Computationally Efficient Steady–State Simulation Algorithms for Finite–Element Models of Electric Machines

by

Jason Pries

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering: Systems)
in the University of Michigan
2015

Doctoral Committee:

Associate Professor Heath Hofmann, Chair
Professor Ian Hiskens
Professor Eric Michielssen
Professor Jeffrey L. Stein

There is no place in this new kind of physics both for the field and matter, for the field is the only reality [1].

–Albert Einstein

We are emphatically in sympathy with Einstein's dictum that measurable reality is the only reality, and feel that the word "real" has no applicability to results obtained by approximate solution of an idealized mathematical model [2].

–M.V.K. Chari and P. Silvester

Table of Contents

Chapter

List of Figures

Figure

List of Tables

Table

List of Algorithms

Algorithm

Del in Cartesian Coordinates

3 Dimensions			
gradient	$\nabla\phi$	$=$	$\frac{\partial\phi}{\partial x}\hat{x} + \frac{\partial\phi}{\partial y}\hat{y} + \frac{\partial\phi}{\partial z}\hat{z}$
divergence	$\nabla\cdot\mathbf{A}$	$=$	$\frac{\partial A_x}{\partial x} + \frac{\partial A_y}{\partial y} + \frac{\partial A_z}{\partial x}$
curl	$\nabla\times\mathbf{A}$	$=$	$\left(\frac{A_z}{\partial y} - \frac{A_y}{\partial z}\right)\hat{x} + \left(\frac{A_x}{\partial z} - \frac{A_z}{\partial x}\right)\hat{y} + \left(\frac{A_y}{\partial x} - \frac{A_x}{\partial y}\right)\hat{z}$
scalar laplacian	$\nabla^2\phi$	$=$	$\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2}$
vector laplacian	$\nabla^2\mathbf{A}$	$=$	$\left(\nabla^2 A_x\right)\hat{x} + \left(\nabla^2 A_y\right)\hat{y} + \left(\nabla^2 A_z\right)\hat{z}$
2 Dimensions			
primal curl	$\nabla\times A$	$=$	$\frac{\partial A}{\partial y}\hat{x} - \frac{\partial A}{\partial x}\hat{y}$
dual curl	$\nabla'\times\mathbf{B}$	$=$	$-\frac{\partial B_x}{\partial y} + \frac{\partial B_y}{\partial x}$
scalar laplacian	$\nabla^2 A$	$=$	$\frac{\partial^2 A}{\partial x^2} + \frac{\partial^2 A}{\partial y^2}$
Relationships Among Operators			
vector laplacian	$\nabla^2\mathbf{A}$	$=$	$\nabla\left(\nabla\cdot\mathbf{A}\right) - \nabla\times\nabla\times\mathbf{A}$
scalar laplacian (3D)	$\nabla^2\phi$	$=$	$\nabla\cdot\left(\nabla\phi\right)$
scalar laplacian (2D)	$\nabla^2 A$	$=$	$-\nabla'\times\nabla\times A$

Chapter 1

Introduction

This thesis concerns the development of time– and frequency–domain steady–state simulation tools for electric machine design and analysis for which the underlying spatial discretization method is based on the finite–element method. Our approach to algorithm development is motivated by the desire to keep the spatial and temporal discretizations orthogonal. Our model equation is expressed as a nonlinear time-varying differential algebraic equation with no specific reference to the underlying spatial discretization. Only the general characteristics of size and sparsity endowed to the matrices by the finite-element process must be kept in mind. In this sense, the algorithms developed here may be more generally applicable than our particular considerations suggest. Conversely, our focus on this particular application domain allows us to examine some very practical issues.

The motivation for this work is described in Section 1.1 and focuses on two themes. One is improved designs: better design tools allow one to produce better–designed machines and have more confidence that the design will perform as expected. The other theme is reduced costs. Obviously, one would consider the lower cost design of two machines with equivalent performance as the better option. The nominal machine cost is not the only cost consideration, however. An easily overlooked fact is the cost in human capital of the design process, which can be reduced by a more computationally efficient design tool.

A general outline of this thesis is presented in Section 1.2. Along the way, we provide some additional details of the various modeling tool choices that an electric machine designer is confronted with on a day–to–day basis. In doing this, we hope to shine a light in the gap between the two methods of analysis typically available; static (non–dynamic) analysis on the one hand, and transient (dynamic) analysis on the other. Our work developing steady–state analysis tools represents a tradeoff between the speed of the former method and the detail of the later. The particular contributions of this thesis are detailed in Section 1.4.

1.1 Motivation

The trend toward vehicle electrification is placing increasingly stringent demands on electric machines in terms of performance, cost, and reliability. The complex interplay between components of an electrified powertrain (internal combustion engine, gear ratio, cooling system, batteries) means that the electric machine cannot be designed in isolation if one is to realize system–level optimal performance. Iterating over powertrain designs requires a commensurate increase in the number of candidate electric machine designs considered for production. In all likelihood, the majority of these candidates are simulated rather than built.

The raw number of simulations that must be run puts increasing demands on design tools in terms of their computational efficiency. Faster simulation times result in faster turnaround times, lower design costs, and the potential for better designs through a more thorough search of the design space. There is a feedback loop between the design of the electric machine and the rest of the powertrain wherein the accuracy of the electric machine characterization affects the evolution of the entire powertrain design.

The importance of this feedback loop is more pronounced in electric vehicle applications due to volume, mass, acoustic, and cost constraints (some fundamental, some consumer driven). The simple open-loop design process of performing the electrical/magnetic design and verifying that the thermal/mechanical performance meet some threshold requirements is somewhat unsatisfactory. However, most simulation tools still reflect the requirements placed on them by the open-loop design process. The development of simulation tools that are oriented towards the modern closed-loop powertrain design process can benefit electric vehicle manufactures in both obvious and subtle ways.

Cost is a preeminent concern among automobile manufactures. Reducing the cost of producing an electric machine is one obvious way to reduce overall vehicle costs. Reducing material costs is one way to effect this change. The high cost and price volatility of rare earth permanent magnets is often cited as the prime motivation behind reducing total permanent magnet content or moving away from rare earth permanent magnet machines altogether.[1]. The assumptions and rules of thumb that have been "proven" to work for permanent magnet machine simulations in the past may fail when moving to novel topologies, and must be relaxed through the use of more robust tools.

A more subtle concern is cost in terms of the man–hours required to perform a simulation. This is directly reflected in the computational efficiency of the simulation tools used. A tool that is oriented toward the requirements of the closed–loop powertrain design process reduces this cost. Ultimately, the reduction in cost per iteration will be reflected by some combination in reduction of product cost and increase in product quality.

Increasing the accuracy of simulation tools can effect cost and performance indirectly by decreasing modeling uncertainty. It is a fundamental tenet of feedback control that one must make a compromise between performance on one hand and the ability to reject disturbances and modeling errors on the other. One must also live with modeling uncertainty with an electric machine design. The technical requirements of a machine (e.g. torque, speed, power) can be roughly translated into lower bounds for machine size and cost assuming perfect knowledge of factors that limit the machine performance. Coping with uncertainty necessarily requires a design to be oversized relative to the ideal case. Therefore, a reduction in modeling

[1] The later point is belied by the dearth of non rare-earth permanent magnet machines seen in production vehicles. This can partly be explained by the superiority of rare-earth permanent magnet machines when compared to other machine topologies using similar control strategies

uncertainty can allow, to some extent, a reduction in machine size and cost. This benefit is seemingly more difficult to quantify.

1.2 Thesis Outline

The main tools used for the analysis of electric machines, and the effects of machine geometry and material properties in particular, are based on Maxwell's equations. Maxwell's equations in their entirety are rarely used for electric machine analysis; typically some simplifying assumptions are made. Generally, the frequencies of interest in such simulations are such that electromagnetic wave propagation does not play a significant role in the results. Ultimately, this allows the effects of electric displacement currents to be ignored in Ampere's law. This results in the magneto–quasistatic approximation of Maxwell's equations that, after the introduction of additional potential fields, are combined into a single magnetic diffusion equation. In addition, certain quantities of interest in electric machine analysis are insensitive to the dynamics introduced by Faraday's law. If, in addition to the displacement field, Faraday's law is also neglected, we arrive at the magnetostatic approximation of Maxwell's equations. These two approximations are the subject of Chapter 2.

Maxwell's equations give us a view of an electric machine at a small scale. On the other hand, the control and performance of electric machines are typically characterized using large scale inputs such as voltage and current. It is necessary to have a robust and consistent method of coupling the fields to these external circuit variables when performing a realistic dynamic simulation. This issue is covered in Chapter 3.

Because of the complex geometries and nonlinear material properties inherent to electric machine analysis, it is usually not possible to use analytical solutions to Maxwell's equations without overly restrictive assumptions. Finite-element analysis is a general and robust discretization scheme capable of handling these issues, and is the topic of Chapter 4. One notable exception to the use of analytical solutions is in the airgap of electric machines, which are well-modeled as a cylindrical annulus. A general method for coupling solutions on discrete, unstructured grids to the analytic solution of Laplace's equation on a cylindrical annulus is also discussed here.

The discretization of electric machine models with nonlinear ferromagnetic materials and rotating parts gives rise to a very general index–1 differential algebraic equation. When external circuits are considered, the equations are generally of index–2. Because of the complicated nature of these equations, it is necessary to use some form of numerical integration to solve them in the temporal dimension. Chapter 5 presents the family of Runge–Kutta numerical integration schemes along with conditions any method must satisfy to achieve a given order of accuracy for our model problem. This chapter also presents a common framework on which the time–domain steady–state analysis methods are based.

Chapter 6 presents the development of several steady–state simulation algorithms. Running transient analysis until convergence is historically the earliest and certainly the simplest method for determining steady–state behavior. Alternative time domain methods are generally called the single–shooting method and direct multiple–shooting method. In the literature on electric machine finite–element analysis, they will also be referred to as the shooting–Newton and time–periodic finite–element method (TPFEM), respectively. All three of these strategies are based on the Runge–Kutta numerical integration techniques of Chapter 5. The harmonic balance method, a frequency domain approach to steady–state simulation, is also presented in Chapter 6.

Until now there has not been a detailed and rigorous study comparing the efficacy of these steady–state analysis algorithms. Chapter 7 is an attempt to fill this void by presenting a wide range of simulation results for several practical electric machine designs. After the improvements proposed in this thesis, the algorithms tend to have surprisingly similar performance with some methods being preferable under certain conditions.

1.3 Historical Overview of Steady-State Simulation Algorithms

1.3.1 Static Analysis

Application of the finite–element method to electric machine analysis began in the 1970s [3, 4, 2, 5, 6, 7, 8, 9, 10, 11, 12]. Until that time, either analytic equation modeling or finite–difference techniques had been used to study electric machines. The finite–difference approach has a serious drawback in that it is difficult to accurately capture the highly irregular geometry of electric machines on a regular grid without extreme grid refinement in areas with small aspect ratios.

Nonuniform meshes are better suited for discretizing irregular geometries, leading to a reduced number of variables and faster computation times for the finite–element method. In addition, the finite–element method is optimal in the sense that it minimizes a nonlinear functional. Applications of the finite–element method to nonlinear magnetic devices initially developed with a focus on areas that were difficult to model using analytic equations. Calculating leakage flux – flux straying outside of the main path indicated by the ferrous core – and analysis of permanent magnets with nonuniform fields are two such areas.

Because Faraday's law is not considered in the partial differential equation, some assumption must be made about the current density distribution. A uniform current density minimizes the total losses when some total value of current is constrained to flow through a fixed cross sectional area. Because the torque output of electric machines is mostly dependent on the total current (and relatively insensitive to how the current is distributed in the stator slots), and electric machines are designed to minimize losses by producing an approximately uniform current distribution, this assumption is good for performing static analysis.

1.3.2 Single Harmonic Approximation

The next major advancement was the application of the finite–element method to the magnetic diffusion equation, commonly called the eddy current problem in the field of electric machine analysis [13, 14, 15, 16, 17, 18, 19]. The magnetic diffusion equation generally captures nonuniform current distributions from the skin effect, proximity effect, and currents induced by time varying fields, all due to Faraday's law. Initially, dynamic simulations were limited to the analysis of stationary problems where the time derivative could be replaced with a single harmonic frequency domain approximation.

The single harmonic approximation operates under the assumption that the time variation of the magnetic vector potential can be described by a single complex exponential function. Nonlinear effects are difficult to incorporate into this model since a sinusoidal input to a nonlinear function does not produce a sinusoidal output. A special treatment of the nonlinearity is required to capture the average effect of the nonlinearity over one period.

Analysis of eddy current problems using this approach is limited (with one trivial exception) to simulations without motion. In fact, incorporating motion is one of the most difficult aspects of modeling electric machines using finite–elements. Some of the first attempts at solving this problem employed a fixed finite–element mesh and an iterative approach to correct for the errors that occur because the superposition principle does not hold for nonlinear magnetic problems [20]. This approach requires the moving piece to be homogeneous in the direction of motion, implying the technique is limited to rotating cylinders. Otherwise, a technique must be employed to distort the mesh, which may easily breakdown if the distortions must occur over very large distances [21]. This occurs, for example, after a complete revolution of the rotor. The fixed reference frame, or Eulerian approach, also requires the use of special "upwind" discretization schemes in order to compensate for numerical instabilities due to advective magnetic diffusion [22].

1.3.3 Transient Analysis I: Implicit–Euler

One of the greatest innovations for finite–element simulation of electric machines with rotational effects was the idea of remeshing [23, 24]; at each rotor position, a new mesh is generated in the airgap region to couple the stator and rotor domains. The difficulties associated with remeshing due to the poor aspect ratio of the airgap and numerical accuracy issues associated with a discretization that appears to be time varying in the airgap are manifold. However, modeling the rotor in a rotating reference frame removes many issues associated with capturing the rotation of geometric configurations with nonhomogeneous material properties. This motivated the development of several different approaches aimed at coupling the finite–element solutions to the analytic solution of Laplace's equation within a cylindrical annulus inside of the airgap [25, 26, 27]. The motional information shows up in terms of time varying boundary conditions imposed on the airgap by a pair of finite–element models, one for the rotor and one for the stator. The problem of modeling unbounded domains, which has a similar aspect ratio issue for the inverse reason, had already received considerable attention [28, 29, 30, 31]. Many of the techniques are similar.

