\ln\left[\frac{\tan(\theta_2/2)}{\tan(\theta_1/2)}\right]. \tag{A.2}$$

Note that this is not really a true analytical solution to the Landau-Lifshitz-Gilbert equation: it gives the switching time and azimuthal angle as a function of polar angle rather than the magnetisation direction as a function of time. However comparing switching time values with those generated by a model is still a useful measure of the error. Additionally the magnetisation at some given time could be calculated from (A.1) using *e.g.* the Newton-Raphson method. Alternatively, using further simplifying assumptions, this solution can be reduced to a "true" analytical solution in the sense that $\mathbf{m}$ can be written analytically as a function of time. This is discussed in Section 7.4.1.

A.2 Wave-like solution in an infinite domain

These solutions are taken from [56], [40], the solution appears to have been originally published in [64] in a more abstract form. For most parameters the solution is just a wave-like excitation in $m_x(\mathbf{x},t)$ and $m_y(\mathbf{x},t)$ whose oscillations are damped towards a uniform state $m_z = \pm 1$. This is the only exact solution of the LLG equation with the exchange effective field which has non-trivial $\mathbf{x}$ dependence, to the best of our knowledge.

For an infinite magnetic domain (*i.e.* using periodic boundary conditions) of spatial dimension D choose constants $c \in \mathbb{R}$ and $\mathbf{k} \in \mathbb{R}^D$. We denote

$$\begin{aligned} \hat{t}(t) &= \frac{t}{1+\alpha^2}, \\ b(t) &= |\mathbf{k}|^2 \alpha \hat{t}(t), \\ d(t) &= \sqrt{\sin^2(c) + \cos^2(c) e^{2b(t)}}, \\ g(t) &= \frac{1}{\alpha} \log_e \left[\frac{d(t) + \cos(c) e^{b(t)}}{1+\cos(c)} \right]. \end{aligned} \tag{A.3}$$

Then

$$\begin{aligned} m_x(t) &= \frac{1}{d(t)} \sin(c) \cos\left(\mathbf{k}\cdot\mathbf{x} + g(t)\right), \\ m_y(t) &= \frac{1}{d(t)} \sin(c) \sin\left(\mathbf{k}\cdot\mathbf{x} + g(t)\right), \\ m_z(t) &= \frac{1}{d(t)} \cos(c) e^{b(t)}, \end{aligned} \tag{A.4}$$

is a solution of the Landau-Lifshitz-Gilbert equation with $\mathbf{h}_{\text{eff}} = \nabla^2 \mathbf{m}$.

The rescaling of time given by $\hat{t}(t)$ is needed to convert from a solution of the Landau-Lifshitz equation (2.41) to a solution of the Landau-Lifshitz form of the Landau-Lifshitz-Gilbert equation (2.39), *i.e.* to account for the correction to the form of the damping as introduced by Gilbert.

The function $d(t)$ is a normalisation parameter to keep $|\mathbf{m}(\mathbf{x},t)| = 1$ everywhere. To see this: note that $\sin^2(c)\cos^2\left(f(\mathbf{x})\right) + \sin^2(c)\sin^2\left(f(\mathbf{x})\right) = \sin^2(c)$, hence the length contribution of the combined x and y terms is $\sin^2(c)$. The vector $\mathbf{k}$ is the wave vector, it determines the direction and speed of propagation of the wave. The parameter c represents in some way the fraction of the initial magnetisation which takes part in the oscillations. If $c = 0$ then $m_x, m_y = 0$ and there is no wave.

Figure A.1 shows the solution in one dimension over time at $\mathbf{x} = 0$ with parameters $k = 2\pi$, $\alpha = 0.05$ and varying c. As c increases the initial value of m_z decreases and the amplitude of the oscillations in m_x and m_y increase.

As c approaches $\frac{\pi}{2}$ the behaviour becomes more complex, this is shown in Figure A.2. Essentially there is an additional initial exponential decay towards a state with moderate m_z after which the dynamics are similar to the case for lower c. At $c = \frac{\pi}{2}$ we have $\mathbf{m}(\mathbf{x},t) = (\cos(\hat{\mathbf{k}}\cdot\mathbf{x}), \sin(\hat{\mathbf{k}}\cdot\mathbf{x}), 0)$ and there is no

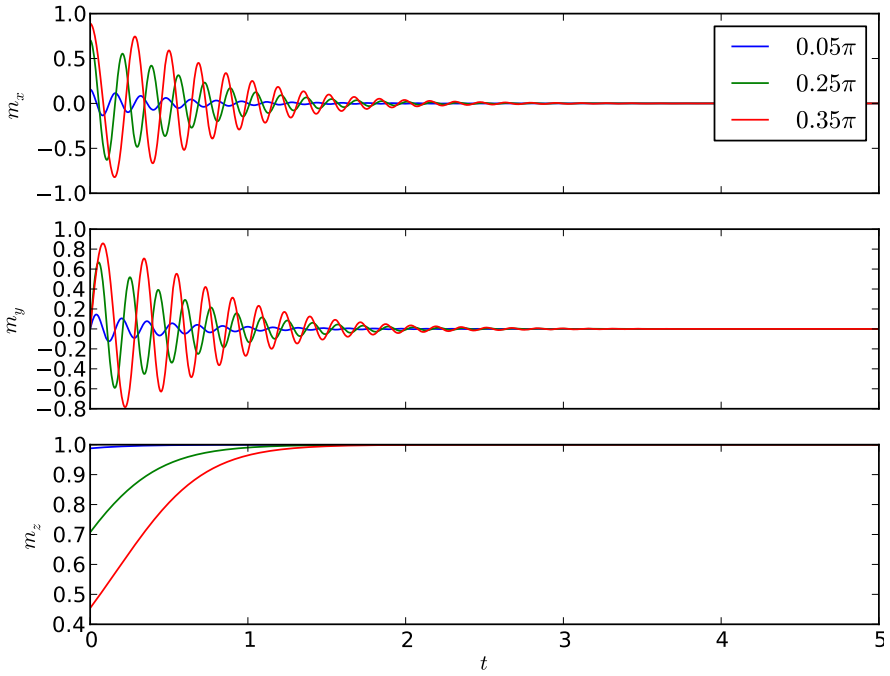

Figure A.1: Magnetisation values over time at $x = 0$ for the one dimensional wave solution with $k = 2\pi$, $\alpha = 0.05$ and various c (taken well away from $\frac{\pi}{2}$).

dynamical behaviour. Above $c = \frac{\pi}{2}$ the initial m_z becomes negative and the behaviour is symmetrical with that below $c = \frac{\pi}{2}$

In the limit of $\alpha \to 0+$ the solution becomes [40]:

$$\begin{aligned} m_x &= \sin(c)\cos\left(\mathbf{k}\cdot\mathbf{x} + |\mathbf{k}|^2\cos(a)t\right), \\ m_y &= \sin(c)\sin\left(\mathbf{k}\cdot\mathbf{x} + |\mathbf{k}|^2\cos(a)t\right), \\ m_z &= \cos(c). \end{aligned} \tag{A.5}$$

This is just a simple periodic wave in space and time.

Note that the behaviour in higher dimensions is essentially the same as in 1D except that the direction of propagation is in the direction described by the vector $\hat{\mathbf{k}}$.

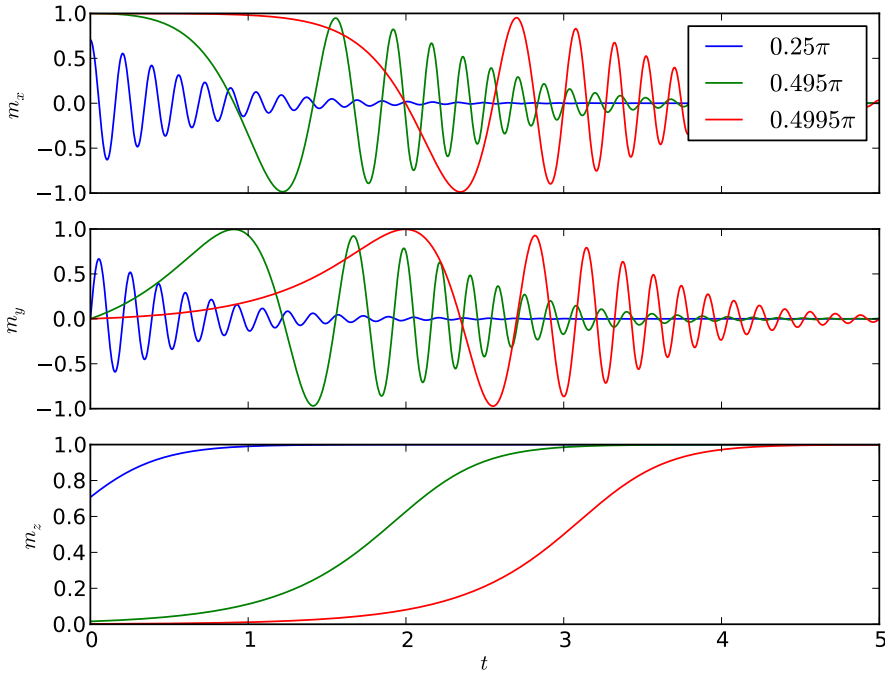

Figure A.2: One dimensional wave solution values over time at $x = 0$ *with* $k = 2\pi$, $\alpha = 0.05$ *and* c *approaching the critical value* $\frac{\pi}{2}$.

Appendix B

Properties of the effective field operator

In this appendix we show some properties of the effective field operator that are used in Section 3.4.9. It seems possible that these properties could be proven for a general effective field using only the definition (2.7), but we have not managed to find such a proof.

B.1 Linear Symmetrical operator

An operator $\mathscr{L}$ is linear if

$$\mathscr{L}[\mathbf{a} + c\mathbf{b}] = \mathscr{L}[\mathbf{a}] + c\mathscr{L}[\mathbf{b}]. \tag{B.1}$$

The vector Laplace, magnetostatic field and magnetocrystalline anisotropy operators are all easily demonstrated to be linear because they involve only linear operations, such as derivatives, integrals and dot products.

An operator $\mathscr{L}$ is symmetrical if

$$(\mathscr{L}\mathbf{a},\, \mathbf{b}) = (\mathbf{a},\, \mathscr{L}\mathbf{b})\,, \tag{B.2}$$

for all functions $\mathbf{a}, \mathbf{b} \in$ in some function space on the real numbers where $(\cdot,\, \cdot)$ is defined.