Because the underlying discretization becomes time varying and the motion of the rotor will produce additional harmonics, all of these approaches to modeling rotational motion are largely incompatible with the single harmonic approximation of the magnetic diffusion equation. Numerical integration is required to marry these two ideas, but there was little theoretical or application oriented work being done to apply numerical integration techniques to any dynamic, nonlinear, magnetic field analysis problems. The first applications of time domain numerical integration used the implicit–Euler method [32] and the Crank–Nicholson method [33]. There was little interest in this approach until several years later [34, 35, 36, 37, 38]. This period coincides with the first analysis of a nonlinear, rotational, eddy current problem appearing in the mid 1980s [39, 40].

Several properties of the semi–discrete magnetic diffusion equation affect the choice of numerical integration method. Most prominently, because many of the materials are modeled as perfect insulators, the conductivity matrix is singular. This the semi–discrete equation is a differential–algebraic equation and cannot be transformed into an ordinary differential equation in a simple way. The most immediate consequence is that implicit numerical integration methods must be used, necessitating the solution of a nonlinear equation at each time step.

A more subtle issue arises because the steady–state solution is oscillatory. The numerical properties of central difference schemes like the trapezoidal and Crank–Nicholson methods lead to issues with high frequency numerical errors in the solution, which mostly negates any theoretical accuracy benefits. Conversely, backward difference schemes, properly designed for nonlinear differential–algebraic equations, will provide enough numerical damping to expediently force the high frequency errors to zero. The implicit–Euler method is the simplest method.

The almost total neglect of application oriented transient finite–element research compared to the very early adoption of the single harmonic approximation of the magnetic diffusion equation reflects the degree to which electric machine designers think of performance in terms of steady–state behavior. At the time, even though it was possible to simulate the general magnetic diffusion equation, it was probably viewed with limited utility since to obtain the steady–state behavior, one had to perform transient analysis until the waveforms converged to the steady–state behavior. In addition, the standard technique for coupling the field equations to external circuits was based on expensive iterative schemes, furthering increasing simulation time.

A major advancement came with the realization that the finite–element method could be directly coupled to external circuits through simple integral formulas relating the field and bulk parameter variables [41, 42, 43, 44, 45, 46, 47, 48]. This development was necessary for simulating multi–path eddy current problems where the total current in a bundle of wires is specified, but each individual wire may have a different total current and current density distribution than the mean. The discrete form of field–current integral constraint, along with any other circuit equations, can be incorporated directly into the semi–discrete equations without changing the form of the model equation. This reduces the simulation time of transient analysis by removing the outer loop of the previously used iterative scheme, although there are other issues with transient analysis. There were some investigations of the state of the art in eddy current modeling around this time that demonstrated an increasing awareness of the inaccuracies inherent to the single harmonic approximation [49, 50, 51]. It is probably not coincidental that the first nonlinear steady–state analysis algorithms began appearing shortly thereafter.

1.3.4 Time–Periodic Finite–Element Method

The first direct steady–state simulation algorithm applied to the magnetic diffusion equation capable of handling rotational motion and nonlinear materials in a general way was the time–periodic finite–element method (TPFEM) [52]. The main idea behind TPFEM is to simultaneously solve multiple coupled boundary value problems. The TPFEM strategy is a specific version of a more general algorithm called the multiple shooting method.

Most of the techniques investigated for solving this type linear equation revolve around relaxations schemes that exploit the block structure of the matrix [53, 54, 55, 56, 57]. For example, the overall matrix has a nearly block–lower triangular form, and an efficient relaxation method can be developed around this structure. For linear problems, the matrix is block-circulant. A useful property of circulant matrices is that they can be diagonalized by a similarity transformation based on a discrete Fourier transform matrix. Likewise, block–circulant matrices have the same property in a block–wise sense. In certain situations nearly circulant matrices can also benefit from this property [58, 59]. Unfortunately for problems with motion, essential information about the motion tends to be discarded whenever a circulant approximation is made since the average coupling between two variables, one in a fixed and another in a rotating reference frame, is zero.

There have been some improvements to the TPFEM algorithm since it first appeared, but the core idea has remained unchanged. It may be known ahead of time that the steady-state solution exhibits a different type of symmetry over an interval shorter than one period. For example, if the periodic solution is known to exhibit half–wave symmetry, then this condition can be used to close the problem and reduce the number of time points that must be solved for by half. More generally, because of the nature of the three–phase winding layouts typical to electric machines, the solution may exhibit a space–time symmetry relating the value of the solution at two different space–time coordinates. This type of condition is important for induction machines that have long fundamental periods but exhibit space–time symmetries on a shorter time scale. These ideas were developed along with some investigation of parallelization of the TPFEM algorithm [60, 61].

1.3.5 Harmonic Balance Method

The harmonic balance method refers to a class of algorithms that, in one way or another, use an expansion of the solution in terms of sinusoidal basis functions. These algorithms encompass the single harmonic approximation of Section 1.3.2 as a simplifying case. The main difference is that their formulations are general with respect to the number of harmonics, indicating they may obtain arbitrary accuracy for problems with periodic, but not necessarily sinusoidal, solutions.

The first application of the harmonic balance method to the magnetic diffusion equation appeared for stationary problems [62]. For stationary problems, the only coupling between different frequencies components occurs due to the magnetic nonlinearity. In the first harmonic balance algorithms, the magnetic nonlinearity was modeled as a power series so that the harmonic coupling could be taken into account “exactly”. There are many problems with this approach. First, exact calculation of the harmonic coupling coefficients requires the manipulation of many cumbersome equations and is not easily extended beyond polynomials of relatively low degree. This is important because ferromagnetic saturation nonlinearities cannot be represented well over a wide range with polynomials due to their asymptotic behavior. Rather, rational or exponential functions are required to model a magnetic nonlinearity using a real analytic function. It is impossible to treat these types of nonlinearities exactly, however, because the frequency response of a rational function to a sinusoidal input is not band–limited.

None–the–less, this approach was used with a 9^{th} order polynomial nonlinearity [63] and specially extended to include a dc–bias component [64]. Polynomial modeling was also used to approximate hysteresis effects by including a time derivative term in the nonlinearity [65]. Due to the large size of the system matrix for harmonic balance problems, iterative methods for solving the equations were investigated that were rather heuristic in nature and dependent on assumptions about the number of harmonics modeled [66]. The method was also extended to include external circuit coupling [67].

The limit of the applicability of exact treatment of multi–harmonic coupling is encountered when rotational motion is considered. Initially, the same unified reference frame approach employed in the single harmonic approximation was used within the harmonic balance method [68]. Because the geometries are limited to those that are uniform in the θ direction, no additional harmonic content is generated by the motion. However, any useful electromechanical device has a rotor with nonuniform material properties. The rotation of an object with material interfaces nonparallel to the direction of rotation modulates the fields setup by the stationary part of the device, generating an infinite frequency response with low–pass behavior. It is nearly impossible to exactly calculate the harmonic coupling due to this phenomenon.

General methods to treat ferromagnetic materials appeared in the form of the collocation and Galerkin methods [69]. Both of these methods treat the problem by sampling the nonlinearity in the time domain and are therefore agnostic to its functional form. Despite the generality of these methods, they were ignored in subsequent work [70, 71, 72] and, apparently, independently reinvented a decade later [73]. The properties of the collocation and

Galerkin harmonic balance methods allowing them to treat ferromagnetic materials in a general way also provide a general method for treating rotational and more arbitrary periodic motion [74, 75].

As with the time–periodic finite–element method, the most recent research into the harmonic balance method has been focused on efficient methods for solving the large system of equations. Because of the full block structure of the underlying matrices in these equations, there are no obvious structural approximations that can be exploited. The most widely used scheme is a relaxation method that chooses an optimal relaxation factor based on characteristics of the nonlinearity [76, 77, 78, 79, 80]. There have also been some investigation into alternative harmonic balance formulations and application oriented algorithms, but most lack the generality and provide no obvious benefit beyond the collocation method described above [81, 82, 83, 84, 85]. For problems without motion and mild nonlinearities, the magnitude of the off–diagonal blocks may be relatively small compared to the diagonal components and a block–diagonal approximation can work well. For problems with motion, however, the information about the motion is encoded on the off–diagonal blocks so simple block–diagonal approximations tend to offer little benefit. Importantly, most of the previously described relaxation methods have not been investigated in the context of problems with motion.

1.3.6 Shooting Newton and Explicit Error Correction Methods

The shooting Newton and explicit error correction methods are steady–state simulation algorithms that embed an additional correction procedure within the transient simulation process. The correction procedure forces the transient part of the solution to decay faster than the natural time constants of the problem. There are potential benefits and drawbacks to both the shooting Newton and EEC methods. Because the shooting Newton method is based on the Newton–Raphson procedure, convergence of the iteration is guaranteed as long as the initial guess is close enough to the true solution. Because of the complicated structure of the shooting Newton Jacobian, an iterative method must be used to solve the linear equation at each Newton iteration [86, 87]. The EEC method is more heuristic in nature, relying on matrices constructed by certain weighted averages and projections [88, 89, 90, 91, 92, 60]. Therefore, there is no guarantee that the iterative procedure will converge. On the other hand, the computation of the correction is much simpler for EEC than the shooting Newton method and will be faster if both methods exhibit similar rates of convergence per iteration.

1.3.7 Transient Analysis II: Runge–Kutta Methods

Runge–Kutta methods are widely used for the numerical integration of ordinary differential equations. Explicit methods are especially popular since they do not require solving nonlinear equations. Unfortunately, explicit methods cannot be used for differential–algebraic equations since they cannot be transformed into ODEs. It is known that, when applied to DAEs, some implicit Runge–Kutta methods fail to achieve the same degree of accuracy observed when applied to ODEs. This phenomenon is called “order reduction”. Further research revealed that additional order conditions are required beyond those for ODEs to guarantee a given order of accuracy for DAEs.

Runge–Kutta methods with full coefficient matrices have a disadvantage that the equations for each stage are coupled, requiring the solution of a nonlinear equation that is a number of times larger than the underlying DAE. Because of this, the application of Runge–Kutta methods to the analysis of magnetic devices is usually limited to diagonally–implicit Runge–Kutta methods having lower–triangular coefficient matrices. Because of the structure of these matrices, the stage–value equations can be solved sequentially [93, 94, 95].

The most obvious benefit of Runge–Kutta methods is that they are more accurate than the implicit–Euler method for a given step size, permitting the use of a larger step size while maintaining similar accuracy. Runge–Kutta methods also facilitate the use of adaptive time stepping methods that coarsen and refine the step size as dictated by some estimate of the solution error [96, 97, 98]. Related techniques can be used to extrapolate the solution to generate initial conditions for the integration procedure in the next interval [99] and more precisely determine switching events causing discontinuities in the solution or its derivatives [100, 101, 102]. In terms of steady–state analysis, the use of Runge–Kutta methods, and adaptive methods in particular, can be used to accelerate the transient analysis procedure toward the periodic steady–state solution [103].

1.4 Major Contributions

The standard by which the computational efficiency and accuracy of electric machine simulations are judged is magnetostatic analysis. Because Faraday's law is neglected, this is certainly the fastest option for performing finite–element analysis when the quantities of interest are reasonably independent of dynamic effects. For example, most electric machines employ some variation of a current regulation control strategy which, because of the nature of Ampere's law and the general topology of electric machine windings, means that the flux density in ferromagnetic materials and torque output of the machine depend primarily on the total applied current and not on the particular current density distribution.

When dynamic effects are of interest, we must move beyond simple static simulations to time–dependent analysis based on the magnetic diffusion equation. The simplest and most popular approach to dynamic analysis is performed by running a time–marching algorithm until the transient part of the solution has decayed sufficiently. The reason the steady–state behavior is of prime importance is intimately related to the various time–scales represented by the system in which the electric machine is embedded. Compared to the mechanical and thermal time scales of interest, the electrical time scales are very short. On the longer time–scales, the electrical transients can be considered to decay instantaneously, leaving us with the steady–state behavior. This assumption becomes apparent when one considers the most common metric of electric machine performance, the efficiency map, which is calculated using steady–state behavior. One can argue that the transient behavior is important when transitioning between steady–state operating points. However, the optimization targets of maximizing efficiency and power density typically redound to limited flexibility when attempting to affect transient performance. Usually the transient performance is implicitly fixed by these other targets.

The popularity of the time–marching approach can probably be attributed to the wide array of general academic research performed in the area of initial value problems that can be easily translated into different application domains. Transient analysis has benefited greatly from the robust literature on numerical integration methods and adaptive time–step selection. There is also an existing body of literature focused specifically on solving the steady–state simulation problem using shooting methods. However, it is much less developed and there are significant areas of potential improvement. In particular, the shooting methods for solving the steady–state simulation problem have only been investigated using first order numerical integration techniques. The major contribution of Chapter 5 is in demonstrating how it is possible to transform the traditional statement of Runge–Kutta methods, formulated in terms of the stage–derivatives, into one formulated purely in terms of the stage–values. This transformation is necessary to imbue the shooting methods of Chapter 6 with several attractive computational properties that arise naturally when the algorithms are formulated using the implicit–Euler method. This allows us to use Newton's method to solve the nonlinear equations and the generalized minimum residual method (GMRES) to efficiently solve the linear equations at each Newton iteration.

The application of time domain adaptivity in this context is particularly interesting since it can be modified to complement the peculiarities of steady state analysis. This results in algorithms that have a flavor closer to spatially adaptive algorithms than to traditional adaptive time–marching. Runge–Kutta methods provide a well understood framework for performing error estimation and adaptation. A major contribution of Chapter 6 is the development of an adaptive time step selection algorithm based on Runge–Kutta error indicators. This algorithm is used to construct adaptive time domain steady–state analysis algorithms by wrapping the existing fixed step size algorithms in an adaptation loop.