In this section we frequently make use of a consequence of the divergence theorem:

$$\int_\Omega \mathbf{f}(\mathbf{x}) \cdot \nabla g(\mathbf{x}) \, \mathrm{d}\Omega = \int_\Gamma g(\mathbf{x}) \left(\mathbf{f}(\mathbf{x}) \cdot \hat{\mathbf{n}}(\mathbf{x})\right) \mathrm{d}\Gamma - \int_\Omega g(\mathbf{x}) \, \nabla \cdot \mathbf{f}(\mathbf{x}) \, \mathrm{d}\Omega, \tag{B.3}$$

where $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^d$ and $g : \mathbb{R}^d \to \mathbb{R}$.

Note that by substituting $\mathbf{f} = \nabla f$ into (B.3) we can derive

$$\begin{aligned} \int_\Omega (\nabla^2 f) g \, \mathrm{d}\Omega &= \int_\Gamma g(\nabla f \cdot \hat{\mathbf{n}}) \, \mathrm{d}\Gamma - \int_\Omega \nabla f \cdot \nabla g \, \mathrm{d}\Omega, \\ &= \int_\Gamma g \frac{\partial f}{\partial \hat{\mathbf{n}}} \, \mathrm{d}\Gamma - \int_\Omega \nabla f \cdot \nabla g \, \mathrm{d}\Omega. \end{aligned} \tag{B.4}$$

B.1.1 Applied field

The applied field part of the effective field is in general *not* linear or symmetrical because it is independent of $\mathbf{m}$.

B.1.2 Vector Laplace operator

Theorem 1. *If $\mathscr{L}$ is a symmetric operator on $v \in L^2$ then so is its "vector equivalent", $\bar{\mathscr{L}}[\mathbf{v}] = \begin{pmatrix} \mathscr{L} v_1 \\ \mathscr{L} v_2 \\ \mathscr{L} v_3 \end{pmatrix}$.*

Proof.

$$\begin{aligned} \left(\bar{\mathscr{L}}\mathbf{a},\, \mathbf{b}\right) &= \int_\Omega \bar{\mathscr{L}}\mathbf{a} \cdot \mathbf{b} \, \mathrm{d}\Omega, \\ &= \int_\Omega \mathscr{L}[a_1] b_1 \, \mathrm{d}\Omega + \int_\Omega \mathscr{L}[a_2] b_2 \, \mathrm{d}\Omega + \int_\Omega \mathscr{L}[a_3] b_3 \, \mathrm{d}\Omega, \\ &= \left(\mathscr{L}[a_1],\, b_1\right) + \left(\mathscr{L}[a_2],\, b_2\right) + \left(\mathscr{L}[a_3],\, b_3\right). \end{aligned} \tag{B.5}$$

So $\bar{\mathscr{L}}$ is symmetrical if and only if $\mathscr{L}$ is symmetrical. □

Theorem 2 (Symmetry of Laplace operator)**.** *If $m_i \in L^2$ and $\frac{\partial m_i}{\partial \hat{\mathbf{n}}} = 0$ on all of Γ then ∇^2 is a linear operator on m_i.*

Proof. Apply equation (B.4) twice: first with $f = a$, $g = b$, then the other way around.

$$\begin{aligned}\left(\nabla^2 a,\, b\right) &= \int_\Omega \left(\nabla^2 a\right) b \,\mathrm{d}\Omega, \\ &= \int_\Gamma b\frac{\partial a}{\partial \hat{\mathbf{n}}} \,\mathrm{d}\Gamma - \int_\Omega \nabla a \cdot \nabla b \,\mathrm{d}\Omega, \\ &= \int_\Gamma b\frac{\partial a}{\partial \hat{\mathbf{n}}} \,\mathrm{d}\Gamma + \int_\Omega a(\nabla^2 b) \,\mathrm{d}\Omega - \int_\Gamma a\frac{\partial b}{\partial \hat{\mathbf{n}}} \,\mathrm{d}\Gamma, \\ &= \int_\Omega a(\nabla^2 b) \,\mathrm{d}\Omega.\end{aligned} \tag{B.6}$$

□

From these two theorems we see that the vector Laplacian operator ∇^2 is symmetrical. Note that the above does not include the case with surface anisotropy or when the length of $\mathbf{m}$ is not constant because in either of these cases we do not necessarily have $\frac{\partial \mathbf{m}}{\partial \hat{\mathbf{n}}} = 0$. The case with periodic boundary conditions is true by the combination of (4.43) and (B.6).

B.1.3 Magnetostatic field operator

For simplicity we write

$$k = \frac{1}{4\pi \left|\mathbf{x} - \mathbf{x}'\right|}. \tag{B.7}$$

Theorem 3 (Symmetry of magnetostatic field operator)**.** *If* $\mathbf{a}, \mathbf{b} \in L^2$ *and* $\nabla\mathbf{a}, \nabla\mathbf{b} \in L^2$ *(i.e.* $\mathbf{a}, \mathbf{b} \in \mathcal{H}^1$*) then the operator*

$$\begin{aligned}\mathscr{H}_{\mathrm{ms}}[\mathbf{a}](\mathbf{x}) &= -\nabla\phi[\mathbf{a}](\mathbf{x}), \\ &= -\nabla\left[-\int_{\Omega'} k\nabla' \cdot \mathbf{a}(\mathbf{x}') \,\mathrm{d}\Omega' + \int_{\Gamma'} k\mathbf{a}(\mathbf{x}') \cdot \hat{\mathbf{n}}(\mathbf{x}') \,\mathrm{d}\Gamma'\right],\end{aligned} \tag{B.8}$$

where primes denote another set of coordinates, is symmetrical.

Proof. Essentially we apply identity (B.3), rearrange the result using the symmetry of the kernel, k, and apply the identity again in reverse. We drop the $\mathbf{x}$ argument from ϕ where it is obvious.

Using (B.3) we get

$$\begin{aligned}(\mathscr{H}_{\mathrm{ms}}[\mathbf{a}],\, \mathbf{b}) &= -\int_{\Omega} \mathbf{b}\cdot\nabla\phi[\mathbf{a}]\,\mathrm{d}\Omega,\\ &= -\int_{\Gamma}\phi[\mathbf{a}](\mathbf{b}\cdot\hat{\mathbf{n}})\,\mathrm{d}\Gamma + \int_{\Omega}\phi[\mathbf{a}](\nabla\cdot\mathbf{b})\,\mathrm{d}\Omega,\\ &= \int_{\Gamma}\int_{\Omega'} k(\nabla'\cdot\mathbf{a}(\mathbf{x}'))(\mathbf{b}(\mathbf{x})\cdot\hat{\mathbf{n}}(\mathbf{x}))\,\mathrm{d}\Omega'\,\mathrm{d}\Gamma\\ &- \int_{\Gamma}\int_{\Gamma'} k(\mathbf{a}(\mathbf{x}')\cdot\hat{\mathbf{n}}(\mathbf{x}'))(\mathbf{b}(\mathbf{x})\cdot\hat{\mathbf{n}}(\mathbf{x}))\,\mathrm{d}\Gamma'\,\mathrm{d}\Gamma\\ &- \int_{\Omega}\int_{\Omega'} k(\nabla'\cdot\mathbf{a}(\mathbf{x}'))(\nabla\cdot\mathbf{b}(\mathbf{x}))\,\mathrm{d}\Omega'\,\mathrm{d}\Omega\\ &+ \int_{\Omega}\int_{\Gamma'} k(\mathbf{a}(\mathbf{x}')\cdot\hat{\mathbf{n}}(\mathbf{x}'))(\nabla\cdot\mathbf{b}(\mathbf{x}))\,\mathrm{d}\Gamma'\,\mathrm{d}\Omega.\end{aligned} \tag{B.9}$$

Changing the order of the integrals gives (since $\mathbf{a}$, $\mathbf{b}$ and their derivatives are in L^2)

$$\begin{aligned}(\mathscr{H}_{\mathrm{ms}}[\mathbf{a}],\, \mathbf{b}) &= \int_{\Omega'}\int_{\Gamma} k(\mathbf{b}(\mathbf{x})\cdot\hat{\mathbf{n}}(\mathbf{x}))\,\mathrm{d}\Gamma(\nabla'\cdot\mathbf{a}(\mathbf{x}'))\,\mathrm{d}\Omega'\\ &- \int_{\Gamma'}\int_{\Gamma} k(\mathbf{b}(\mathbf{x})\cdot\hat{\mathbf{n}}(\mathbf{x}))\,\mathrm{d}\Gamma(\mathbf{a}(\mathbf{x}')\cdot\hat{\mathbf{n}}(\mathbf{x}'))\,\mathrm{d}\Gamma'\\ &- \int_{\Omega'}\int_{\Omega} k(\nabla\cdot\mathbf{b}(\mathbf{x}))\,\mathrm{d}\Omega(\nabla'\cdot\mathbf{a}(\mathbf{x}'))\,\mathrm{d}\Omega'\\ &+ \int_{\Gamma'}\int_{\Omega} k(\nabla\cdot\mathbf{b}(\mathbf{x}))\,\mathrm{d}\Omega(\mathbf{a}(\mathbf{x}')\cdot\hat{\mathbf{n}}(\mathbf{x}'))\,\mathrm{d}\Gamma'.\end{aligned} \tag{B.10}$$

Finally we swap $\mathbf{x}$ with $\mathbf{x}'$ (which we can do because k is symmetrical in its arguments) and collect terms with the same (outer) integral domain

$$\begin{aligned}(\mathscr{H}_{\mathrm{ms}}[\mathbf{a}],\, \mathbf{b}) &= \int_{\Omega}\phi[\mathbf{b}](\nabla\cdot\mathbf{a})\,\mathrm{d}\Omega - \int_{\Gamma}\phi[\mathbf{b}](\mathbf{a}\cdot\hat{\mathbf{n}})\,\mathrm{d}\Gamma,\\ &= (\mathscr{H}_{\mathrm{ms}}[\mathbf{b}],\, \mathbf{a})\,.\end{aligned} \tag{B.11}$$

□

B.1.4 Magnetocrystalline anisotropy

Here we only examine the case of uniaxial anisotropy as it is the most technologically relevant

$$\mathbf{h}_{\mathrm{ca}}[\mathbf{m}] = \mathcal{K}_1(\mathbf{m}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}}. \tag{B.12}$$

We can easily see that the operator is symmetrical by writing out the definitions

$$\begin{aligned}(\mathbf{h}_{\mathrm{ca}}[\mathbf{a}],\, \mathbf{b}) &= \int_\Omega \mathcal{K}_1(\mathbf{a}\cdot\hat{\mathbf{e}})(\hat{\mathbf{e}}\cdot\mathbf{b})\,\mathrm{d}\Omega,\\ &= \int_\Omega (\mathcal{K}_1(\mathbf{b}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}})\cdot\mathbf{a}\,\mathrm{d}\Omega,\\ &= (\mathbf{h}_{\mathrm{ca}}[\mathbf{b}],\, \mathbf{a})\,.\end{aligned} \tag{B.13}$$

B.2 Effective field and energy

A useful relationship between the effective field and the energy is:

$$\begin{aligned}\mathscr{H}[\mathbf{m}] &= \nabla^2\mathbf{m} + \mathbf{h}_{\mathrm{ms}} + \mathbf{h}_{\mathrm{ca}},\\ -\frac{1}{2}\,(\mathbf{m},\, \mathscr{H}[\mathbf{m}]) &= -\frac{1}{2}\int_\Omega \mathbf{m}\cdot\mathscr{H}[\mathbf{m}]\,\mathrm{d}\Omega = e,\end{aligned} \tag{B.14}$$

note that this does not include the applied field energy.

For the magnetostatic field this relationship is obvious from the definition of e_{ms}, (2.47). For a uniaxial magnetocrystalline anisotropy effective field the derivation is very simple:

$$-\frac{1}{2}\int_\Omega (\mathcal{K}_1(\mathbf{m}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}})\cdot\mathbf{m}\,\mathrm{d}\Omega = -\frac{\mathcal{K}_1}{2}\int_\Omega (\mathbf{m}\cdot\hat{\mathbf{e}})^2\,\mathrm{d}\Omega = e_{\mathrm{ca}}. \tag{B.15}$$

For the exchange effective field we use (B.6) to obtain

$$-\frac{1}{2}\int_\Omega \mathbf{m}\cdot\nabla^2\mathbf{m}\,\mathrm{d}\Omega = -\frac{1}{2}\int_\Gamma \mathbf{m}\cdot\frac{\partial\mathbf{m}}{\partial\hat{\mathbf{n}}}\,\mathrm{d}\Gamma + \frac{1}{2}\int_\Omega \nabla\mathbf{m}:\nabla\mathbf{m}\,\mathrm{d}\Omega. \tag{B.16}$$

Then applying Neumann or periodic boundary conditions on $\mathbf{m}$ gives the result

$$-\frac{1}{2}\int_\Omega \mathbf{m}\cdot\nabla^2\mathbf{m}\,\mathrm{d}\Omega = \frac{1}{2}\int_\Omega (\nabla\mathbf{m})^2\,\mathrm{d}\Omega = e_{\mathrm{ex}}. \tag{B.17}$$

Appendix C

Truncation error derivation for the implicit midpoint rule

In this appendix we give details of the derivation of the local truncation error of the implicit midpoint rule. We begin from (3.21) at the end of Section 3.4.3:

$$T_n^{\mathrm{IMR}} = \underbrace{\frac{\Delta_n^3}{24}\mathbf{y}'''(t_{n+\frac{1}{2}})}_{\mathrm{I}} + \underbrace{\Delta_n\left[\mathbf{y}'(t_{n+\frac{1}{2}}) - \mathbf{f}\left(t_{n+\frac{1}{2}}, \frac{\mathbf{y}(t_n)+\mathbf{y}_{n+1}}{2}\right)\right]}_{\mathrm{II}} + \mathrm{O}\big(\Delta_n^4\big). \quad \text{(C.1)}$$

In order to be able to cancel terms in (C.1) we now need to Taylor expand $\mathbf{f}\left(t_{n+\frac{1}{2}}, \frac{\mathbf{y}(t_n)+\mathbf{y}_{n+1}}{2}\right)$ in $\mathbf{y}$ about $\mathbf{y}(t_{n+\frac{1}{2}})$, *i.e.* we need an expansion of the form

$$\begin{aligned}\mathbf{f}(t_{n+\frac{1}{2}}, \frac{\mathbf{y}(t_n)+\mathbf{y}_{n+1}}{2}) &= \mathbf{f}(t_{n+\frac{1}{2}}, \mathbf{y}(t_{n+\frac{1}{2}}) + \boldsymbol{\delta}_n), \\ &= \mathbf{f}(t_{n+\frac{1}{2}}, \mathbf{y}(t_{n+\frac{1}{2}})) + F_{n+\frac{1}{2}} \cdot \boldsymbol{\delta}_n + \mathrm{O}\big(\boldsymbol{\delta}_n^2\big),\end{aligned} \quad \text{(C.2)}$$

where $F_{n+\frac{1}{2}} = F(t_{n+\frac{1}{2}}, \mathbf{y}(t_{n+\frac{1}{2}}))$ is a *matrix* of partial derivatives[1] of each element of $\mathbf{f}$ with respect to each element of the vector $\mathbf{y}$. Note that the $\mathbf{f}$ term is multiplied by an additional factor of Δ_n in (3.13). Hence in the derivation of $\boldsymbol{\delta}_n$ we can drop terms of order greater than $\mathrm{O}(\Delta_n^2)$ while retaining the same asymptotic accuracy. We now derive the required correction $\boldsymbol{\delta}_n$. From (C.2) we have

$$\boldsymbol{\delta}_n = \frac{\mathbf{y}(t_n)+\mathbf{y}_{n+1}}{2} - \mathbf{y}(t_{n+\frac{1}{2}}). \quad \text{(C.3)}$$

[1]Note that this is *not* equivalent to the Jacobian matrix associated with the Newton-Raphson method which is the derivative of the residual function with respect to each element of $\mathbf{y}$.

However, we cannot expand $\mathbf{y}_{n+1}$ to get $\boldsymbol{\delta}_n$ in terms of only values at the midpoint. Instead we use the LTE of IMR to rewrite (C.3) as

$$\boldsymbol{\delta}_n = \frac{\mathbf{y}(t_n) + \mathbf{y}(t_{n+1}) - T_n^{\text{IMR}}}{2} - \mathbf{y}(t_{n+\frac{1}{2}}). \tag{C.4}$$

Substituting in the Taylor expansions of $\mathbf{y}(t_n)$ and $\mathbf{y}(t_{n+1})$ about $t_{n+\frac{1}{2}}$ (from (3.19) and (3.20)) gives

$$\begin{aligned} \boldsymbol{\delta}_n &= \mathbf{y}(t_{n+\frac{1}{2}}) + \frac{\Delta_n^2}{8}\mathbf{y}''(t_{n+\frac{1}{2}}) - \mathbf{y}(t_{n+\frac{1}{2}}) - \frac{1}{2}T_n^{\text{IMR}} + \mathrm{O}\big(\Delta_n^4\big), \\ &= \frac{\Delta_n^2}{8}\mathbf{y}''(t_{n+\frac{1}{2}}) - \frac{1}{2}T_n^{\text{IMR}} + \mathrm{O}\big(\Delta_n^4\big). \end{aligned} \tag{C.5}$$

Now we substitute $\boldsymbol{\delta}_n$ from (C.5) into the Taylor series expansion of $\mathbf{f}$ from (C.2) to obtain

$$\mathbf{f}(t_{n+\frac{1}{2}}, \frac{\mathbf{y}(t_n) + \mathbf{y}_{n+1}}{2}) = \mathbf{y}'(t_{n+\frac{1}{2}}) + \frac{\Delta_n^2}{8}F_{n+\frac{1}{2}} \cdot \mathbf{y}''(t_{n+\frac{1}{2}}) - \frac{1}{2}F_{n+\frac{1}{2}} \cdot T_n^{\text{IMR}} + \mathrm{O}\big(\Delta_n^4\big). \tag{C.6}$$

and using (C.6) in (3.21) gives the local truncation error

$$\begin{aligned} (I + \frac{\Delta_n}{2}F_{n+\frac{1}{2}}) \cdot T_n^{\text{IMR}} &= \Delta_n\mathbf{y}'(t_{n+\frac{1}{2}}) + \frac{\Delta_n^3}{24}\mathbf{y}'''(t_{n+\frac{1}{2}}) - \Delta_n\mathbf{y}'(t_{n+\frac{1}{2}}) \\ &\quad - \frac{\Delta_n^3}{8}F_{n+\frac{1}{2}} \cdot \mathbf{y}''(t_{n+\frac{1}{2}}) + \mathrm{O}\big(\Delta_n^4\big), \\ &= \frac{\Delta_n^3}{24}\left[\mathbf{y}'''(t_{n+\frac{1}{2}}) - 3F_{n+\frac{1}{2}} \cdot \mathbf{y}''(t_{n+\frac{1}{2}})\right] + \mathrm{O}\big(\Delta_n^4\big). \end{aligned} \tag{C.7}$$

Using a geometric series representation we can show that, if all eigenvalues of $-\frac{\Delta_n}{2}F_{n+\frac{1}{2}}$ are such that $|\lambda| < 1$, then[2]

$$(I + \frac{\Delta_n}{2}F_{n+\frac{1}{2}})^{-1} = I - \frac{\Delta_n F_{n+\frac{1}{2}}}{2} + \mathrm{O}\big(\Delta_n^2\big), \tag{C.8}$$

and finally

$$T_n^{\text{IMR}} = \frac{\Delta_n^3}{24}\left[\mathbf{y}'''(t_{n+\frac{1}{2}}) - 3F_{n+\frac{1}{2}} \cdot \mathbf{y}''(t_{n+\frac{1}{2}})\right] \quad + \mathrm{O}\big(\Delta_n^4\big). \tag{C.9}$$

[2]If $F_{n+\frac{1}{2}}$ is not dependent on Δ_n then this will always be true for some sufficiently small Δ_n, the case where $F_{n+\frac{1}{2}}$ is a function of Δ_n is covered in Section 3.4.6.

Appendix D

Alternative magnetisation re-normalisation methods

In this this appendix we experimentally compare three methods for re-normalising the magnetisation:

1. Re-normalisation after every step, the simplest magnetisation re-normalisation method.
2. The self-correcting LLG method used by Nmag.
3. Tolerance-based re-normalisation used by magpar.