A common theme among all the algorithms in this thesis are block–wise decompositions. The linear matrix equations that must be solved as part of the nonlinear iteration are the result of concatenating many time steps/samples (or frequency components) of the underlying differential equation with appropriate time–derivative approximations. When compared to the somewhat random nature of matrices arising from finite–element analysis on an unstructured grid, time domain concatenation leads to a very orderly structure when the matrix is viewed block–wise. The essence of these algorithms is that linear iterative methods can exploit the matrix structure by viewing it as a "matrix of matrices" instead of a monolith. A major contribution of Chapter 6 with respect to the harmonic balance method is the development of an "overlapping" time and frequency domain preconditioner that effectively incorporates information about the time variation and dynamics of the problem. This is essential to the efficiency of the algorithm, as the standard frequency domain block–diagonal preconditioner is ineffective when used alone for problems with motion.

Some of this work is fairly general in nature, so that it could be applied to applications other than electric machine analysis. The model equations we use in deriving the algorithms are cast in a general form in order to emphasize this fact. On the other hand, from a practical point of view, there are issues of algorithm selection and tuning that depends heavily on the particular application. In the domain of electric machine analysis, algorithm performance further depends on the type of machine under consideration and the modeling assumptions made along the way. The nature of these effects is somewhat ambiguous and have not been studied in a systematic way. A major contribution of this thesis in Chapter 7 is a direct comparison of several different steady–state algorithms and the effects that various modeling choices have on their relative performance.

Chapter 2

Field Equations

In this chapter, the partial differential equation that will be studied in the subsequent chapters is derived. Section 2.1 presents Maxwell's equations in their unadulterated form. Section 2.2 introduces the material–dependent relationships between certain fields, and discusses the modeling assumptions that will be used throughout this thesis. Section 2.3 separates the dominant and secondary effects captured by Maxwell's equations in order to arrive at a quasistatic approximation that simplifies the field models on the temporal and spatial scales of interest for electric machine simulations. The magnetic vector potential and electric scalar potential are introduced in Section 2.4, which further simplify the representation of the quasistatic field equations. Section 2.5 introduces the field approximations used to derive two–dimensional models of electric machines. Boundary conditions for the partial differential equation based on periodicity and "magnetic insulation" are discussed in Section 2.6. Somewhat related to boundary conditions, Section 2.7 discusses the analytic solution to Laplace's equation in a cylindrical annulus. This result can be used to couple equations expressed in two different references frames and provides an elegant method for modeling rotational motion.

2.1 Maxwell's Equations

The differential form of Maxwell's equations are given below:

$$\nabla \cdot \mathbf{D} = \rho, \tag{2.1}$$

$$\nabla \cdot \mathbf{B} = 0, \tag{2.2}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \tag{2.3}$$

$$\nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t}. \tag{2.4}$$

Equation (2.1) is Gauss's law relating the free charge density ρ and the electric flux density **D**. Equation (2.2) is Gauss's law for magnetism. It is qualitatively distinct from (2.1) in that it states there are no magnetic monopoles; from any point in space, any line of magnetic flux density **B** passing through the point under consideration can be traced back to it. Faraday's law in equation (2.3) relates a time–varying magnetic flux density to the electric field **E**. Ampere's law in (2.4) relates the magnetic field intensity **H** to the free charge current density **J** and the time rate of change of the electric flux density.

Equations (2.2) and (2.3) are self-consistent in that taking the divergence of (2.3) and substituting (2.2) yields a logical truth:

$$\nabla \cdot \nabla \times \mathbf{E} = -\nabla \cdot \frac{\partial \mathbf{B}}{\partial t} = -\frac{\partial \nabla \cdot \mathbf{B}}{\partial t} = 0. \tag{2.5}$$

Similarly, taking the divergence of (2.4) and substituting (2.1) produces the equation for conservation of electric charge:

$$\nabla \cdot \mathbf{J} = -\frac{\partial \rho}{\partial t}. \tag{2.6}$$

Note that the conservation of charge does not have to be assumed; (2.6) is a natural consequence of Maxwell's equations. However, consideration of (2.6) can provide some useful insights so we shall carry it along with the rest of Maxwell's equations for the remainder of the chapter.

2.2 Constitutive Equations

At the scale that we are interested in modeling, only 2 of the 5 fields, along with the free charge density, are unique. The non–unique fields are related through constitutive equations. Because the constitutive equations are intended to capture relationships observable at the macroscopic level, they are based on some material–dependent modeling assumptions. The proper modeling assumptions are highly dependent on the effects that are required to be captured accurately, and also on the time scale on which those effects occur.

In general, it will be assumed that the electric flux density and electric current density are linearly and isotropically related to the electric field intensity:

$$\mathbf{D} = \epsilon_0 \mathbf{E} + \mathbf{P}, \tag{2.7}$$

$$\mathbf{J} = \sigma \mathbf{E}, \tag{2.8}$$

where ϵ_0 is the vacuum permittivity, $\mathbf{P}$ is the polarization of the material, and σ is the electrical conductivity of the material. The polarization is generally some function of $\mathbf{E}$, the details of which turn out to be relatively unimportant for the applications we are interested in. This will be evident after the discussion of the magnetoquasistatic approximation in Section 2.3.

The modeled value of the conductivity σ can range from 0 for nonconducting materials to values on the order of 10^7 for high quality conductors such as copper. Since material conductivity is strongly dependent on temperature, some assumption on the temperature distribution must be made in order to fix this value for each material. We will assume that each material region is at a constant uniform temperature, although discrete jumps may occur whenever material discontinuities are encountered. This is a fair assumption, as materials with large electrical conductivities tend to have large thermal conductivities, resulting in a feedback mechanism that prevents the generation of appreciable temperature gradients over the spatial scales of interest when Joule heating is the dominant loss mechanism.

Several possible constitutive relationships exist between **H** and **B**. In nonmagnetic materials, the relationship is simply

$$\mathbf{H} = \nu_o \mathbf{B}, \tag{2.9}$$

where $\nu_0 = \frac{10^{-7}}{4\pi}$ is the vacuum reluctivity. Often magnetic materials are modeled using the linear relationship

$$\mathbf{H} = \nu \cdot \mathbf{B} \tag{2.10}$$

where ν is the reluctivity tensor. A further simplification can be made if the material is assumed to be isotropic. Then it is common to write

$$\mathbf{H} = \nu \mathbf{B} \tag{2.11}$$

where ν is now a scalar. The reluctivity values can be normalized to the vacuum reluctivity by introducing the relative reluctivity ν_r such that

$$\nu = \nu_0 \nu_r \tag{2.12}$$

It is also possible to write the linear **B**–**H** relationships using permeability properties, usually denoted μ, that is inversely related to the reluctivity:

$$\mu = \nu^{-1} \tag{2.13}$$

This is perhaps more common, but for our purposes the permittivity model turns out to be more convenient.

No magnetic material is truly linear. While linear relationships are useful for analytical calculations, an essential detail that must be captured when operating at high field levels is the presence of magnetic saturation. A standard way of doing this is to assume ν is a nonlinear function of $\mathbf{B}$ (or possibly $\mathbf{H}$). Instead, the approach taken here is to start with a cursory consideration of another important magnetic phenomenon, hysteresis, and simplify it somewhat to obtain a logically equivalent model that somewhat better represents the physical underpinnings of magnetism.

In ferromagnetic materials, the most general relationship between $\mathbf{B}$ and $\mathbf{H}$ is governed by the evolution of the magnetic domains intrinsic to the material occurring on the micrometer level [104, 105]. The details of the magnetic domains are mesoscopic effects, which depend on defects in the crystal lattice structure and even local stresses and strains. This is more appropriately the subject of micromagnetics, and is too detailed a consideration for our purposes. If instead we consider the ensemble average effect of the individual magnetic domains, a quite general macroscopic constitutive relationship can be posited:

$$\mathbf{B}\left(t\right) = \mu_0 \left[\mathbf{H}\left(t\right) + \mathbf{M}\left(\mathbf{H}\left(\tau\right); \tau \leq t\right)\right], \tag{2.14}$$

where $\mathbf{M}$ is the (macroscopic) magnetization. The arguments of the magnetization indicate that $\mathbf{M}$ may depend on the entire history of $\mathbf{H}$.

The evolution of the magnetization is governed by the minimization of complex, nonconvex energy potentials that are determined by the coupling of various phenomena occurring at disparate spatial scales. At the macroscopic level, this effect is apparent through the observation of hysteresis loops in plots of $|\mathbf{M}|$ or $|\mathbf{B}|$ versus $|\mathbf{H}|$. Magnetic hysteresis is a significant loss mechanism in magnetic devices, and accurately capturing this effect is essential to optimizing certain designs.

Hysteresis modeling is fraught with difficulties, due to a dearth of empirical data and the lack of a clear theory for passing from first–principles micromagnetic models to the macroscopic scale. In order to simplify our modeling, we shall assume the following (instantaneous) constitutive relationship holds:

$$\mathbf{H} = \nu_0 \mathbf{B} - \mathbf{M}\left(\mathbf{B}\right). \tag{2.15}$$

Notice that $\mathbf{M}$ is now considered as a function of $\mathbf{B}$, which will prove useful later on. Equation (2.15) represents a general non–isotropic relationship where $\mathbf{M}$ and $\mathbf{B}$ do not necessarily point in the same direction. Such a relationship is necessary for modeling materials that demonstrate a strong "preferred" magnetization direction.

From (2.15), we make two more material dependent assumptions. First, for permanent magnet materials, we assume an affine relationship holds:

$$\mathbf{H} = \nu_0 \mathbf{B} - \mathbf{M}_r, \tag{2.16}$$

where $\mathbf{M}_r$ is the remanent magnetization. This relationship for the permanent magnets is only valid below the Curie temperature and when the applied field opposing $\mathbf{M}_r$ is not too large. For non–grain–oriented steels, we make the further simplifying assumption that

$$\mathbf{M} = \chi \mathbf{B}, \tag{2.17}$$

where χ is the magnetic susceptibility; comparing (2.17) to (2.15), evidently

$$\chi = (1 - \nu_r) = \frac{|\mathbf{M}|}{|\mathbf{B}|}. \tag{2.18}$$

Note that this definition of χ is somewhat different than the standard definition of magnetic susceptibility as $\mathbf{M} = \chi \mathbf{H}$. Equation (2.18) demonstrates an equivalency between magnetization and reluctivity modeling under these assumptions. We will assume all nonlinear magnetic materials are non–grain–oriented.

If we accept the fact that the effects of hysteresis are neglected, the preceding assumptions are reasonable when simulating electric machines. Most electric machines use non–grain–oriented steel as there is not a single preferred direction for the magnetic flux. Similarly, if the field levels are high enough to appreciably demagnetize any permanent magnets, it is necessary to include hysteretic effects in order to accurately capture this phenomenon. Further, the demagnetization characteristics of hard magnetic materials are temperature dependent, which adds additional complications beyond the inclusion of temperature–independent hysteresis effects.

Substituting these relationships into Equations (2.1)–(2.6) yields the following modified Maxwell's equations:

$$\epsilon_0 \nabla \cdot \mathbf{E} + \nabla \cdot \mathbf{P} = \rho, \tag{2.19}$$

$$\nabla \cdot \mathbf{B} = 0, \tag{2.20}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \tag{2.21}$$

$$\nu_0 \nabla \times \mathbf{B} - \nabla \times \mathbf{M} = \sigma \mathbf{E} + \epsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \frac{\partial \mathbf{P}}{\partial t}, \tag{2.22}$$

$$\nabla \cdot (\sigma \mathbf{E}) = -\frac{\partial \rho}{\partial t}. \tag{2.23}$$

Notice that, because σ may contain finite discontinuous jumps at material interfaces, it is not possible to use the linearity of the divergence to put (2.23) into a simpler form. The magnetic nonlinearity is assumed to be included in $\mathbf{M}$ in the right hand side of (2.22) along with the remanent magnetization of any permanent magnets.

2.3 Magnetoquasistatic Approximation

Equations (2.19)–(2.23) are quite general; they contain a menagerie of details that are often irrelevant on the spatial and temporal scales of interest. In order to tease out the important effects, they can be normalized by introducing a characteristic length λ, characteristic time τ, and characteristic flux density $\mathcal{B}$ [106]. The following change of variables is then performed:

$$\begin{aligned}
t &= \tau \underline{t} \\
\nabla &= \frac{1}{\lambda}\underline{\nabla} \\
\mathbf{B} &= \mathcal{B}\underline{\mathbf{B}} \\
\mathbf{M} &= \nu_0 \mathcal{B}\underline{\mathbf{M}} \\
\mathbf{E} &= \frac{\lambda\mathcal{B}}{\tau}\underline{\mathbf{E}} \\
\mathbf{P} &= \frac{\epsilon_0\lambda\mathcal{B}}{\tau}\underline{\mathbf{P}} \\
\rho &= \frac{\epsilon_0\mathcal{B}}{\tau}\underline{\rho}
\end{aligned} \tag{2.24}$$

Making these substitutions into Equations (2.19)–(2.23) (and dropping the underbars for convenience) gives the normalized equations:

$$\nabla \cdot \mathbf{E} + \nabla \cdot \mathbf{P} = \rho, \tag{2.25}$$

$$\nabla \cdot \mathbf{B} = 0, \tag{2.26}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \tag{2.27}$$

$$\nabla \times \mathbf{B} - \nabla \times \mathbf{M} = \frac{\lambda^2\sigma}{\tau\nu_0}\mathbf{E} + \frac{\lambda^2}{\epsilon_0\nu_0\tau^2}\left(\frac{\partial \mathbf{E}}{\partial t} + \frac{\partial \mathbf{P}}{\partial t}\right), \tag{2.28}$$

$$\nabla \cdot (\sigma\mathbf{E}) = -\frac{\lambda\epsilon_0}{\tau}\frac{\partial \rho}{\partial t}. \tag{2.29}$$

The normalized Ampere's law (2.28) and charge conservation law (2.29) can be simplified by introducing three characteristics times related to the time–scales on which different phenomena occur. The charge relaxation time, τ_e, given by

$$\tau_e = \frac{\varepsilon_0}{\sigma}, \tag{2.30}$$

is related to the time it takes for a local charge density to dissipate in a conducting medium.