These three methods are described in Section 3.4.8. As our example problem we use the relaxing nano-sphere described in Section 7.4.

D.1 Implementation details

As in Section 7.4.2 we use the Landau-Lifshitz (LL) form of the LLG. The Newton-Raphson method is used for linearisation with the Jacobian calculated analytically and solved using a direct solver. The Newton-Raphson tolerance set to $\epsilon_{\mathrm{N}} = 10^{-8}$. We use BDF2 for these experiments because we need a time integrator with no geometric integration properties in order to properly test the re-normalisation methods (TR is equivalent to IMR for the undamped ODE LLG problem so it

has the same geometric integration properties in this case, see Section 7.4.3). The time step size is fixed at $\Delta_n = 0.1$.

Tolerance-based re-normalisation is implemented as follows: if the error in the magnetisation length, $\mathcal{E}_{|\mathbf{m}|} = |1 - |\mathbf{m}||$, is less than the tolerance, ϵ_{ml}, then nothing is done and the integration continues. On the other hand if $\mathcal{E}_{|\mathbf{m}|} > \epsilon_{\text{ml}}$ then we set $\mathbf{m}_i$ to $\mathbf{m}_i / |\mathbf{m}_i|$ for the values of the magnetisation at all times stored by the time integrator (for BDF2 this corresponds to $i = n+1, n, n-1$). Note that re-normalising the history values of TR is more complex because one of them is a derivative (at least in our implementation, see (4.57)), which would need to be recalculated using the newly re-normalised magnetisation values for consistency.

We use tolerance values of $\epsilon_{\text{ml}} = 0, 10^{-6}, 10^{-2}, 10^{200}$. Note that $\epsilon_{\text{ml}} \gg 1$ (*e.g.* $\epsilon_{\text{ml}} = 10^{200}$) corresponds to no re-normalisation, while $\epsilon_{\text{ml}} = 0$ corresponds to re-normalisation after every time step. The default value used in magpar[1] is $\epsilon_{\text{ml}} = 10^{-2}$.

The self correcting LLG is implemented by replacing the standard residual, $\mathbf{r}_{\text{ll}}$ given in (7.39), with the modified residual

$$\mathbf{r} = \mathbf{r}_{\text{ll}} - \beta \mathbf{m}\big(1 - |\mathbf{m}|^2\big), \tag{D.1}$$

where β is a parameter that can be tuned. The Jacobian of the modified residual is given by

$$\mathrm{J} = \mathrm{J}_{\text{ll}} + 2\beta(\mathbf{m} \otimes \mathbf{m}) - \beta\big(1 - |\mathbf{m}|^2\big)\mathrm{I}, \tag{D.2}$$

where J_{ll} is the Jacobian of $\mathbf{r}_{\text{ll}}$ and is given in (7.40).

We experiment with a range of parameter values: $\beta = 0, 0.1, 1, 10, 100, 1000$. Larger values were not used because the Newton-Raphson method often failed to converge within ten steps for $\beta \gtrsim 1000$.

Note that, due to implementation details, values are output *after* any re-normalisation for that step has been performed.

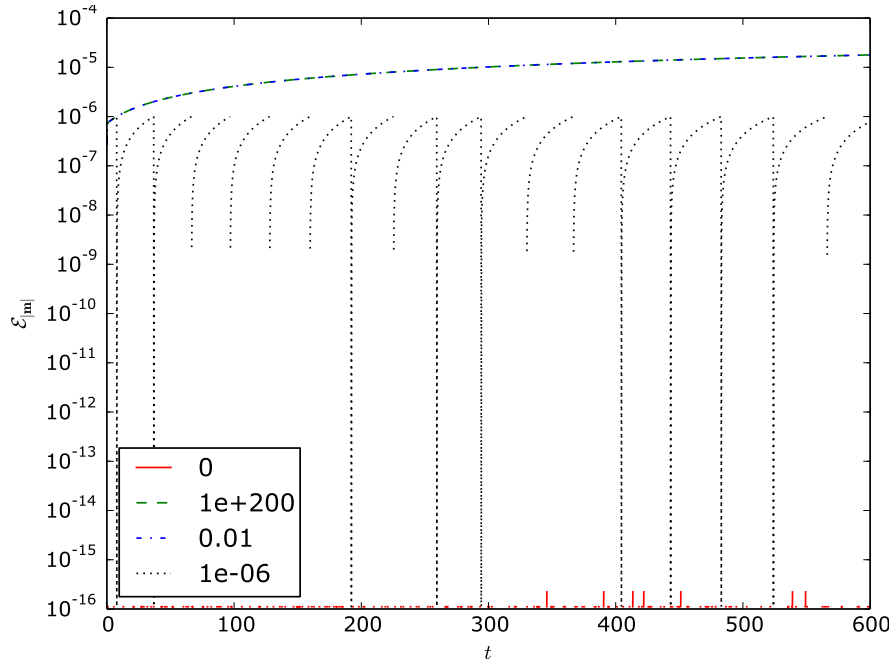

Figure D.1: Error in magnetisation length, $\mathcal{E}_{|\mathbf{m}|}$, over time for the relaxing nanosphere problem with $\alpha = 0$. Magnetisation length is enforced by tolerance-based renormalisation, the legend indicates the value of the tolerance ϵ_{ml}.

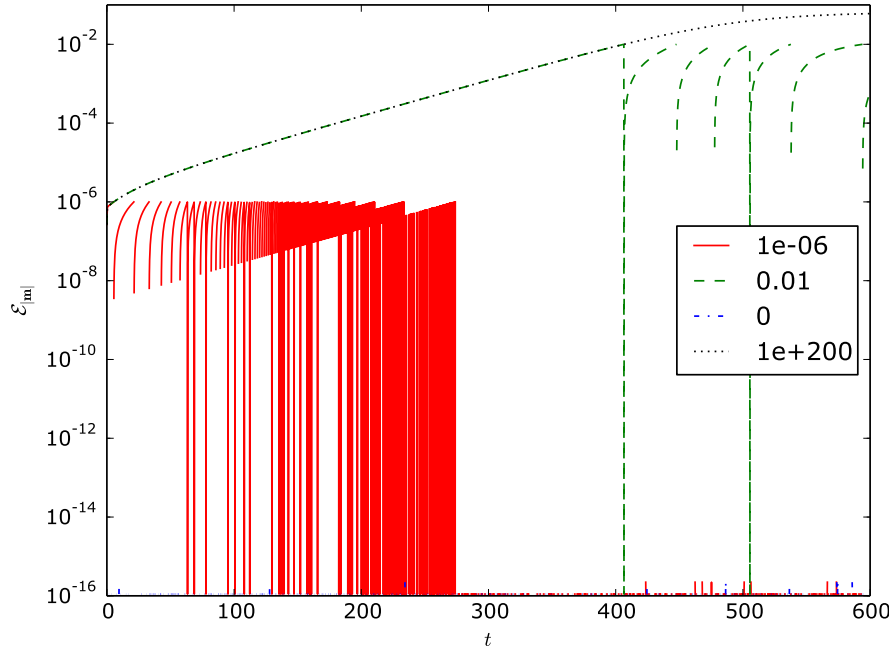

Figure D.2: Error in magnetisation length, $\mathcal{E}_{|\mathbf{m}|}$, over time for the relaxing nanosphere problem with $\alpha = 0.01$. Magnetisation length is enforced by tolerance-based renormalisation, the legend indicates the value of the tolerance ϵ_{ml}.

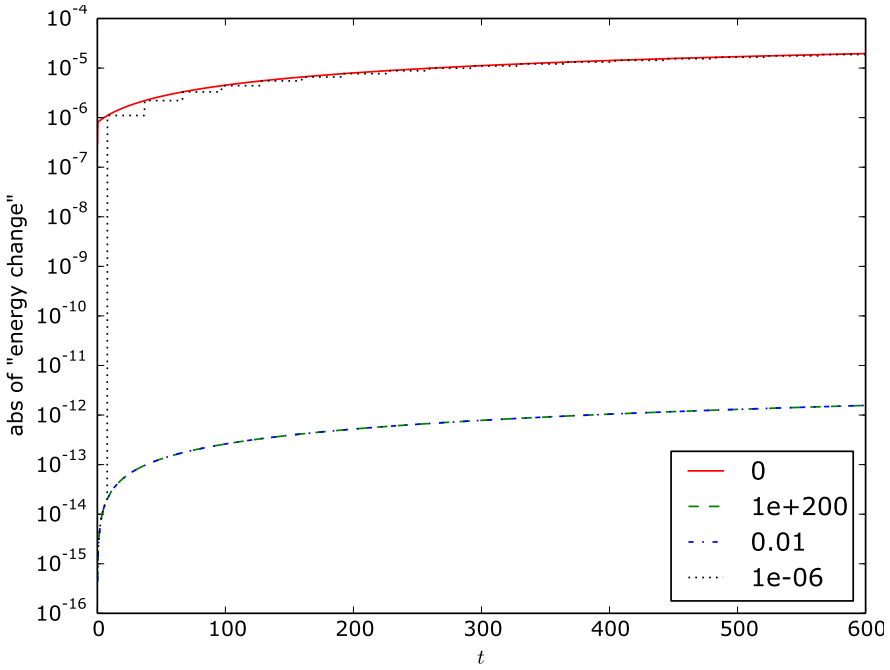

Figure D.3: Error in energy over time for the relaxing nano-sphere problem with $\alpha = 0$. Magnetisation length is enforced by tolerance-based renormalisation, the legend indicates the value of the tolerance ϵ_{ml}.

D.2 Results: tolerance-based renormalisation

In Figures D.1 and D.2 we show the error in the magnetisation length over time

$$\mathcal{E}_{|\mathbf{m}|} = ||\mathbf{m}| - 1| \tag{D.3}$$

for various values of ϵ_{ml} and $\alpha = 0.01, 0$ respectively. For the case with $\alpha = 0.01$ and $\epsilon_{\mathrm{ml}} = 10^{-6}$ re-normalisation is performed with increasing frequency until $t \gtrsim 280$, after which it is performed after almost every step. In both the damped and undamped cases, the use of the intermediate tolerance value, $\epsilon_{\mathrm{ml}} = 10^{-6}$, results in oscillations of the magnetisation length. For $\epsilon_{\mathrm{ml}} = 10^{-2}$ in the undamped case the error in the magnetisation length never reaches the tolerance and the behaviour is identical to the case of no re-normalisation. In the damped case the tolerance $\epsilon_{\mathrm{ml}} = 10^{-2}$ is reached and the behaviour is similar to that of $\epsilon_{\mathrm{ml}} = 10^{-6}$, except that the oscillations in the magnetisation length begin at a later time.

In Figure D.3 the errors in the energy (see Section 7.4.1) for the undamped case

[1]Based on the source code of version 0.9, the relevant function is `CheckIterationLL_Init`.

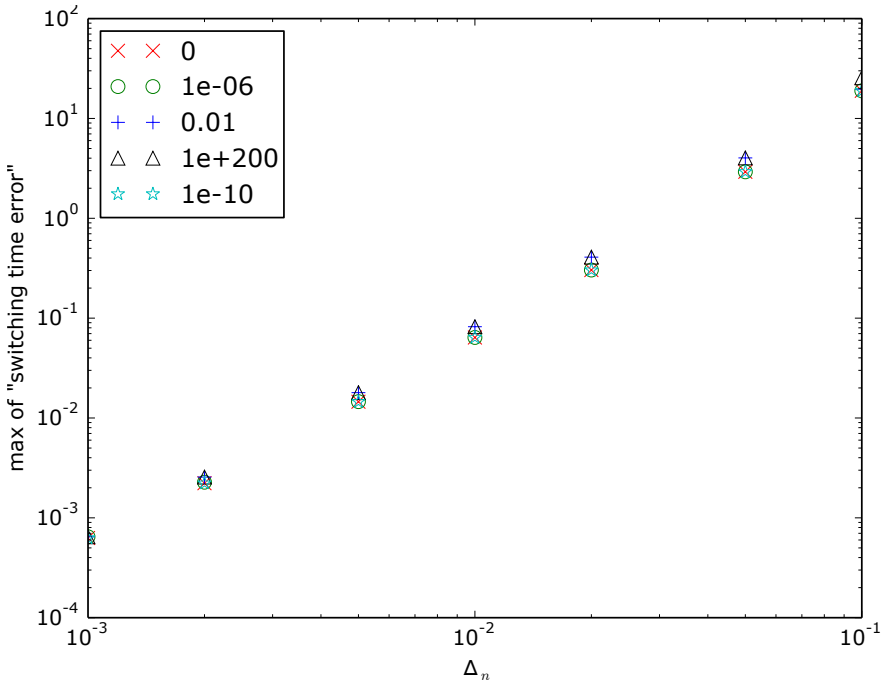

Figure D.4: Convergence of the maximum (over all steps) of the error in the switching time against step size for the relaxing nano-sphere problem with $\alpha = 0.01$. *Magnetisation length is enforced by tolerance-based renormalisation, the legend indicates the value of the tolerance* ϵ_{ml}.

are shown. As noted previously, $\epsilon_{\text{ml}} = 10^{-2}$ behaves as the non-renormalised case for this example. When $\epsilon_{\text{ml}} = 10^{-6}$ we see an error similar to that for the always re-normalised case, except with additional oscillations in the energy corresponding to times where a re-normalisation is carried out.

Finally in Figure D.4 we show the convergence of the method (in terms of the error in the switching time, (7.33)) as the step size is reduced for the damped case. The use of $\epsilon_{\text{ml}} = 10^{-2}$ or no re-normalisation ($\epsilon_{\text{ml}} = 10^{200}$) gives slightly worse errors than the tighter tolerances.

D.3 Results: the self-correcting LLG

In Figures D.5 and D.6 we show the errors in magnetisation length over time with various values of the parameter β for the damped and undamped problems respectively. As would be expected larger values of β reduce the error in both cases. Note that in the damped case (Figure D.6) with $\beta = 1000$ the curve stops

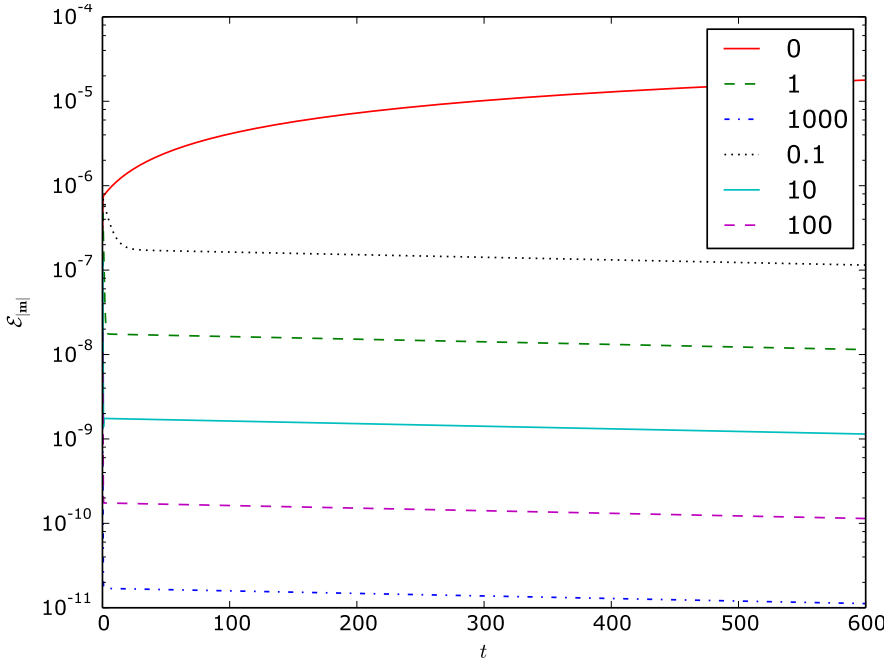

Figure D.5: Error in magnetisation length, $\mathcal{E}_{|\mathbf{m}|}$, over time for the relaxing nano-sphere problem with $\alpha = 0$. Magnetisation length is enforced by use of the self-correcting LLG. The legend indicates the values of β.

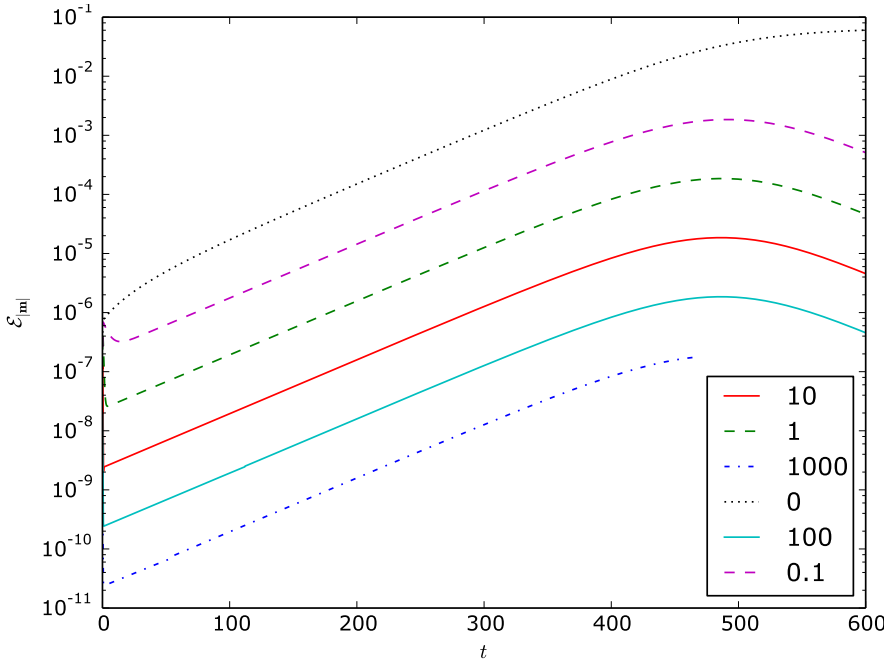

Figure D.6: Error in magnetisation length, $\mathcal{E}_{|\mathbf{m}|}$, over time for the relaxing nano-sphere problem with $\alpha = 0.01$. Magnetisation length is enforced by use of the self-correcting LLG, the legend indicates the values of β.

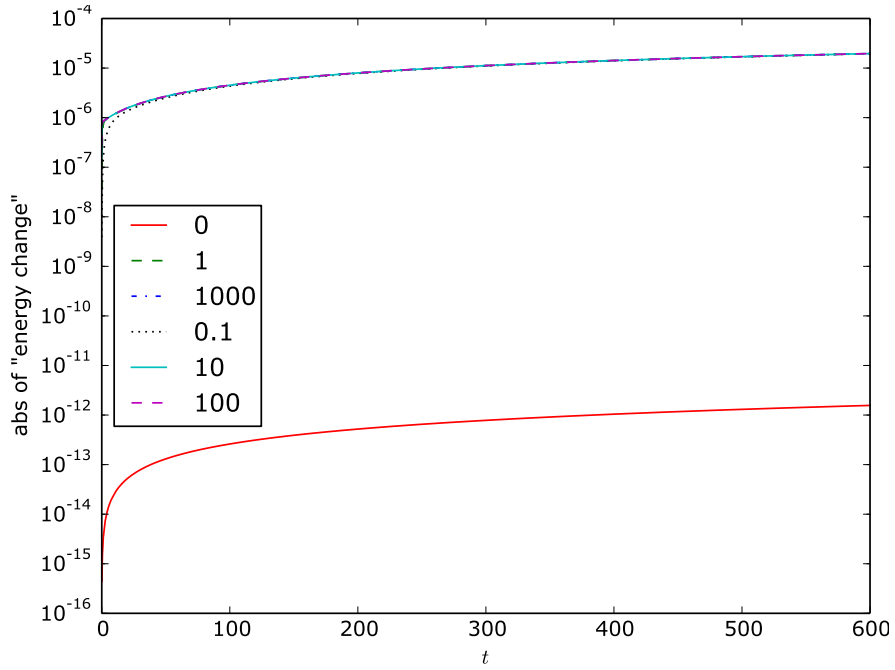

Figure D.7: Error in energy over time for the relaxing nano-sphere problem with $\alpha = 0$. Magnetisation length is enforced by use of the self-correcting LLG, the legend indicates the values of β.

at $t \sim 480$, this is due to non-convergence of the Newton-Raphson method within the ten iteration limit.