The magnetic diffusion time, τ_m, given by,

$$\tau_m = \frac{\sigma \lambda^2}{\nu_0}, \tag{2.31}$$

is related to the time required for magnetic fields to penetrate into a conducting medium at the spatial scale of interest. Finally, the electromagnetic wave propagation time, τ_{em}, given by

$$\tau_{em} = \sqrt{\frac{\epsilon_0}{\nu_0}}\lambda, \tag{2.32}$$

is related to the time for an electromagnetic wave in the material to travel over the spatial scale of interest. The wave propagation time is the geometric mean of τ_e and τ_m:

$$\tau_{em} = \sqrt{\tau_e \tau_m}. \tag{2.33}$$

In terms of these times, (2.28) and (2.29) become, respectively,

$$\nabla \times \mathbf{B} - \nabla \times \mathbf{M} = \frac{\tau_m}{\tau}\mathbf{E} + \frac{\tau_{em}^2}{\tau^2}\left(\frac{\partial \mathbf{E}}{\partial t} + \frac{\partial \mathbf{P}}{\partial t}\right), \tag{2.34}$$

$$\nabla \cdot (\sigma \mathbf{E}) = -\frac{\tau_e}{\tau}\frac{\partial \rho}{\partial t}. \tag{2.35}$$

The magnetoquasistatic approximation is based on the observation that, on the temporal and spatial scales of interest, τ_m is $O(\tau_c)$ and $\tau_e \ll \tau_m$. Further, as a consequence of (2.33), $\tau_e \ll \tau_m$ implies $\tau_e \ll \tau_{em} \ll \tau_m$. We arrive at the desired approximation by taking the limit of (2.34) and (2.35) as $\tau_e \to 0$. In words, the limiting procedure implies that displacement current effects are negligible and no significant charge accumulation occurs.

Some consideration of typical values for λ and τ is illuminating. The characteristic length can range from 10^{-4} meters for the cross sections of small wires to 10^1 meters for the circumference of generators for power systems. Characteristic times of interest are mostly governed by excitation frequencies, and can range from 10^{-4} seconds at inverter switching

Table 2.1: Typical ranges of characteristic times in seconds

τ_e	τ_{em}	τ_m	τ
10^{-19}–10^{-15}	10^{-12}–10^{-8}	10^{-5}–10^{1}	10^{-4}–10^{-1}

frequencies up to 10^{-1} seconds at line frequencies.

Table 2.1 gives ranges for the characteristic times based on some typical material properties. It is evident that the magnetoquasistatic approximation is universally good at the scales of interest. Only once the excitation reaches the megahertz frequency range does electromagnetic wave propagation begin to have a significant effect. Magnetic diffusion is a different matter; the range of τ_m overlaps broadly with the timescales of interest and will play a significant role in many instances.

Reverting to the unnormalized equations, we apply this approximation and reorder the equations by importance:

$$\nu_0 \nabla \times \mathbf{B} - \nabla \times \mathbf{M} = \sigma \mathbf{E}, \tag{2.36}$$

$$\nabla \cdot \mathbf{B} = 0, \tag{2.37}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \tag{2.38}$$

$$\nabla \cdot (\sigma \mathbf{E}) = 0, \tag{2.39}$$

$$\epsilon_0 \nabla \cdot \mathbf{E} + \nabla \cdot \mathbf{P} = \rho. \tag{2.40}$$

Ampere's law (2.36) and Gauss's law for magnetism (2.37) govern the fundamental field behavior, with dynamics entering through Faraday's law (2.38). The charge conservation equation (2.39) is instructive as to the type of boundary conditions that the current density $\sigma\mathbf{E}$ must obey at material interfaces. The charge density is rarely of interest in such problems; equation (2.40) can be regarded as determining ρ as an output of the magnetoquasistatic system in those cases when it is of interest. From now on, (2.40) will be disregarded.

2.4 Potential Formulation

The three dominant equations of the magnetoquasistatic approximation can be combined into a single equation using a potential formulation of Ampere's law. This has the effect of reducing the number of field components from 6, 3 for each of **B** and **E**, down to 4. The charge conservation equation augments the potential form of Ampere's law to provide the appropriate number of degrees of freedom.

Because the flux density is solenoidal (2.37), **B** can be written in terms of the curl of the magnetic vector potential **A**:

$$\mathbf{B} = \nabla \times \mathbf{A}. \tag{2.41}$$

Substituting this into Faraday's law (2.38), we find

$$\nabla \times \left(\mathbf{E} + \frac{\partial \mathbf{A}}{\partial t} \right) = 0. \tag{2.42}$$

Because $\mathbf{E} + \frac{\partial \mathbf{A}}{\partial t}$ is a conservative field, it can be written in terms of the gradient of the electric scalar potential ϕ:

$$\mathbf{E} = -\frac{\partial \mathbf{A}}{\partial t} - \nabla \phi. \tag{2.43}$$

Substituting (2.41) and (2.43) into Amperes's law (2.36) and the charge conservation law (2.39) gives the following equations:

$$\sigma \frac{\partial \mathbf{A}}{\partial t} + \nu_0 \nabla \times \nabla \times \mathbf{A} - \nabla \times \mathbf{M} + \sigma \nabla \phi = 0, \tag{2.44}$$

$$\nabla \cdot \left(\sigma \frac{\partial \mathbf{A}}{\partial t} + \sigma \nabla \phi \right) = 0. \tag{2.45}$$

A vector field can be completely specified by its curl and divergence. In this sense, there is still an extra degree of freedom associated with divergence of **A** because only its curl has been specified through (2.41). How this matter is addressed is an issue of great sensitivity when considering problems with non–uniform magnetic properties. Often, the approach taken depends on the method used to solve the equations and any simplifying assumptions that are employed. For example, the issue disappears when the two–dimensional approximation of Section 2.5 is applied.

2.5 Cartesian Two-Dimensional Approximation

Often, the geometry of certain problems is dominated by a single characteristic direction or plane. The construction of radial–flux electric machines involves stacking ferromagnetic laminations along the z–direction in a uniform or quasi–uniform way. Conductors run through the laminations so that the dominant electric field and electric currents are in the z–direction and produce magnetic fields that are confined primarily to the x–y plane. Because of (2.43), the condition that **E** lies purely in the z–direction implies that

$$\mathbf{A} = \begin{bmatrix} 0 \\ 0 \\ A_z \end{bmatrix} \tag{2.46}$$

and

$$\nabla\phi = \begin{bmatrix} 0 \\ 0 \\ \frac{\partial\phi}{\partial z} \end{bmatrix}. \tag{2.47}$$

The condition that E_z is constant along the z–direction implies that both A_z and $\frac{\partial\phi}{\partial z}$ are constant as well. This situation models an infinitely long electric machine with uniform properties in the axial direction. In two dimensions, the only essential characteristic of ϕ is its partial derivative in the z–direction, which is interpreted as a voltage per unit length. For notational convenience, we will write

$$E_\phi = \frac{\partial \phi}{\partial z}, \tag{2.48}$$

that is, E_ϕ is the part of $E = E_z$ produced by $\nabla\phi$. Because the two other components of $\nabla\phi$ are zero, E_ϕ must be assumed to be constant in the x–y plane, with finite jumps only occurring at material interfaces.

Substituting these assumptions into (2.44) gives the two–dimensional magnetoquasistatic version of Ampere's law:

$$\sigma\frac{\partial A}{\partial t} + \nu_0 \nabla' \times \nabla \times A - \nabla' \times \mathbf{M} + \sigma E_\phi = 0 \tag{2.49}$$

where the subscript has been dropped from $A_z = A$. Now, when we write

$$\nabla \times A = \frac{\partial A_z}{\partial y}\hat{x} - \frac{\partial A_z}{\partial x}\hat{y} \tag{2.50}$$

it is understood to be the restriction to the part of the curl operator that acts on the z–component of the input vector. Similarly, when we write

$$\nabla' \times \mathbf{B} = -\frac{\partial B_x}{\partial y} + \frac{\partial B_y}{\partial x} \tag{2.51}$$

it is understood to be the restriction to the part of the curl operator that produce a vector pointing in the z-direction. Finally, we note that the current conservation equation (2.45) is naturally satisfied under these assumptions, so only (2.49) is required to determine the fields.

In (2.49), we could have used the fact that $\nabla' \times \nabla\times = -\nabla^2$ to write the problem using the scalar Laplacian. For expository purposes, however, it conceptually useful to remain with the curl–curl notation since it reminds us that the underlying problem we are solving is fundamentally different from a scalar or vector Laplacian equation. This is important for two reasons. First, the different underlying operator requires us to interpret boundary conditions differently, as will be discussed in the next section. Second, even though the simplification to the scalar Laplacian occurs in two dimensions without additional assumptions, this is not the case for three–dimensional problems. In fact, it is common to see the assumption of the Coulomb gauge ($\nabla \cdot A = 0$) for two–dimensional problems even though it is not necessary. For many years this assumption was also used in three–dimensional analysis, albeit with great difficulty. It is now well known that three–dimensional curl–curl problems with general types of material discontinuities must be treated with discretization schemes that are different than those used for vector Laplacian problems. This occurs precisely because of different requirements on field continuity and the types of singularities that occur at sharp corners. Even though we will not be performing any three–dimensional analysis, we would like to keep the two– and three– dimensional analogy as tight as possible.

2.6 Boundary Conditions

Two different types of boundary conditions are common for electric machines. The first arises from θ–periodicity conditions, and the second from flux containment considerations. There is a third type of boundary that arises when modeling rotational motion, which is discussed in detail in Section 2.7.

2.6.1 Periodic Boundary Conditions

Radial flux–electric machines exhibit a special periodic structure in both their geometry and excitation. This results in solutions that are periodic (in polar coordinates) with a certain fraction $f_p = \frac{p}{N_p}$, p even, of the machine:

$$A\left(\theta\right) = A\left(\theta + 2\pi f_p k\right), \tag{2.52}$$

$$E_\phi\left(\theta\right) = E_\phi\left(\theta + 2\pi f_p k\right), \tag{2.53}$$

$$\tag{2.54}$$

where N_p is the number of machine poles and k is an arbitrary integer. When the solution also exhibits half–wave symmetry, we can go one step further and choose an odd p such that

$$A\left(\theta\right) = -A\left(\theta + 2\pi f_p k\right), \tag{2.55}$$

$$E_\phi\left(\theta\right) = -E_\phi\left(\theta + 2\pi f_p k\right), \tag{2.56}$$

$$\tag{2.57}$$

which is called antiperiodicity. When either (2.52) and (2.53), or (2.55) and (2.56) are satisfied, the continuum model can be restricted to a fraction f_p of the machine. Thus, we require the (anti)periodic condition

$$A\left[r, \theta\left(r\right)\right] = s_p A\left[r, \theta\left(r\right) + 2\pi f_p\right] \tag{2.58}$$

with

$$s_p = 1 - 2\,\texttt{modulo}\left(p, 2\right) = \mp 1 \tag{2.59}$$

to be satisfied as part of the boundary conditions of the partial differential equation. A similar condition is required on E_ϕ. However, since E_ϕ is piecewise constant in two–dimensions, care can be taken to defined the boundary line $\left(r, \theta\left(r\right)\right)$ of the model around any regions associated with constant E_ϕ and nonzero conductivity whenever possible. If the geometry dictates this is not possible for a particular region, the antiperiodicity assumptions imply $E_\phi = 0$ in this area. Therefore, under the antiperiodicity assumptions we can always choose $\left(r, \theta\left(r\right)\right)$ to be drawn through regions where $E_\phi = 0$.

2.6.2 Magnetic Insulation (Dirchlet) Boundary Condition

Electric machines, and magnetic devices in general, are designed to have preferred magnetic flux paths. Fringing fields occurring outside of these paths are generally regarded as undesirable. Therefore, if the boundary of the partial differential equation is chosen suitably far away from the fields of interest, it is natural to assume that no magnetic flux passes through the boundary:

$$\mathbf{B} \cdot \mathbf{n} = 0 \text{ on } \Gamma_0 \tag{2.60}$$

where $\mathbf{n}$ is the unit vector normal to the boundary Γ_0. Since $\mathbf{B} = \nabla \times A$, (2.60) is satisfied whenever A is constant on Γ_0. If we also assume that the solution is antiperiodic, this implies (2.60) is equivalent to

$$A = 0 \text{ on } \Gamma_0. \tag{2.61}$$

When the solution is merely periodic, A can be any constant (the value of which is unimportant), so that (2.61) is uniformly valid.

2.6.3 Tangential Field (Neumann) Boundary Condition

It is the tangential part of $\mathbf{H}$ that is continuous across material boundaries. Since the Dirchlet boundary condition is equivalent to specifying the normal flux density $\mathbf{B}$, Neumann boundary conditions for magnetic field problems are properly described in terms of the tangential part of magnetic field intensity $\mathbf{H}$:

$$\mathbf{H} \times \mathbf{n} = H_t \text{ on } \Gamma_t. \tag{2.62}$$

Neumann boundary conditions are seldom used in isolation for electric machine analysis. More often they are used to couple solutions on independently modeled domains, as described in the following section.

2.7 Boundary Transfer Relationships

It is usually possible to decompose an electric machine into regions with complex geometry associated with the materials (laminations, windings, magnets) that are surrounded by free space or gaps having simple shapes. For radial flux machines, the gaps are modeled as cylindrical annuli and the fields obey Laplace's equation

$$-\nu_0 \nabla \times \nabla \times A = \nu_0 \nabla^2 A = 0. \tag{2.63}$$

This equation has a known analytical solution in terms of boundary harmonics. The analytical solution can be used to couple the solutions on disjoint subdomains with complex geometries. The benefit of doing this is the ease with which the coupling framework allows us to model rotational motion by a spatial phase shift corresponding to multiplication by a diagonal matrix. The development in this section closely follows the presentation in [106].