In Figure D.7 we show the error in the energy for the undamped problem for various values of the parameter β. We see that any $\beta > 0$ causes an error in the energy similar to that caused by the various re-normalisation approaches shown in Figure D.3.

In Figure D.8 we show the convergence of the maximum error in the switching time against the time step size Δ_n with two values of β. For comparison we also show the results when re-normalisation after every step is used instead of the self correcting term (*i.e.* $\epsilon_{\text{ml}} = 0$ and $\beta = 0$). We see that the resulting errors are essentially identical for all three cases.

In Table D.1 we show the number of Newton-Raphson iterations required for convergence for each value of β. For comparison we also show the result when using IMR with Newton tolerance $\epsilon_{\text{N}} = 10^{-12}$ (recall that a sharp linearisation tolerance is required for IMR to attain good geometric integration properties). We see that as β is increased the mean number of Newton iterations to converge also increases. In addition we see that the tighter tolerance required for geometric

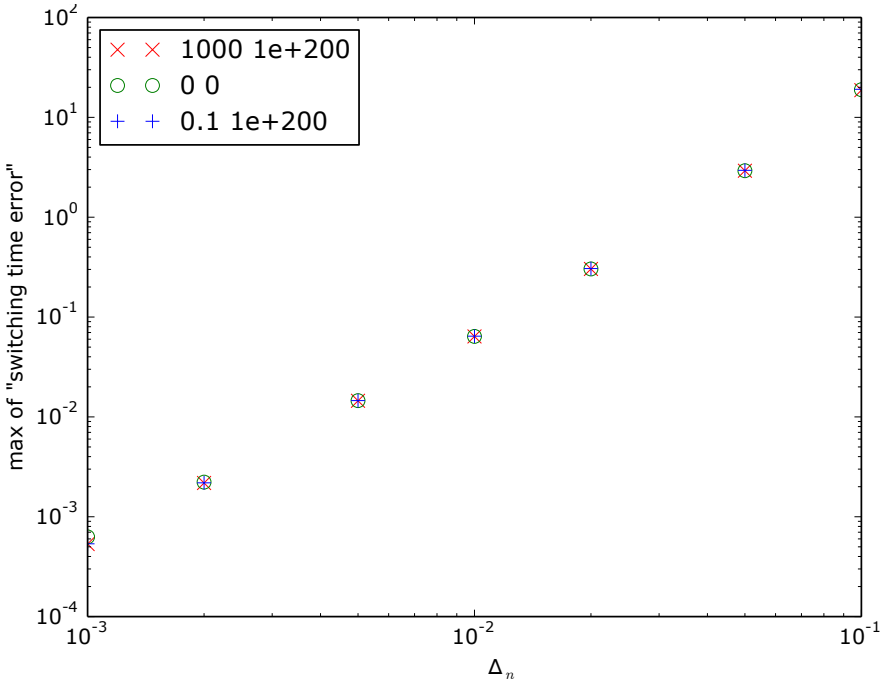

Figure D.8: Convergence of the maximum (over all steps) of the error in the switching time against step size for the relaxing nano-sphere problem with $\alpha = 0.01$. *Magnetisation length is enforced by use of the self-correcting LLG or re-normalisation after every step (i.e. tolerance based renormalisation with* $\epsilon_{\text{ml}} = 0$*). The legend indicates, respectively, the values of* β *and the tolerance* ϵ_{ml}.

β	Mean Newton iterations
0	2.0
0.1	2.0
1	2.46
10	2.68
100	3.13
1000	3.12*

(a) Solved with BDF2 and $\epsilon_{\text{N}} = 10^{-8}$. *Magnetisation length is enforced by use of the self-correcting LLG.*

β	Mean Newton iterations
0	2.0
0.1	-
1	-
10	-
100	-
1000	-

(b) Solved with IMR and $\epsilon_{\text{N}} = 10^{-12}$

Table D.1: Mean number of Newton iterations (over all time steps) for convergence for the relaxing nano-sphere problem with $\alpha = 0.01$. *The time integration methods used are indicated in the sub-captions. Values of* $\beta \neq 0$ *are not considered when using IMR because no self-correcting term is needed to conserve* $|\mathbf{m}|$. **With* $\beta = 1000$ *the Newton-Raphson method fails to converge at* $t \sim 480$.

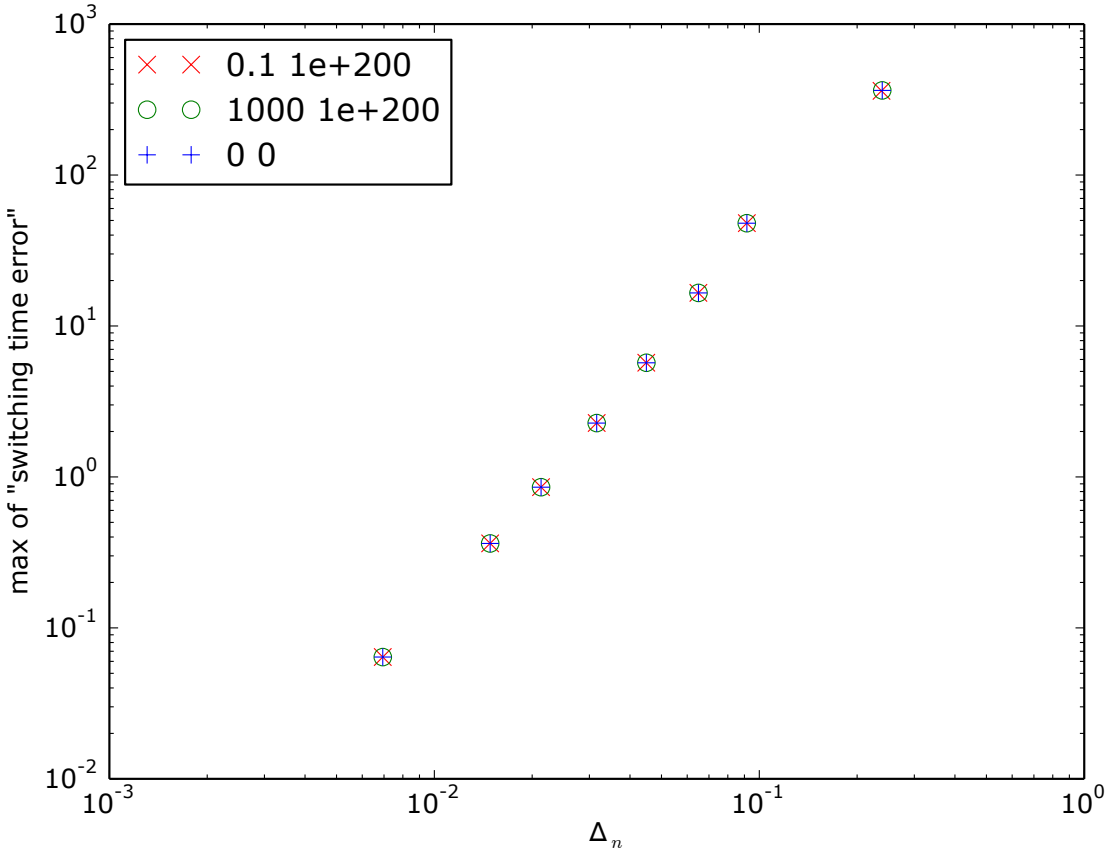

Figure D.9: Convergence of the maximum (over all steps) of the error in the switching time against step size for the relaxing nano-sphere problem with $\alpha = 0.01$ *solved using adaptive BDF2. Magnetisation length is enforced by use of the self-correcting LLG or re-normalisation after every step (i.e. tolerance based renormalisation with* $\epsilon_{\text{ml}} = 0$*). The legend indicates, respectively, the values of* β *and the tolerance* ϵ_{ml}*.*

integration with IMR has no effect on the number of iterations required.

Since $\beta \geq 100$ is required to control $\mathcal{E}_{|\mathbf{m}|}$ to even the fairly loose tolerance of 10^{-6}, the use of the self correcting LLG requires, on average, at least one additional Newton-Raphson iteration per time step. As the majority of the computation time for a time step is spent within the Newton-Raphson method, this increases the cost of each step by approximately 50%. Since the use of the self correcting LLG does not provide any improvement in the overall accuracy or energy error over re-normalising after every step (so the same Δ_n is required to obtain the same accuracy), this increases the overall cost of the method by the same factor.

Finally we study the effect of the use of the self-correcting LLG on adaptive time integration. Since the self correcting term is part of the differential equation

(as opposed to standard renormalisation, which is carried out separately) we might expect adaptive time integration methods to be able to better predict the error, and choose more appropriate step sizes. In Figure D.9 we show the convergence behaviour when using adaptive BDF2 (as described in Section 7.3.1) with tolerances of $\epsilon_\Delta = 10^{-4}, 3\times10^{-5}, 10^{-5}, 3\times10^{-6}, 10^{-6}, 3\times10^{-7}, 10^{-7}, 10^{-8}$ with various values of β. For comparison we also show the convergence behaviour when the magnetisation is re-normalised after every step. If the use of the self-correcting LLG is beneficial we would expect to see lower errors for the same average step size, but the plots are indistinguishable.

D.4 Conclusions and future work

By design, the use of tolerance-based renormalisation allows larger errors to appear in $|\mathbf{m}|$ than re-normalisation after every step. It also introduces spurious oscillations in $|\mathbf{m}|$ and in the error in the energy for the undamped case. It has a slight negative effect on convergence when compared to re-normalisation after every step, and no effect on the computational cost of the re-normalisation.

The self correcting LLG is worse at controlling errors in $|\mathbf{m}|$ than re-normalisation after each step, and gives very similar errors in the energy. It gives essentially identical convergence properties as re-normalisation after every step. Finally, it requires additional step(s) of the Newton-Raphson method convergence (typically one more step is needed, increasing the computation time for the simulation by a factor of $\sim 50\%$).

Hence there appears to be no obvious reason to prefer either tolerance-based renormalisation or the self correcting LLG over re-normalisation after every step for ODE problems. Since the methods used here match those used for PDE problems, except for the obvious lack of any spatial discretisation, this conclusion is likely to extend to PDE problems.