2.7.1 Flux-Potential Transfer Relationships

Let Ω be a cylindrical domain with inner boundary Γ_β with radius β and outer boundary Γ_α with radius α. We assume that the magnetic vector potential on each boundary can be decomposed into a sum of spatial harmonics in the θ direction:

$$A\left(\alpha, \theta\right) = \sum_k \tilde{A}_k^\alpha e^{jk\theta} \tag{2.64}$$

$$A\left(\beta, \theta\right) = \sum_k \tilde{A}_k^\beta e^{jk\theta} \tag{2.65}$$

Because Laplace's equation is linear, no harmonic mixing occurs and this decomposition allows us to focus on the solution to (2.63) for a single spatial harmonic. We can assume the solution to the equation takes the form

$$\tilde{A}_k\left(r,\theta\right) = \tilde{A}_k\left(r\right)e^{jk\theta}. \tag{2.66}$$

Substitution into Laplace's equations yields the following differential equation in a single variable r,

$$\frac{\partial^2 \tilde{A}_k}{\partial r^2} + \frac{1}{r}\frac{\partial \tilde{A}_k}{\partial r} - \frac{k^2}{r^2}\tilde{A}_k = 0, \tag{2.67}$$

subject to the Dirchlet boundary conditions $\tilde{A}_k\left(\alpha\right) = \tilde{A}_k^{\alpha}$ and $\tilde{A}_k\left(\beta\right) = \tilde{A}_k^{\beta}$. The solution to this problem can be written as a linear combination of the polynomials r^k and r^{-k} [106] in the form

$$\tilde{A}_k\left(r\right) = \tilde{A}_k^{\alpha}\mathcal{T}_k\left(\frac{\beta}{r},\frac{\beta}{\alpha}\right) + \tilde{A}_k^{\beta}\mathcal{T}_k\left(\frac{r}{\alpha},\frac{\beta}{\alpha}\right), \tag{2.68}$$

where $\mathcal{T}$ is given by

$$\mathcal{T}_k\left(x,y\right) = \frac{x^k - x^{-k}}{y^k - y^{-k}}. \tag{2.69}$$

The tangential field intensity, pointing in the θ–direction in cylindrical coordinates, for each spatial harmonic $\tilde{H}_{t,k}$ is found by taking the derivative of (2.69) with respect to r:

$$\tilde{H}_{t,k} = \nu_0\frac{\partial \tilde{A}_k\left(r\right)}{\partial r}. \tag{2.70}$$

Taking the derivative of $\mathcal{T}_k$ with respect to r and using the chain rule, we find the general relationship

$$\frac{\partial \mathcal{T}_k\left(x,y\right)}{\partial x}\frac{\partial x}{\partial r} = \frac{kx^{k-1} + kx^{1-k}}{y^k - y^{-k}}\frac{\partial x}{\partial r}. \tag{2.71}$$

Of particular interest is the evaluation of this function at each boundary, $r = \alpha$ and $r = \beta$, for each functional form of x. The four results display a certain degree of symmetry that can be summarized in two functions capturing the effects of the tangential field intensity on the "near" boundary, for which $x = \frac{\beta}{\alpha}$, and the "far" boundary, for which $x = 1$. We summarize the near– and far–effects in the functions $\mathcal{F}_k$ and $\mathcal{G}_k$:

$$\mathcal{F}_k\left(x,y\right) = \nu_0 \frac{k}{y} \frac{\frac{x}{y}^k + \frac{y}{x}^k}{\frac{x}{y}^k - \frac{y}{x}^k}, \tag{2.72}$$

$$\mathcal{G}_k\left(x,y\right) = \nu_0 \frac{2k}{x} \frac{1}{\frac{x}{y}^k - \frac{y}{x}^k}. \tag{2.73}$$

On the boundaries $r = \alpha$ and $r = \beta$, the tangential field intensity harmonics are written using these functions as follows:

$$\tilde{H}_t^{\alpha} = \mathcal{F}_k\left(\beta,\alpha\right)\tilde{A}_k^{\alpha} + \mathcal{G}_k\left(\alpha,\beta\right)\tilde{A}_k^{\beta} \tag{2.74}$$

$$\tilde{H}_t^{\beta} = \mathcal{G}_k\left(\beta,\alpha\right)\tilde{A}_k^{\alpha} + \mathcal{F}_k\left(\alpha,\beta\right)\tilde{A}_k^{\beta} \tag{2.75}$$

2.7.2 Harmonic Expansion

In order to couple the analytical solution in the airgap to the field equations in complex domains less amenable to analytic treatment, it is necessary to represent the boundary values of these domains in terms of spatial harmonics. This is conceptually straightforward since the functions $e^{-jk\theta}$ form an orthogonal basis for periodic functions on the one dimensional manifold in cylindrical coordinates represented by constant r. The harmonic representation is the spatial solution expressed in this basis, identified by the Fourier integrals:

$$\tilde{A}_k\left(r\right) = \frac{1}{2\pi}\int_0^{2\pi} e^{-jk\theta} A\left(r,\theta\right) d\theta \tag{2.76}$$

This basis is countably infinite, which creates some difficulties in computer implementations and will be dealt with later. In the mean–time, we can represent the calculation of the entire set of harmonics through the operator $\mathcal{D}$:

$$\tilde{A} = \mathcal{D}A \tag{2.77}$$

Likewise there is an inverse operator $\mathcal{P}(\theta)$ that transforms the harmonic coefficients to the value of A at angle θ in the physical space:

$$A(\theta) = \mathcal{P}(\theta)\,\tilde{A} \tag{2.78}$$

In the continuous case, there is no need to make distinctions between the projection $\mathcal{D}$ (or $\mathcal{P}(\theta)$) occurring at different values of r since the circular boundaries are isometric with respect to θ. The situation is different in implementation. In all likelihood, each circular boundary will have a different discretization and require a different treatment in terms of transforming between the spatial and harmonic representations. Again, this will be dealt with later. For right now, we simply anticipate the need to distinguish transformations on different boundaries by attaching subscripts to the $\mathcal{D}$ and $\mathcal{P}(\theta)$ operators.

This allows us to write the tangential field intensity at $r = \alpha$ and $r = \beta$ using only the values of A:

$$H_t^{\alpha}(\theta) = \mathcal{P}_{\alpha}(\theta)\,\mathcal{F}_{\beta,\alpha}\mathcal{D}_{\alpha}A_{\alpha} + \mathcal{P}_{\alpha}(\theta)\,\mathcal{G}_{\beta,\alpha}\mathcal{D}_{\beta}A_{\beta} \text{ on } \Gamma_{\alpha}, \tag{2.79}$$

$$H_t^{\beta}(\theta) = \mathcal{P}_{\beta}(\theta)\,\mathcal{G}_{\alpha,\beta}\mathcal{D}_{\alpha}A_{\alpha} + \mathcal{P}_{\beta}(\theta)\,\mathcal{F}_{\alpha,\beta}\mathcal{D}_{\beta}A_{\beta} \text{ on } \Gamma_{\beta}, \tag{2.80}$$

where $\mathcal{F}$ and $\mathcal{G}$ are diagonal scaling operators with values corresponding to the functions $\mathcal{F}_k$ and $\mathcal{G}_k$ previously described:

$$\{\mathcal{F}_{x,y}\}_{k,k} = \mathcal{F}_k(x,y), \tag{2.81}$$

$$\{\mathcal{G}_{x,y}\}_{k,k} = \mathcal{G}_k(x,y). \tag{2.82}$$

These expressions can be used to couple two computational domains that are separated by a cylindrical annulus. First, the equations are formulated separately in each domain with Neumann boundary conditions, represented by equation (2.62), on the two boundaries of the annulus. The previously developed expressions for the tangential field intensity in terms of the magnetic vector potential are then substituted into these boundary conditions, resulting in two Dirchlet-like conditions that must be satisfied simultaneously with the rest of the field equations.

2.7.3 Rotational Motion

One interesting property of the coupling framework previously described is the ease of implementing relative rotational motion between two domains described in different reference frames. Consider a reference frame that is fixed in space. If a second reference frame is rotating with respect to the first with a constant rotational velocity ω_r, the difference between an angle measured in the two reference frames is given by $\omega_r t$. In other words, angles in the two reference frames are related by

$$\theta = \theta' + \omega_r t. \tag{2.83}$$

If instead of θ we use θ' in (2.76), i.e. perform the change of basis in the rotating reference frame, by substituting (2.83) and noting $\frac{\partial\theta'}{\partial\theta} = 1$, we find

$$\begin{aligned}\tilde{A}'_k =& \frac{1}{2\pi}\int_0^{2\pi} e^{-jk\theta'} A' d\theta' \\ =& \frac{1}{2\pi}\int_0^{2\pi} e^{-jk(\theta-\omega_r t)} A d\theta \\ =& \tilde{A}_k e^{jk\omega_r t}.\end{aligned} \tag{2.84}$$

In words, harmonics in the two reference frames differ by a phase–shift of magnitude $k\omega_r t$. This is easily incorporated into the previously developed framework because a phase shift is implemented by a complex diagonal scaling operation in the harmonic domain. To this end, we can introduce a rotation operator $\mathcal{R}$ with nonzero "diagonal entries"

$$\{\mathcal{R}\}_{k,k} = e^{jk\omega_r t} \tag{2.85}$$

and rewrite the relationship between H_t and A to take into account the fact that the variables on the boundaries Γ_α and Γ_β may be expressed in different frames of reference:

$$H_t^\alpha(\theta) = \mathcal{P}_\alpha(\theta)\,\mathcal{F}_{\beta,\alpha}\mathcal{D}_\alpha A_\alpha + \mathcal{P}_\alpha(\theta)\,\mathcal{G}_{\beta,\alpha}\mathcal{R}\mathcal{D}_\beta A_\beta \text{ on } \Gamma_\alpha, \tag{2.86}$$

$$H_t^\beta(\theta) = \mathcal{P}_\beta(\theta)\,\mathcal{G}_{\alpha,\beta}\mathcal{R}^H\mathcal{D}_\alpha A_\alpha + \mathcal{P}_\beta(\theta)\,\mathcal{F}_{\alpha,\beta}\mathcal{D}_\beta A_\beta \text{ on } \Gamma_\beta. \tag{2.87}$$

Chapter 3

Field–Circuit Coupling

Electric machines are constructed with many current–carrying conductors that are connected in series and/or parallel. When a two–dimensional approximation of a machine is used for simulations, only the cross sections of the conductors are modeled. This necessitates the use of additional constraints between the currents and voltages due to the fact that a single conductor intersects the modeled cross section multiple times. The most general coupling method is based on field–current constraints and is discussed in Section 3.1. If certain conditions are satisfied, voltage constraints may be utilized. The requirements for this type of coupling are addressed in Section 3.2. Most of these techniques are scattered in various places in the literature and this chapter represents an attempt to present them within a unified framework [41, 42, 43, 44, 45, 46, 47, 48].

Even when the two–dimensional approximation is not used, field–current and field–voltage constraints are necessary at the winding terminals in order to couple the field equations to any external circuits. The simplest case occurs when each terminal is excited independently. The different types of configurations for independent excitation are discussed in 3.3. Of particular interest is the coupling equations for three–phase electric machines. Section 3.4 develops the external circuit equations for delta– and wye–connected three–phase machines.

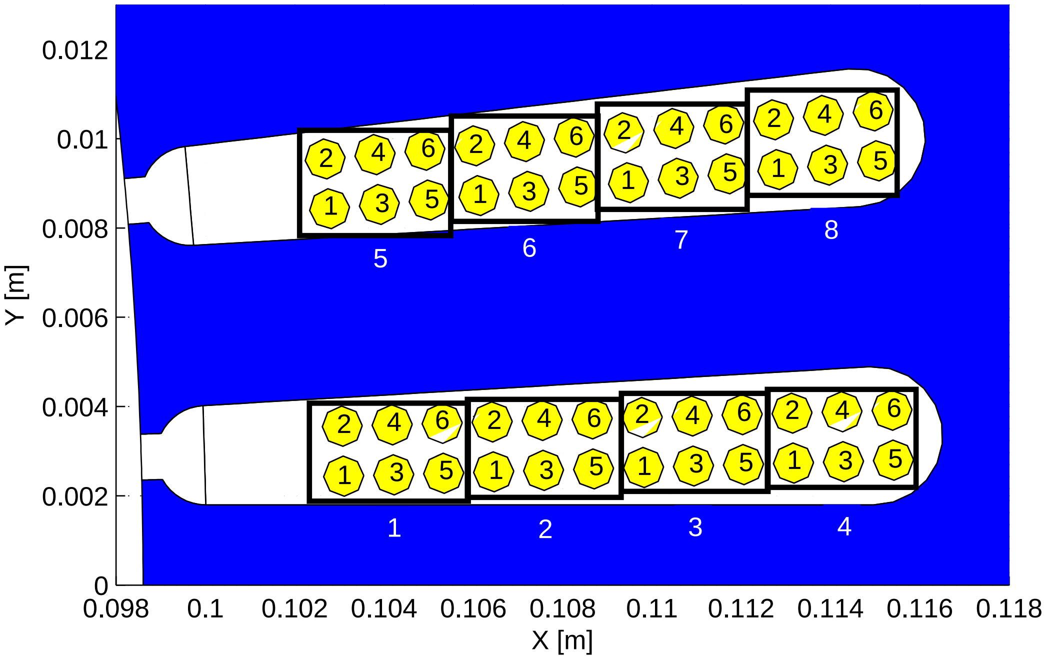

Figure 3.1: The depicted stator has two slots per phase, 4 turns per slot for a total of 8 turns per pole, and 6 strands per turn. The unique strands are labeled 1 through 6. The turns in these two slots are labeled 1 through 8.

3.1 Dynamic Field–Circuit Coupling

The details of electric machine windings can be fairly complex. The two slots of a stator depicted in Fig. 3.1 will help to clarify the terminology. Each winding can be composed of several bundles of wires connected in parallel. The individual wires composing a bundle will be referred to as strands. In the two–dimensional approximation, each bundle passes through the modeled cross section multiple times. We will refer to each intersection of a bundle with the cross section as a turn. The regions within each turn associated with the same strand must carry the same total current. The sign of the current depends on the direction of the strand through the machine.

Let us assume that there are $1 \leq i \leq N_b$ bundles, $1 \leq j \leq N_s\left(i\right)$ strands per bundle, and $N_t\left(i\right)$ turns per bundle. This means that there are $1 \leq k \leq N_c$ regions Ω_k in the cross section of the machine that conductors pass through, where

$$N_c = \sum_{i=1}^{N_b} N_s\left(i\right) N_t\left(i\right). \tag{3.1}$$

The connectivity of strand (i, j) is defined by the subset of regions Ω_k in the cross section that the strand passes through. We introduce the path sets $\mathbb{P}_{i,j}$ that contain a set of tuples (k, s) when strand (i, j) passes through region Ω_k, with $s = \pm 1$ depending on the direction of the strand through the machine.

With this notation, the total strand current is $i_{i,j}$. The strand current constraints can be written in terms of the field variables as

$$\int_{\Omega_k} \sigma \frac{\partial A}{\partial t} d\Omega + \sigma |\Omega_k| E_{\phi,k} + s i_{i,j} = 0, \quad \forall \left(k, s\right) \in \mathbb{P}_{i,j}, \quad \forall i, j. \tag{3.2}$$

Because the strands within each bundled are connected in parallel, the total voltage drop across each strand is the same as the voltage drop across the bundle ($v_{i,j} = v_i$). To account for the voltage drop experienced by the portion of the strand not in the cross section of the machine, i.e. within the "end turns", an external resistance $R_{i,j}$ and inductance $L_{i,j}$ are introduced.[1] Therefore, we consider the following set of equations for the bundle voltages:

$$\sum_{\mathbb{P}_{i,j}} s E_{\phi,k} + \frac{R_{i,j}}{l_s} i_{i,j} + \frac{L_{i,j}}{l_s} \frac{\partial i_{i,j}}{\partial t} + \frac{1}{l_s} v_i = 0, \quad \forall j \tag{3.3}$$

where $E_{\phi,k}$ is the voltage drop per unit length across the part of the strand associated with region Ω_k.The stack length l_s corresponds to the part of the length of the machine in the z–direction that is occupied by magnetic laminations. The total terminal current i_i for bundle i is the sum of the strand currents:

$$\sum_j i_{i,j} = i_i. \tag{3.4}$$

[1] We could also consider mutual external inductances $M_{i,j}$ that induces a voltage in strand i from a time-varying current in strand j, but this is rarely done in practice.

Because there is a factor of $\frac{1}{l_s}$ multiplying v_i in (3.3), we will also multiply the current constraint (3.4) by the same constant:

$$\frac{1}{l_s}\sum_j i_{i,j} = \frac{1}{l_s} i_i. \tag{3.5}$$

The reason for doing so is that, once the equations are put into matrix form, this row multiplication is necessary to produce a symmetric matrix representation of the system of equations.

When only a fraction of the electric machine is modeled, the sum in (3.3) must be expressed in terms of only the turns that have been modeled. This is complicated by the fact that an individual strand may pass through a region of an adjacent pole that does not correspond to a strict rotation from the original pole. Often, this is done on purpose; certain clever choices of strand and turn paths can produce more desirable current density and magnetic field distributions.

Assume that a fraction f_p of the machine has been modeled and let the constant $s_p = \pm 1$ be defined as in (2.59) where $s_p = 1$ when periodic (p even) or $s_p = -1$ when antiperiodic (p odd) boundary conditions are assumed. We will call this p^{th} of the machine the master model. We will number Ω_k in a uniform way, such that if Ω_k is rotated by $2\pi f_p$, then the resulting region is identical to $\Omega_{k+f_pN_c}$. In terms of the fields, the (anti)periodicity assumption implies that the fields in Ω_k and $\Omega_{k+f_pN_c}$ are identical to within a change of sign. In terms of the path sets $\mathbb{P}_{i,j}$, the (anti)periodicity assumption implies that the tuple (k, s) can be mapped to a tuple corresponding to a region contained in the master model as

$$(k, s) \leftarrow \left(k_f, s_p^{k_s} s\right), \tag{3.6}$$

where

$$\begin{aligned} k_f &= \texttt{modulo}\left(k-1, f_p N_c\right)+1, \\ k_s &= \left\lfloor \frac{k-1}{f_p N_c} \right\rfloor. \end{aligned} \tag{3.7}$$

The value of k_f accounts for the mapping of the region to the master model. The value of k_s accounts for the possibility of a difference in the polarity of the source and destination region. We will write $(k, s) \in \hat{\mathbb{P}}_{i,j}$ to signify the (anti)periodic path sets obtained by applying (3.6) to each tuple in $\mathbb{P}_{i,j}$. Because different values of k and s can be mapped to the same value of k_p and k_s, it is possible that the same value of k can appear multiple times in a given $\hat{\mathbb{P}}_{i,j}$ and in multiple path sets with distinct j.

We introduce the region set $\mathbb{R}_{i,m}$ associated with the current $i_{i,m}$ and voltage drop per unit length $E_{\phi,m}$ such that (k, s) appears in $\mathbb{R}_{i,m}$ once and only once if the region k carries current $i_{i,m}$. We also introduce the strand set $\mathbb{S}_{i,m}$ such that $j \in \mathbb{S}_{i,m}$ if there is some (k, s) such that $(k, s) \in \hat{\mathbb{P}}_{i,j}$ and $(k, s) \in \hat{\mathbb{P}}_{i,l}$ for $j \neq l$. The (anti)periodicity assumption implies that any pair of distinct strands containing the same region in their path set must carry the same total current and must experience the same voltage drop per unit length within that region. Therefore, if two distinct strands j and l are in the same strand set $\mathbb{S}_{i,m}$, the corresponding region set $\mathbb{R}_{i,m}$ must contain all unique tuples (k, s) from both $\hat{\mathbb{P}}_{i,j}$ and $\hat{\mathbb{P}}_{i,l}$. The strand sets and the region sets are duals in the sense that $\mathbb{S}_{i,m}$ indexes the strands and $\mathbb{R}_{i,m}$ indexes the regions associated with the current $i_{i,m}$. The number of region/strand sets corresponds to the number of unique strand currents. Now the strand current constraints are written in terms of the region sets:

$$\int_{\Omega_k} \sigma \frac{\partial A}{\partial t} d\Omega + \sigma |\Omega_k| E_{\phi,k} + s i_{i,m} = 0, \quad \forall \left(k, s\right) \in \mathbb{R}_{i,m}, \quad \forall i, m. \tag{3.8}$$

There are several ways to derive the bundle voltage drop. We write it in terms of the sum of $E_{\phi,m}$ over the region sets $\mathbb{R}_{i,m}$, which will realize a set of symmetric equations later on. The sum of $s l_s E_{\phi,k}$ over $\mathbb{R}_{i,m}$ corresponds to the voltage drop experienced by the bundle over

f_p of the path, multiplied by the number of individual strands $|\mathbb{S}_{i,m}|$ that carry the current $i_{i,m}$. Therefore, we must introduce the scaling factor $|\mathbb{S}_{i,m}|f_p$ to the bundle voltage equation:

$$\sum_{\mathbb{R}_{i,m}} sE_{\phi,k} + \frac{f_p|\mathbb{S}_{i,m}|R_{i,m}}{l_s} i_{i,m} + \frac{f_p|\mathbb{S}_{i,m}|L_{i,m}}{l_s}\frac{\partial i_{i,m}}{\partial t} + \frac{f_p|\mathbb{S}_{i,m}|}{l_s} v_i = 0, \quad \forall i, m. \tag{3.9}$$

Because of the (anti)periodicity constraints, we must have

$$R_{i,m} = R_{i,j}, \quad L_{i,m} = L_{i,j}, \quad \forall j \in \mathbb{S}_{i,m}. \tag{3.10}$$

The terminal currents can be expressed as a sum of the strand sets $\mathbb{S}_{i,m}$ after taking into account their multiplicity:

$$\sum_{m} \frac{f_p|\mathbb{S}_{i,m}|}{l_s} i_{i,m} = \frac{f_p}{l_s} i_i, \quad \forall i, \tag{3.11}$$

where we have introduce the factor f_p for symmetry purposes.

3.1.1 Periodic and Antiperiodic Boundary Conditions

From a computational efficiency standpoint, we want to chose $f_p = \frac{p}{N_p}$ as small as possible. In the two–dimensional model, if the geometry and permanent magnet excitation alone allows us to choose a value of p that corresponds to a valid mapping of the entire machine to the master model, there is nothing inherent in the field structure that disallows the use of that value of p and the corresponding (anti)periodic boundary conditions. The situation changes slightly when we consider the effects of end turns. It is entirely possible, even likely, that different strands in the same bundle have different lengths. The only way to capture this effect in two dimensions is by choosing different external resistance and inductance values $R_{i,j}$ and $L_{i,j}$ for each strand. However, each p implicitly places constraints on the value of the strand external impedance through (3.10). Therefore, if we have the geometry of the problem and a set of external strand impedances at hand, the best choice of p that is strictly valid is the smallest one dictated by the geometry that does not violate (3.10).

That being said, it is not an easy task to specify individual values for the strand impedances, especially the value of the inductance. Further, the external impedance only comes into play in the development of the bundle voltage due to end turn effects. For realistic impedance values, the bundle voltage drop will be dominated by the contribution from $E_{\phi,k}$ and, in particular, the non–resistive voltage drop due to Faraday's law. Therefore, from a modeling perspective, using a single mean value for the external impedance values may be acceptable. From a computational efficiency standpoint, this is desirable as it allows us to model the smallest part of the machine dictated by the geometry.

3.1.2 Macroscopic Strand Modeling

Often, the details of the strands are unimportant and modeling them can dramatically increase the computational burden. If the characteristic time τ is much smaller than the magnetic diffusion time τ_m, with the diameter of the strand playing the role of the characteristic length λ, then it may be reasonable to assume that $E_{\phi,k}$ is the same for each strand within a turn and that the magnetic vector potential is nearly uniform across the turn. In this case we may consider replacing the individual strands with one macroscopic strand so that each bundle (and turn) is modeled with a single massive conductor occupying the domain $\Omega_{\hat{k}}$. The individual values of $E_{\phi,k}$ for each strand are replaced with a single value $E_{\phi,\hat{k}}$.

In order to accurately capture the mean strand behavior, the macroscopic domain $\Omega_{\hat{k}}$ should be situation at some nominal center (x_0, y_0) of the strands and be roughly the same shape as the arrangement of the individual strands. Two possible situations can arise. If the strands are semi–regularly distributed so that the macroscopic strand may have a simple geometry, we can require the area of $\Omega_{\hat{k}}$ be equal to the total cross sectional area of the strands in the bundle. As a result, approximately the same net current is calculated in the macroscopic strand as in the original stranded bundle.

Because we are interested in capturing the mean turn behavior, another logical proposition is to require that $\Omega_k \in \hat{\Omega}_{\hat{k}}$ for each strand in the turn. This will almost surely result in

$\Omega_{\hat{k}}$ having a cross sectional area that is greater than the sum of the individual turns, an effect that must be compensated for in the current constraint equations. To do this, we assume $A \approx A_0$ and introduce the factor $f_{p,s}$ so that the integral over the macroscopic strand and the sum of the integrals over the individual strands vanish to 0^{th} order:

$$\int_{\Omega_{\hat{k}}} f_{p,s}\sigma \frac{\partial A_0}{\partial t} d\Omega + f_{p,s}\sigma |\Omega_{\hat{k}}| E_{\phi,\hat{k}} = \sum_k \left[\int_{\Omega_k} \sigma \frac{\partial A_0}{\partial t} d\Omega + \sigma |\Omega_k| E_{\phi,k} \right]. \tag{3.12}$$

Because A_0 is spatially constant and we must have $E_{\phi,\hat{k}} = E_{\phi,k}$ over the turn, $f_{p,s}$ must be the ratio of the total strand area to the macroscopic strand area:

$$f_{p,s} = \frac{\sum_k |\Omega_k|}{|\Omega_{\hat{k}}|}. \tag{3.13}$$

A more subtle effect that must be considered is the change in characteristic magnetic diffusion time associated with each turn. Calculating τ_m for a stranded turn is complicated by the fact that the conductivity is discontinuous across the area of the turn. That is, $\sigma = 0$ in the areas outside of the individual strands. Because $\tau_m = 0$ in non–conductors, we can think of replacing σ with an effective conductivity $f_{p,s}\sigma$ where $f_{p,s}$ is the fraction of conducting material in the turn.

In the case where the macroscopic strand area is equal to the sum of the area of the strands, we have $f_{p,s}\lambda^2_{strand} \approx \lambda^2_{macro}$ and the magnetic diffusion times end up being roughly equivalent. In the case where the turn and the macroscopic strand have the same characteristic length, $\lambda_{strand} = \lambda_{macro}$. To adjust the diffusion time of the macroscopic strand, using the value of $f_{p,s}\sigma$ as the conductivity will produce the same characteristic magnetic diffusion time as the stranded turn. Evidently, after consideration of (3.12), it is simply a matter of making the modification $\sigma \leftarrow f_{p,s}\sigma$ to realize a 0^{th} order correction in the field and current equations. From this point on, we will drop $f_{p,s}$ from the notation with the understanding that modifications to the material conductivity are necessary in certain situations when ignoring individual strand behavior.

3.2 Static Field–Circuit Coupling

It may happen that the characteristic length of a turn is small enough that the resulting magnetic diffusion time is much shorter than the characteristic time of interest. When this occurs, it can be useful to replace all of the turns in a localized area with a single massive conductor and adjust the field constraints correspondingly. When this happened with the individual strands in the turns, we found that the magnetic diffusion times τ_m were additive since each strand was connected in parallel. Because the turns are connected in series, the magnetic diffusion time associated with the bundle is the same as an individual turn. Similarly, whereas each strand may carry a different current, each turn, being composed of the individual strands, must carry the same current. The assumption that τ_m associated with the turn is smaller than the characteristic time implies that no significant current redistribution occurs due to magnetic diffusion. Therefore, multiple turns carrying the same current throughout a given region produce a roughly uniform current distribution.