Ideally we would also show wall-clock timing results for CVODE or VODE (two widely used ODE integration packages) when using these re-normalisation methods, and compare these results with adaptive IMR. However the VODE implementation used previously in this thesis uses a derivative function, $\mathbf{f}$, written in python. This has a huge performance penalty: for the relaxing nano-sphere problem VODE is 500x slower than oomph-lib, so any timing results would be

meaningless. Hopefully in the future adaptive IMR will be implemented in the CVODE/VODE framework and a fair comparison of the wall-clock timings can be made.

Bibliography

[1] A. Aharoni. *Introduction to the Theory of Ferromagnetism*. 2nd ed. Oxford: Oxford University Press, 1996.

[2] G. Albuquerque, J. Miltat and A. Thiaville. "Self-consistency based control scheme for magnetization dynamics". *Journal of Applied Physics* 89.11 (2001), p. 6719. DOI: 10.1063/1.1355322.

[3] F. Alouges. "Computation of the demagnetizing potential in micromagnetics using a coupled finite and infinite elements method". *ESAIM: Control, Optimisation and Calculus of Variations* 6.October (Aug. 2002), pp. 629–647. DOI: 10.1051/cocv:2001126.

[4] C. Andreas, S. Gliga and R. Hertel. "Numerical micromagnetism of strong inhomogeneities". *Journal of Magnetism and Magnetic Materials* 362 (Aug. 2014), pp. 7–13. DOI: 10.1016/j.jmmm.2014.02.097.

[5] D. Apalkov et al. "Spin-transfer torque magnetic random access memory (STT-MRAM)". *ACM Journal on Emerging Technologies in Computing Systems* 9.2 (May 2013), pp. 1–35. DOI: 10.1145/2463585.2463589.

[6] K. E. Atkinson. *User's guide to a boundary element package for solving integral equations on piecewise smooth surfaces*. 1994. URL: ftp://ftp.math.uiowa.edu/pub/atkinson/bie.package/.

[7] K. E. Atkinson, W. Han and D. E. Stewart. *Numerical Solution of Ordinary Differential Equations*. 1st ed. Hoboken: Wiley, 2009.

[8] L. Banas. "On dynamical Micromagnetism with Magnetostriction". PhD thesis. Ghent University, 2005.

[9] S. Bartels and A. Prohl. "Constraint preserving implicit finite element discretization of harmonic map flow into spheres". *Mathematics of Computation* 151.3712 (Feb. 2007), p. 859. DOI: 10.1126/science.151.3712.859-a.

[10] S. Bartels and A. Prohl. "Convergence of an Implicit Finite Element Method for the Landau–Lifshitz–Gilbert Equation". *SIAM Journal on Numerical Analysis* 44.4 (2006), p. 1405. DOI: 10.1137/050631070.

[11] R. Beatson and L. Greengard. "A short course on fast multipole methods". In: *Wavelets, Multilevel Methods and Elliptic PDEs*. Oxford University Press, 1997, pp. 1–37. URL: http://math.nyu.edu/faculty/greengar/shortcourse_fmm.pdf.

[12] M. Bonnet and M. Guiggiani. "Direct evaluation of double singular integrals and new free terms in 2D (symmetric) Galerkin BEM". *Computer Methods in Applied Mechanics and Engineering* 192.22-24 (June 2003), pp. 2565–2596. DOI: 10.1016/S0045-7825(03)00286-X.

[13] S. Börm and L. Grasedyck. *HLib*. URL: http://www.hlib.org/ (visited on 16/09/2014).

[14] S. Borm, L. Grasedyck and W. Hackbusch. *Hierarchical Matrices*. June. 2003.

[15] O. Bottauscio, M. Chiampi and A. Manzin. "A Finite Element Procedure for Dynamic Micromagnetic Computations". *IEEE Transactions on Magnetics* 44.11 (Nov. 2008), pp. 3149–3152. DOI: 10.1109/TMAG.2008.2001666.

[16] O. Bottauscio and A. Manzin. "Efficiency of the Geometric Integration of Landau–Lifshitz–Gilbert Equation Based on Cayley Transform". *IEEE Transactions on Magnetics* 47.5 (May 2011), pp. 1154–1157. DOI: 10.1109/TMAG.2010.2095831.

[17] W. Briggs, V. E. Henson and S. F. McCormick. *A Multigrid Tutorial*. 2nd ed. Society for Industrial and Applied Mathematics, 2000.

[18] W. F. Brown. *Micromagnetics*. Wiley, 1963.

[19] R. Burden and J. Faires. *Numerical Analysis*. 9th ed. Brooks/Cole, 2001.

[20] R. Chang et al. "FastMag: Fast micromagnetic simulator for complex magnetic structures". *Journal of Applied Physics* 109.7 (2011), pages. DOI: 10.1063/1.3563081.

[21] R. W. Chantrell et al. *Micromagnetics : Finite Element Approach*. 2001.

[22] I. Cimrák. "A Survey on the Numerics and Computations for the Landau-Lifshitz Equation of Micromagnetism". *Archives of Computational Methods in Engineering* 15.3 (May 2008), pp. 277–309. DOI: 10.1007/s11831-008-9021-2.

[23] J. M. D. Coey. *Magnetism and Magnetic Materials*. 1st ed. Cambridge: Cambridge University Press, 2010.

[24] M. d'Aquino, C. Serpico and G. Miano. "Geometrical integration of Landau-Lifshitz-Gilbert equation based on the mid-point rule". *Journal of Computational Physics* 209.2 (Nov. 2005), pp. 730–753. DOI: 10.1016/j.jcp.2005.04.001.

[25] M. d'Aquino et al. "Midpoint numerical technique for stochastic Landau-Lifshitz-Gilbert dynamics". *Journal of Applied Physics* 99.8 (2006), 08B905. DOI: 10.1063/1.2169472.

[26] M. J. Donahue. *OOMMF website*. URL: http://math.nist.gov/oommf/ (visited on 06/03/2012).

[27] D. A. Dunavant. "High degree efficient symmetrical Gaussian quadrature rules for the triangle". *International Journal for Numerical Methods in Engineering* 21.6 (June 1985), pp. 1129–1148. DOI: 10.1002/nme.1620210612.

[28] R. F. L. Evans et al. "Atomistic spin model simulations of magnetic nanomaterials". *Journal of Physics: Condensed Matter* 26.10 (Mar. 2014), p. 103202. DOI: 10.1088/0953-8984/26/10/103202.

[29] R. F. L. Evans et al. "Stochastic form of the Landau-Lifshitz-Bloch equation". *Physical Review B* 85.1 (Jan. 2012), p. 014433. DOI: 10.1103/PhysRevB.85.014433.

[30] H. Fanghor. Personal communication. 2014.

[31] H. Fanghor and M. Donahue. Personal communication. 2014.

[32] H. Fangohr, T. Fischbacher and M. Franchin. *Nmag home*. URL: http://nmag.soton.ac.uk/nmag/ (visited on 22/03/2012).

[33] H. Fangohr et al. *NMAG User Manual*. Southampton, 2012. URL: `http://nmag.soton.ac.uk/nmag/0.2/manual/html/manual`.

[34] J. Fidler and T. Schrefl. "Micromagnetic modelling - the current state of the art". *Journal of Physics D: Applied Physics* 33.15 (Aug. 2000), R135–R156. DOI: `10.1088/0022-3727/33/15/201`.

[35] T. Fischbacher and H. Fangohr. "Continuum multi-physics modeling with scripting languages : the Nsim simulation compiler prototype for classical field theory". *arXiv* (2009), pp. 1–50. arXiv: `arXiv:0907.1587v1`.

[36] T. Fischbacher et al. "A Systematic Approach to Multiphysics Extensions of Finite-Element-Based Micromagnetic Simulations: Nmag". *IEEE Transactions on Magnetics* 43.6 (June 2007), pp. 2896–2898. DOI: `10.1109/TMAG.2007.893843`.

[37] H. Forster et al. "Fast boundary methods for magnetostatic interactions in micromagnetics". *IEEE Transactions on Magnetics* 39.5 (Sept. 2003), pp. 2513–2515. DOI: `10.1109/TMAG.2003.816458`.

[38] D. R. Fredkin and T. R. Koehler. "Hybrid Method for Computing Demagnetizing Fields". *IEEE Transactions on Magnetics* 26.2 (Nov. 1990), pp. 415–417. DOI: `10.1109/TUFFC.2011.2101`.

[39] H. Fukushima et al. "Magnetization reversal below the Stoner–Wohlfarth field". *Journal of Magnetism and Magnetic Materials* 290-291 (Apr. 2005), pp. 526–529. DOI: `10.1016/j.jmmm.2004.11.518`.

[40] A. Fuwa, T. Ishiwata and M. Tsutsumi. "Finite difference scheme for the Landau–Lifshitz equation". *Proceedings of the Czech–Japanese Seminar in Applied Mathematics* (2006), pp. 107–113.

[41] C. Garcia-Cervera and A. Roma. "Adaptive Mesh Refinement for Micromagnetics Simulations". *IEEE Transactions on Magnetics* 42.6 (June 2006), pp. 1648–1654. DOI: `10.1109/TMAG.2006.872199`.

[42] T. Gilbert. "A phenomenological theory of damping in ferromagnetic materials". *IEEE Transactions on Magnetics* 40.6 (2004), pp. 3443–3449.

[43] B. Gladman, M. Duncan and J. Candy. "Symplectic integrators for long-term integrations in celestial mechanics". *Celestial Mechanics and Dynamical ...* (1991), pp. 221–240.

[44] P. M. Gresho and R. L. Sani. *Incompressible flow and the finite element method: Advection-diffusion and isothermal laminar flow*. 1st ed. Chichester: Wiley, 1998.

[45] P. M. Gresho, D. F. Griffiths and D. J. Silvester. "Adaptive time-stepping for incompressible flow part I: Scalar advection-diffusion". *SIAM Journal on Scientific Computing* 30.4 (2008), pp. 2018–2054.

[46] E. Hairer, S. P. Norsett and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. 2nd ed. Berlin: Springer-Verlag, 1991.

[47] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*. 2nd ed. Berlin: Springer, 1996.

[48] S. C. Harvey, R. K.-Z. Tan and T. E. Cheatham III. "The Flying Ice Cube: Velocity Rescaling in Molecular Dynamics Leads to Violation of Energy Equipartition". *Journal of Computational Chemistry* 19.7 (1998), pp. 726–740.

[49] A. Hatfield. *Everspin debuts first Spin-Torque MRAM for high performance storage systems*. Tech. rep. Everspin Technologies, 2012.

[50] V. E. Henson and U. M. Yang. "BoomerAMG : A parallel algebraic multigrid solver". *Applied Numerical Mathematics* 41 (2002), pp. 155–177.

[51] A. C. Hindmarsh and R. Serban. *User Documentation for cvode v2.7.0*. 2012. URL: `http://computation.llnl.gov/casc/sundials/documentation/documentation.html` (visited on 15/04/2014).

[52] A. Hubert and R. Schafer. *Magnetic Domains*. 1st ed. Springer-Verlag, 1998.

[53] *HYPRE – High performance preconditioning library*. URL: `http://www.llnl.gov/CASC/hypre/software.html`.

[54] A. Iserles. *A First Course in the Numerical Analysis of Differential Equations*. 2nd ed. Cambridge: Cambridge University Press, 2009.

[55] D. Jeong and J. Kim. "A Crank–Nicolson scheme for the Landau–Lifshitz equation without damping". *Journal of Computational and Applied Mathematics* 234.2 (May 2010), pp. 613–623. DOI: `10.1016/j.cam.2010.01.002`.

[56] D. Jeong and J. Kim. "An accurate and robust numerical method for micromagnetics simulations". *Current Applied Physics* 14.3 (Mar. 2014), pp. 476–483. DOI: 10.1016/j.cap.2013.12.028.

[57] E. Jones, T. Oliphant, P. Peterson et al. *SciPy: Open source scientific tools for Python*. 2001–. URL: http://www.scipy.org/.

[58] M. Jones and J. J. Miles. "An accurate and efficient 3-D micromagnetic simulation of metal evaporated tape". *Journal of Magnetism and Magnetic Materials* 171 (1997), pp. 190–208.

[59] D. Kincaid and W. Cheney. *Numerical Analysis: Mathematics of Scientific Computing*. Ed. by P. Sally. 3rd ed. Brooks/Cole, 2002.

[60] A. Knittel. "Micromagnetic simulations of three dimensional core-shell nanostructures". PhD thesis. University of Southampton, 2011.

[61] A. Knittel et al. "Compression of boundary element matrix in micromagnetic simulations". *Journal of Applied Physics* 105.7 (2009), p. 07D542. DOI: 10.1063/1.3072032.

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

[63] H. Kronmüller and R. Fischer. "Micromagnetism and the microstructure in nanocrystalline materials". *Journal of Magnetism and Magnetic Materials* (1997).

[64] M. Lakshmanan, T. Ruijgrok and C. Thompson. "On the dynamics of a continuum spin system". *Physica A: Statistical Mechanics and its Applications* (1976), pp. 577–590. DOI: 10.1037/e497462006-003.

[65] L. Landau and E. Lifshitz. "On the theory of the dispersion of magnetic permeability in ferromagnetic bodies". *Phys. Z. Sowjetunion* 8 (1935), pp. 153–169.

[66] D. Lewis and N. Nigam. "Geometric integration on spheres and some interesting applications". *Journal of Computational and Applied Mathematics* 34.3 (Feb. 2003), pp. 3842–170. DOI: 10.1016/S0377-0427(02)00743-4.

[67] X. S. Li. "An overview of SuperLU: Algorithms, implementation, and user interface". *ACM Transactions on Mathematical Software* 31.3 (Sept. 2005), pp. 302–325.

[68] D. Lindholm. "Three-dimensional magnetostatic fields from point-matched integral equations with linearly varying scalar sources". *IEEE Transactions on Magnetics* 20.5 (Sept. 1984), pp. 2025–2032. DOI: 10.1109/TMAG.1984.1063254.

[69] A. Malidi, S. Dufour and D. N'dri. "A study of time integration schemes for the numerical modelling of free surface flows". *International Journal for Numerical Methods in Fluids* 48.10 (Aug. 2005), pp. 1123–1147. DOI: 10.1002/fld.981.

[70] J. C. Mallinson. "Damped gyromagnetic switching". *IEEE Transactions on Magnetics* 36.4 (2000), pp. 1976–1981.

[71] J. C. Mallinson. "On damped gyromagnetic precession". *IEEE Transactions on Magnetics* 23.4 (1987), pp. 2003–2004.

[72] MathWorks. *ode45*. 2014. URL: http://www.mathworks.co.uk/help/matlab/ref/ode45.html (visited on 03/07/2014).

[73] B. McMichael. *μMAG – Micromagnetic Modeling Activity Group*. URL: http://www.ctcms.nist.gov/~rdm/mumag.org.html (visited on 22/03/2012).

[74] J. H. Mentink et al. "Stable and fast semi-implicit integration of the stochastic Landau-Lifshitz equation." *Journal of Physics: Condensed Matter* 22.17 (May 2010), p. 176001. DOI: 10.1088/0953-8984/22/17/176001.

[75] Y. Nakatani, Y. Uesaka and N. Hayashi. "Direct solution of the Landau-Lifshiftz-Gilbert equation for micromagnetics". *Japanese Journal of Applied Physics* 28.12 (1989), pp. 2485–2507.

[76] O. Nevanlinna and A. H. Sipilä. "A Nonexistence Theorem for Explicit A-Stable Methods". *Mathematics of computation* 28.128 (1974), pp. 1053–1055.

[77] D. Porter and M. J. Donahue. "Magnetization Normalization Methods for Landau-Lifshitz-Gilbert". In: *MMM presentation*. 2005.

[78] M. Puso and J. Chen. "Meshfree and finite element nodal integration methods". *International Journal for Numerical Methods in Engineering* September 2007 (2008), pp. 416–446. DOI: 10.1002/nme.

[79] G. Rado and J. Weertman. "Spin-wave resonance in a ferromagnetic metal". *Journal of Physics and Chemistry of Solids* 11 (1959), pp. 315–333. DOI: 10.1016/0022-3697(59)90233-1.

[80] H. T. Rathod, B. Venkatesudu and K. V. Nagaraja. "Gauss legendre quadrature formulas over a tetrahedron". *Numerical Methods for Partial Differential Equations* 22.1 (Jan. 2006), pp. 197–219. DOI: 10.1002/num.20095.

[81] Y. Saad. "A Flexible Inner-Outer Preconditioned GMRES Algorithm". *SIAM Journal on Scientific Computing* 14.2 (Mar. 1993), pp. 461–469. DOI: 10.1137/0914028.

[82] Y. Saad. *Iterative methods for sparse linear systems*. 2nd ed. SIAM, 2000.

[83] W. Scholz. *MagparWiki*. URL: http://www.magpar.net/ (visited on 22/03/2012).

[84] W. Scholz. "Scalable parallel micromagnetic solvers for magnetic nanostructures". *Computational Materials Science* 28.2 (Oct. 2003), pp. 366–383. DOI: 10.1016/S0927-0256(03)00119-8.

[85] T. Schrefl. "Finite elements in numerical micromagnetics". *Journal of Magnetism and Magnetic Materials* 207.1-3 (Dec. 1999), pp. 45–65. DOI: 10.1016/S0304-8853(99)00532-6.

[86] T. Schrefl et al. "A Higher Order FEM-BEM Method for the Calculation of Domain Processes in Magnetic Nano-Elements". *IEEE Transactions on Magnetics* 33.5 (1997), pp. 4182–4184.

[87] C. Serpico, I. D. Mayergoyz and G. Bertotti. "Numerical technique for integration of the Landau–Lifshitz equation". *Journal of Applied Physics* 89.11 (2001), p. 6991. DOI: 10.1063/1.1358818.

[88] D. Shepherd. *generate-bdf-method.py*. 2014. URL: https://github.com/davidshepherd7/generate-bdf-method.

[89] D. Shepherd et al. "Discretisation induced stiffness in micromagnetic simulations". *IEEE Transactions on Magnetics* (2014).

[90] H. Si. *Tetgen: A Quality Tetrahedral Mesh Generator and a 3D Delaunay Triangulator*. 2013. URL: http://wias-berlin.de/software/tetgen.

[91] D. Silvester, H. Elman and A. Wathen. *Finite Elements and Fast Iterative Solvers*. 1st ed. Oxford University Press, 2006.

[92] J. Slonczewski. "Current-driven excitation of magnetic multilayers". *Journal of Magnetism and Magnetic Materials* 159.1-2 (June 1996), pp. L1–L7. DOI: 10.1016/0304-8853(96)00062-5.

[93] A. W. Spargo, P. H. W. Ridley and G. W. Roberts. "Geometric integration of the Gilbert equation". *Journal of Applied Physics* 93.10 (2003), p. 6805. DOI: 10.1063/1.1557274.

[94] W. J. Sternberg and T. L. Smith. *Theory of Potential and Spherical Harmonics*. 2nd ed. Oxford University Press, 1946.

[95] D. Suess et al. "Time resolved micromagnetics using a preconditioned time integration method". *Journal of Magnetism and Magnetic Materials* 248.2 (July 2002), pp. 298–311. DOI: 10.1016/S0304-8853(02)00341-4.

[96] *SymPy: Python library for symbolic mathematics*. 2014.

[97] H. Szambolics et al. "A constrained finite element formulation for the Landau–Lifshitz–Gilbert equations". *Computational Materials Science* 44.2 (Dec. 2008), pp. 253–258. DOI: 10.1016/j.commatsci.2008.03.019.

[98] L. Trefethen. "Is Gauss quadrature better than Clenshaw-Curtis?" *SIAM Review* 50.1 (2008), pp. 67–87.

[99] A. Vansteenkiste and B. Van de Wiele. "MuMax: A new high-performance micromagnetic simulation tool". *Journal of Magnetism and Magnetic Materials* 323.21 (Nov. 2011), pp. 2585–2591. DOI: 10.1016/j.jmmm.2011.05.037.

[100] B. V. D. Wiele. "Extending the Landau-Lifshitz-Gilbert Approach from Nanoscale to Microscale". PhD thesis. Ghent University, 2010.

[101] L. C. Wrobel. *The Boundary Element Method*. 1st ed. Chichester: Wiley, 2002.

[102] B. Yang. "Numerical Studies of Dynamical Micromagnetics". PhD thesis. University of California, 1997.

[103] O. C. Zeinkiewicz and R. L. Taylor. *The Finite Element Method*. 4th ed. London: McGraw-Hill, 1967.