To clarify this, we will retain the same bundle numbering $1 \leq i \leq N_b$ as in the previous section. We will assume the turns within each bundle can be grouped into $1 \leq j \leq N_r\left(i\right)$ regions. In analogy with the preceding section, we will denote each distinct region as Ω_k. Now we introduce turn–sets $\mathbb{T}_i$ that contains the tuple (k, s) when Ω_k is a region containing N_k turns of bundle i with $s = \pm 1$ depending on the direction of the turns. If the bundle is modeled as a set of stranded conductors with more than one strand per bundle, N_k is taken to be the number of strands per bundle, $\frac{1}{N_s(i)}$. The number of turns must satisfy

$$N_t\left(i\right) = \sum_{\mathbb{T}_i} N_k. \tag{3.14}$$

In order to decouple Ampere's law and Faraday's law, the bundle current i_i acts as an input to field equations producing a spatially uniform current density in each region:

$$-\sigma\frac{\partial A}{\partial t}-\sigma E_{\phi,k}=\frac{sN_k}{|\Omega_k|}i_i, \quad \forall\,(s,k)\in\mathbb{T}_i, \quad \forall i. \tag{3.15}$$

Using this expression for the current density gives us Ampere's law for magnetostatics:

$$\nu_0\nabla'\times\nabla\times A-\nabla'\times\mathbf{M}=J, \tag{3.16}$$

where

$$J=\begin{cases}\frac{sN_k}{|\Omega_k|}i_i, \;\; (s,k)\in\mathbb{T}_i & \text{if } \;(x,y)\in\Omega_k \\ 0 & \text{otherwise}\end{cases}. \tag{3.17}$$

Faraday's law enters through the external circuit constraints where the voltage per unit length is calculated from A and i. Even though the current density is constant, the value of A calculated from (3.15) still varies spatially. Physically, the constant value taken by E_ϕ should satisfy the constraints placed on the total current flowing through each Ω_k as dictated by (3.15). Mathematically, this results in an averaging equation that calculates $E_{\phi,k}$ to fulfill this constraint:

$$E_{\phi,k}=-\frac{1}{|\Omega_k|}\int_{\Omega_k}\frac{\partial A}{\partial t}d\Omega-\frac{sN_k}{\sigma|\Omega_k|}i_i, \quad \forall(k,s)\in\mathbb{T}_i, \quad \forall i. \tag{3.18}$$

The total bundle voltage is the sum of the $E_{\phi,k}$ associated with each bundle, multiplied by the number of turns per region and the stack length, plus the sum of an external resistive and inductive voltage drops due to the bundle current:

$$-\sum_{\mathbb{T}_i}sN_kE_{\phi,k}+\frac{R_i}{l_s}i_i+\frac{L_i}{l_s}\frac{\partial i_i}{\partial t}+\frac{1}{l_s}v_i=0. \tag{3.19}$$

The difference between the sign of the $E_{\phi,k}$ component of (3.3) and (3.19) can be explained in the following way. In (3.3), the voltage across the model is considered a reaction to the applied voltage v_i and the model acts as a load to the source. In (3.19), we have specified the current density through (3.15) and the voltage drop across the model is a reaction to the load attached to the terminals. That is to say, the model acts like a current source attached to a load. The load itself may include additional independent sources. The passive sign convention dictates that the polarity of the voltage must be defined so the current flows into the positive terminal for (3.3) and into the negative terminal for (3.19). Because the direction of the current is fixed in the positive z–direction for both cases, the polarity of the voltage must be reversed between the two equations.

After substitution of the expression for $E_{\phi,k}$, the bundle voltage equation becomes

$$-\sum_{\mathbb{T}_i} \frac{sN_k}{|\Omega_k|} \int_{\Omega_k} \frac{\partial A}{\partial t} d\Omega - \sum_{\mathbb{T}_i} \frac{N_k^2}{\sigma|\Omega_k|} i_i - \frac{R_i}{l_s} i_i - \frac{L_i}{l_s} \frac{\partial i_i}{\partial t} = \frac{1}{l_s} v_i, \tag{3.20}$$

where we have multiplied through by -1 for symmetry. When only f_p of a machine is modeled, the sum over $\mathbb{T}_i$ represents only f_p of the total voltage drop, and the equation must be adjusted accordingly:

$$-\sum_{\mathbb{T}_i} \frac{sN_k}{|\Omega_k|} \int_{\Omega_k} \frac{\partial A}{\partial t} d\Omega - \sum_{\mathbb{T}_i} \frac{N_k^2}{\sigma|\Omega_k|} i_i - \frac{f_p R_i}{l_s} i_i - \frac{f_p L_i}{l_s} \frac{\partial i_i}{\partial t} = \frac{f_p}{l_s} v_i. \tag{3.21}$$

3.3 Field–Circuit Equations: Independent Excitation

From the proceeding sections, we can identify four cases of interest divided by excitation and constraint type. In this section, we summarize these cases when each bundle is excited independently. The case where the bundle excitation is constrained by a three-phase external circuit is discussed in the next section.

3.3.1 Dynamic Simulation with Voltage Input

This type of simulation includes the effects of magnetic diffusion using a stranded or macroscopic-strand model. The inputs are the bundle voltages v_i in (3.9). The field equations are coupled through the strand current constraints in (3.8).

3.3.2 Dynamic Simulation with Current Input

Magnetic diffusion is included in this type of simulation as well. The inputs are the terminal currents i_i in (3.11). The bundle voltage and individual strand currents are determined dynamically through (3.9). As in the voltage input case, the fields are coupled to the source through the strand currents using (3.8).

3.3.3 Pseudostatic Simulation with Voltage Input

Magnetic diffusion is ignored in this type of simulation. Faraday's law is introduce in order to include the voltage induced by the time–varying flux coupling the turns. The field equations are given by (3.16). The inputs from the source are the v_i in equation (3.21).

3.3.4 Static Simulation with Current Input

Magnetic diffusion is ignored again using (3.16), and dynamics only enter into the picture through the circuit variables. The terminal currents i_i act as the inputs. In this case, the fields represented by (3.16) can be simulated independently of the external circuits. Equation (3.21) is used for an after–the–fact determination of the terminal voltages v_i.

3.3.5 Summary

Table 3.1 summarizes the different types of simulation models that arise for different input sources and assumptions on the effects of magnetic diffusion. Different bundles may employ

Table 3.1: Summary of Simulation Models for Independent Bundle Excitation

Simulation Type	Field Equation	Circuit Equations	Input	Unknowns
Dynamic, Voltage Driven	(2.49)	(3.8), (3.9)	v_i	A, $E_{\phi,m}$, $i_{i,m}$
Dynamic, Current Driven	(2.49)	(3.8), (3.9), (3.11)	i_i	A, $E_{\phi,m}$, $i_{i,m}$, v_i
Pseudostatic, Voltage Driven	(3.16)	(3.21)	v_i	A, i_i
Static, Current Driven	(3.16)		i_i	A

different models, leading to a mixture of simulation types within a given model. The case of static, current driven problems are special in that no dynamics are included in the simulation at all.

3.4 Idealized Three-Phase Source Coupling

This section discusses the important case of machine windings excited by an ideal three–phase voltage or current source. There are several ways of modeling this problem. In a physical setting, the true source would likely be a voltage source inverter, which is most closely related to three ideal voltage sources in a grounded–wye configuration. The voltage source inverter is made to appear as a current source through feedback regulation, a configuration which can be modeled as three ideal current sources, also in a grounded–wye configuration. This is the approach we take.

There is not a unique way to couple the bundles to a given source. Our approach to coupling is focused first on maintaining symmetry of the equations, and secondly on modularity. With respect to modularity, we would like to write the coupling equations in terms of the bundle voltages v_i and bundle currents i_i. In this way, we decouple the modeling of the bundles from modeling of the external circuits. This allows us to develop simpler and more flexible software. The price paid is that the number of external circuit equations is not necessarily minimum. These additional equations amount to a negligible increase when considering the total number of equations resulting from the discretization of the partial differential equation.

The three–phase voltage source will be modeled as three ideal voltage sources in a grounded–wye configuration with identical R–L source impedances between an ideal source and corresponding phase bundle. The three–phase current source will be modeled as three ideal current sources in a grounded–wye configuration. It is common to see three-phase current sources modeled as three ideal sources in a delta configuration due to the possibility of a common mode current during unbalanced operation. In practice, feedback control is used to endow the voltage source inverter with current source type behavior, so the wye configuration seems more appropriate. It is for this reason that we ignore the source impedance of the three–phase current source; it must only be considered if we are to work backward to find the equivalent voltage source waveforms. We should take care to only specify balanced three-phase currents when modeling the current source attached to an ungrounded–wye connected load, lest Kirchoff's current law is violated at the common node of the phases.

3.4.1 Ungrounded–Wye Connected Load with Voltage Source Excitation

Consider three bundles in an ungrounded–wye configuration with phase voltages v_i and line currents i_i. The input is specified as line–to–neutral voltages $v_{s,i}$ and the source currents equal the bundle currents, $i_i = i_{s,i}$. In the ungrounded configuration, we must allow for the possibility of a common mode voltage v_0 at the node shared by the bundles. The bundle current equals the line current, so we can write the voltage equation using Kirchoff's current law once for each phase:

$$-v_i + R_{s,i} i_i + L_{s,i} \frac{\partial i_i}{\partial t} + v_0 = -v_{s,i}. \tag{3.22}$$

Because we have introduced an extra variable, v_0, one extra equation is required. The calculation of v_0 enters through the constraint placed on the common mode current in the ungrounded–wye configuration, namely

$$\sum_i i_i = 0. \tag{3.23}$$

Notice that v_0 does not show up explicitly in (3.23). In fact, we could add an external capacitance and resistance between the common node of the phases and the ground, which would introduce additional current terms $C_0 \frac{\partial v_0}{\partial t}$ and $\frac{v_0}{R_0}$ on the right-hand side of (3.23). If we ignore the effects of the common–mode capacitance, the external circuit parameters naturally take the values of $C_0 = 0$ and $R_0 = \infty$. Finally, we introduce the scaling factor $\frac{f_p}{l_s}$ for symmetry purposes, which leaves us with the final form of the equations:

$$-\frac{f_p}{l_s} v_i + \frac{f_p}{l_s} R_{s,i} i_i + \frac{f_p}{l_s} L_{s,i} \frac{\partial i_i}{\partial t} - \frac{f_p}{l_s} v_0 = -\frac{f_p}{l_s} v_{s,i}. \tag{3.24}$$

$$-\frac{f_p}{l_s} \sum_i i_i = 0. \tag{3.25}$$

To simulate ungrounded wye-connected phases with voltage source excitation, we append equations (3.24) and (3.25) to those for the independently excited current models from Section 3.3.2 for dynamic simulations. For pseudostatic simulations, we append them to the equations of Section 3.3.3.

3.4.2 Ungrounded–Wye Connected Load with Current Source Excitation

This is perhaps the simplest case. The current source specifies the line currents, which are equal to the bundle currents when the phases are in a wye configuration. This is equivalent to independent excitation of the bundles, with the caveat that the common mode current must be zero. Therefore, this behavior is achieved using the same set of equations as in Section 3.3.2 for dynamic simulations or 3.3.4 for static simulations. The source impedance is only necessary if an after the fact determination of the source voltage is desired.

3.4.3 Delta Connected Load with Voltage Source Excitation

In this instance, the bundle voltage drop is the difference between the corresponding source line–to–neutral voltages:

$$-v_i + v_{s,i} - v_{s,i-1} = 0. \tag{3.26}$$

The source line currents are the difference between the corresponding bundle currents:

$$i_i - i_{i+1} - i_{s,i} = 0. \tag{3.27}$$

Finally, the line–to–neutral voltages of the source can be expressed in terms of the terminal voltages and the voltage drop due to the source impedance:

$$-v_{s,i} + R_{s,i} i_{s,i} + L_{s,i} \frac{\partial i_{s,i}}{\partial t} = -v_{s,i}. \tag{3.28}$$

Notice that no constraints on the common mode current or voltage appear explicitly in the above equations. In fact, there is no constraint on the common mode current for the bundle loop. The constraint that the common mode voltage of the bundles must be zero is implied by (3.26), which can be seen by taking the summation over i:

$$\sum_i v_i = (v_{s,1} - v_{s,3}) + (v_{s,2} - v_{s,1}) + (v_{s,3} - v_{s,2}) = 0. \tag{3.29}$$

As always, the equations must be scaled for symmetry:

$$-\frac{f_p}{l_s} v_i + \frac{f_p}{l_s} v_{s,i} - \frac{f_p}{l_s} v_{s,i-1} = 0, \tag{3.30}$$

$$\frac{f_p}{l_s} i_i - \frac{f_p}{l_s} i_{i+1} - \frac{f_p}{l_s} i_{s,i} = 0, \tag{3.31}$$

$$-\frac{f_p}{l_s} v_{s,i} + \frac{f_p R_{s,i}}{l_s} i_{s,i} + \frac{f_p L_{s,i}}{l_s} \frac{\partial i_{s,i}}{\partial t} = -\frac{f_p}{l_s} v_{s,i}. \tag{3.32}$$

To perform dynamic simulations of delta connected bundles with voltage source excitation, equations (3.30)–(3.32) must be appended to those for the independently excited case in Section 3.3.2. For pseudostatic simulations, as in the wye connected case, we append them to the equations specified in Section 3.3.3.

3.4.4 Delta Connected Load with Current Source Excitation

In this case, the source line currents are specified. Because the phases are in a delta connection, the specified currents are not in unique correspondence with the bundle currents due to the possibility of a common mode current. The phase currents, common mode currents, and line currents are related through the following:

$$i_i - i_0 = \frac{1}{3}(i_{s,i} - i_{s,i-1}). \tag{3.33}$$

When $i_0 = 0$, the source currents uniquely determine the bundle currents. In fact, this situation can be modeled by the equations for independent bundle excitation.

When the common mode current is not equal to zero, the phases cannot generally be considered independently excited. The line currents do not uniquely specify the phase currents in this case. Instead, the line currents specify the difference between the currents of adjacent phases. The common mode current is determined dynamically as the result of a possible phase imbalance. Since we are mostly interested in balanced three-phase operation, we will ignore this effect and model the phases as independently excited by setting

$$\frac{f_p}{l_s} i_i = \frac{f_p}{3l_s}(i_{s,i} - i_{s,i-1}) \tag{3.34}$$

in the appropriate equations.

Chapter 4

Finite-Element Analysis

A general method for simulating electric machines must be capable of solving Maxwell's equations in their most general form including an accurate representation of the underlying geometry. Finite–element analysis is one such technique. The finite–element method is a general procedure for solving partial differential equations on geometrically complex domains by approximating the solution in a vector space spanned by a particular set of basis functions. In a mathematically rigorous treatment, the basis functions belong to a particular Sobolev space, which is a Hilbert space equipped with an inner product dependent upon the differential operator being analyzed [107]. In an intuitive sense, the basis functions interpolate the field and potential variables over the computational domain while maintaining essential continuity/discontinuity properties across material interfaces. The basis functions are defined to have support on a small number of non-overlapping, connected sub–domains contained within the overall computational domain. Each sub–domain is referred to as an element, and the collection of elements is called a mesh or tessellation. These properties of the elements and basis functions lead directly to many attractive computational properties and usage of the term finite–element analysis.

Because the mesh may not (and usually does not) exactly match the underlying continuous geometry, the discretization must be taken into account in all aspects of the problem. The first difficulty is associated with modeling airgaps, as presented in Section 2.7, since the discretized boundaries are no longer perfectly cylindrical. The resolution of this problem using a small–angle approximation and projection between subspaces spanned by finite–element basis functions and trigonometric basis functions is presented in Section 4.5.2. Because the field equations are coupled through external circuits as described in Chapter 3, the discretization process affects these relationships as well.

4.1 Master and Mapped Elements

There are different equivalent ways of thinking about discretizing a problem using finite–elements. From a procedural point of view, the discretization process can be thought of as constructing a tessellation of the underlying geometry using a set of elements that are topologically equivalent to a very small set of simple shapes (triangles, quadrilaterals, etc). Unless great care is taken, a mesh typically only approximates the underlying geometry wherever there is a material discontinuity. The approximation of the geometry can be improved by decreasing the size of the elements.

To describe mathematically what an element looks like, we typically assume that all of the elements in a mesh can be written as a transformation of a limited set of master elements. We will assume only one master element, the master triangle:

$$\Omega_T = \{\eta \ s.t. \ 0 < \eta < 1, \ 0 < \zeta < 1, \ 0 < \eta + \zeta < 1\}, \tag{4.1}$$

where $\eta = (\eta, \zeta)$ is the coordinate vector in the master element space. For reference, see the left side of Fig. 4.1. Given the master element, we assume each element Ω_i in a mesh can be written as

$$\Omega_i = \mathfrak{f}_i\left(\Omega_T\right) = \{\mathbf{x} = \mathfrak{f}_i\left(\eta\right) \ s.t. \ \eta \in \Omega_T\} \tag{4.2}$$

where $\mathbf{x} = (x, y)$ is the coordinate vector in the physical space. The function $\mathfrak{f}_i$ is a differ-

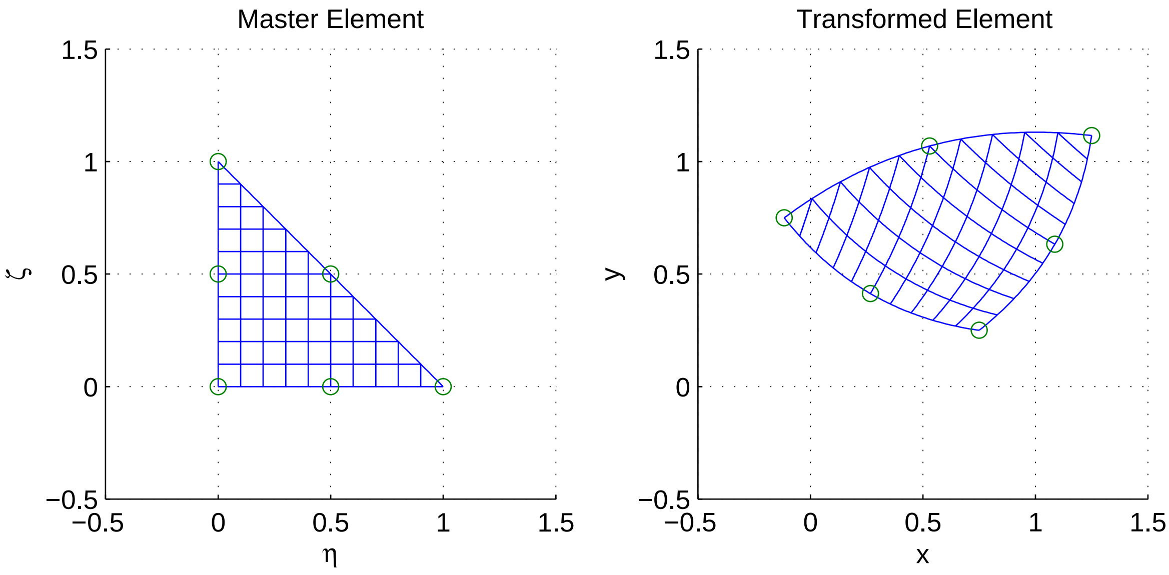

Figure 4.1: Master triangular element (left) and an example transformed element (right).

entiable mapping of the master element onto the transformed element with a nonsingular Jacobian $\bar{F}_i$ on Ω_T, i.e.,

$$\det \bar{F}_i \neq 0, \quad \forall \eta \in \Omega_T, \tag{4.3}$$

where

$$\bar{F}_i = \nabla_\eta \mathfrak{f}_i = \begin{bmatrix} \frac{\partial \mathfrak{f}_x}{\partial \eta} & \frac{\partial \mathfrak{f}_y}{\partial \eta} \\ \frac{\partial \mathfrak{f}_x}{\partial \zeta} & \frac{\partial \mathfrak{f}_y}{\partial \zeta} \end{bmatrix} \tag{4.4}$$

This is equivalent to requiring that the determinant of the Jacobian does not change sign in Ω_T. Intuitively, these requirements prevent the mapping from introducing "knots" into the elements. An example of a transformed element is given on the right side Fig. 4.1. To ease the notational burden, we drop the subscript i when there is no confusion in doing so, understanding that $\mathfrak{f}$ depends on the particular element under consideration.

4.2 Master Element Basis Functions

The choice of master element(s) and mappings $\mathfrak{f}$ determine the approximation of the geometry in the finite-element method. The choice of basis functions determine the approximation of the solution of the partial differential equation on the approximated geometry. One of the simplest procedures is to define a set of basis functions on each master element and use the mappings $\mathfrak{f}$ to define the corresponding basis functions on each element. On the master triangle Ω_T, the first-order Lagrangian basis functions are defined as

$$\begin{aligned} \lambda_1 &= 1 - \zeta - \eta \\ \lambda_2 &= \zeta \\ \lambda_3 &= \eta \end{aligned} \tag{4.5}$$

They are called Lagrangian basis functions because, when combined as a weighted-sum, the weights represent an interpolation of the values occurring at the corners of the triangle. Higher-order Lagrangian basis functions exist as well. For example, the second-order basis functions can be written as follows:

$$\begin{aligned} \lambda_1 &= (1 - \zeta - \eta)(1 - 2\zeta - 2\eta) & \lambda_4 &= 4\zeta\eta \\ \lambda_2 &= 4\zeta(1 - \zeta - \eta) & \lambda_5 &= \eta\,(2\eta - 1) \\ \lambda_3 &= \zeta\,(2\zeta - 1) & \lambda_6 &= 4\eta(1 - \zeta - \eta). \end{aligned} \tag{4.6}$$

There are different forms of basis functions that span the same space. For example, it is possible to write a basis for the space spanned by the functions in (4.6) that includes the basis for the subspace in (4.5):

$$\begin{aligned} \lambda_1 &= 1-\zeta-\eta & \lambda_4 &= \zeta^2 \\ \lambda_2 &= \zeta & \lambda_5 &= \eta^2 \\ \lambda_3 &= \eta & \lambda_6 &= \zeta\eta. \end{aligned} \tag{4.7}$$

These are sometimes referred to as hierarchical basis functions and are used in polynomial adaptive solution refinement.

For the most part, this work is not concerned with the basis functions per se, so we shall assume the use of the first–order Lagrange elements when details about the basis functions are necessary or an example is instructive. In all other places, we will attempt to make our presentation as broadly applicable as possible. When referring to the basis functions in the physical space, we will use an additional subscript to indicate the particular element under consideration:

$$\lambda_{i,j} = \begin{cases} \lambda_j, & \mathbf{x} \in \Omega_i \\ 0, & \mathbf{x} \in \Omega_k,\ k \neq i \end{cases} \tag{4.8}$$

4.3 Inter–Element Continuity

So far, the basis functions have been discussed disjointly for each element in the mesh. The physics of the problem imply certain continuity conditions should be observed in a reasonable solution. For example, the magnetic vector potential $\mathbf{A}$ is know to be tangentially–continuous at material interfaces. In the Cartesian 2-dimensional approximation where $A = A_z$ is the unknown, the implied tangent is in the z–direction. Therefore, continuity of the approximation of A should be preserved by its expansion in terms of the piecewise continuous polynomial basis functions. This can be thought of as requiring a Dirchlet boundary condition for an individual element to be satisfied by its neighboring elements.

For the Lagrangian basis functions, this process is straightforward. Each basis function interpolates a unique point on the master element that corresponds to a point in physical space determined by the mapping function for each element. These interpolatory points are called nodes. The number of unique nodes determines the total number of degrees of freedom in the finite–element approximation. Two basis functions sharing the same node are identified with the same degree of freedom. Such a pair of basis functions are necessarily defined on different elements, since all basis functions within an element are associated with different nodes.

To make this precise, we assume the nodes of the tessellation are unique (and uniquely numbered) ordered pairs (x_i, y_i) . Node i has associated with it a set of globally numbered indices $\mathbb{N}_i$ corresponding to the basis functions that interpolate the solution at that particular point:

$$\mathbb{N}_i = \left\{ (j,k) \middle| \lambda_{j,k} \left(x_i, y_i\right) = 1 \right\} \tag{4.9}$$

It is also useful to define two functions, $\mathbb{L}_k(i)$ and $\mathbb{G}_k(j)$, where

$$\mathbb{L}_k(i) = j, \quad s.t. \quad (j,k) \in \mathbb{N}_i, \tag{4.10}$$

and

$$\mathbb{G}_k(j) = i, \quad s.t. \quad (j,k) \in \mathbb{N}_i. \tag{4.11}$$

In words, $\mathbb{L}_k(i)$ returns the local node number of $\mathbb{N}_i$ in element k while $\mathbb{G}_k(j)$ returns the global node number of local node j in element k. These three constructs are mathematical representations of common data structures used in finite–element implementations.[1]

We define the shape functions associated with each degree of freedom as a linear combi-

[1] Similar constructs are possible for other forms of the basis functions, but one generally has to consider sets of basis functions that are equal when evaluated on element edges or facets in the physical space in order to impose the exact continuity requirements. The interpolatory nature of the nodal basis functions makes this procedure somewhat easier to understand.

nation of the interpolatory basis functions for a particular node.

$$\varphi_i = \sum_{(j,k)\in N_i} \lambda_{j,k} \tag{4.12}$$

The support of each shape function is the union of the support of the individual basis functions in the sum of (4.12):

$$\Omega_i^{\varphi} = \bigcup_{(j,k)\in \mathbb{N}_i} \Omega_k^{\lambda}. \tag{4.13}$$

We will denote the corresponding weights for each shape function as a_i. Our discretized approximation of the magnetic vector potential is given by

$$a\left(\mathbf{x}\right) = \sum_i a_i \varphi_i \left(\mathbf{x}\right). \tag{4.14}$$

4.4 Galerkin's Method

Multiplying by a weight function ψ_i, substitution of our discrete approximation of A, and using integration by parts leads to the method of weighted residuals for solving (2.49). Each weight function results in one equation. Therefore, we need an equal number of weight and shape functions to produce an equal number of equations and variables. Choosing the weight functions to be equal to the shape functions, $\psi_i = \varphi_i$, results in Galerkin's Method.

Before integrating by parts, equation (2.49) becomes

$$\sum_j \int_\Omega \varphi_i \varphi_j \sigma \frac{\partial a_j}{\partial t} + \varphi_i \nu_0 \nabla' \times \nabla \times \varphi_j a_j - \varphi_i \nabla' \times \mathbf{M} d\Omega + \int_{\Omega_i} \varphi_i \sigma E_\phi d\Omega = 0, \tag{4.15}$$

where we have considered E_ϕ as an input for now. The second two terms in the integral of (4.15) can be integrated by parts, using a variation on Green's first identity:

$$\begin{aligned} &\int_\Omega \varphi_i \nu_0 \nabla' \times \nabla \times \varphi_j a_j - \varphi_i \nabla' \times \mathbf{M} d\Omega \\ &= \int_\Omega \nu_0 \nabla \times \varphi_i \cdot \nabla \times \varphi_j a_j - \nabla \times \varphi_i \cdot \mathbf{M} d\Omega - \int_\Gamma \varphi_i \left(\nu_0 \nabla \times \varphi_j a_j - \mathbf{M}\right) \times \mathbf{n} d\Gamma \end{aligned} \tag{4.16}$$

The term $\nu_0 \nabla \times \varphi_j a_j - \mathbf{M}$ appearing on the right hand side will be recognized as the discrete 2–dimensional approximation of the magnetic field intensity $\mathbf{H}$. Using this relationship, the boundary integral can be rewritten as the integral of the tangential component of the field intensity:

$$\int_\Gamma \varphi_i \left(\nu_0 \nabla \times \varphi_j a_j - \mathbf{M}\right) \times \mathbf{n} d\Gamma = \int_\Gamma \varphi_i H_t d\Gamma. \tag{4.17}$$

This term represents the natural Neumann boundary conditions for the problem. It is this term that is used to couple the finite–element model to an analytical solution to Laplace's equation in a cylindrical annulus. Otherwise, this input to the discrete equation disappears for boundaries where periodic or magnetic insulation boundary conditions are applied.

The discrete equation in terms of the shape functions then becomes the following:

$$\begin{aligned} &\sum_j \left[\int_\Omega \sigma \varphi_i \varphi_j d\Omega \cdot \frac{\partial a_j}{\partial t} + \int_\Omega \nu_0 \nabla \times \varphi_i \cdot \nabla \times \varphi_j d\Omega \cdot a_j\right] \\ &\quad + \int_\Omega \nabla \times \varphi_i \cdot \mathbf{M} d\Omega + \int_\Omega \varphi_i \sigma E_\phi d\Omega - \int_\Gamma \varphi_i H_t d\Gamma = 0. \end{aligned} \tag{4.18}$$

For evaluation, it necessary to expand (4.18) in terms of a summation over all elements and basis functions in each element. Since each φ is defined as the union of several pair–wise disjoint basis functions, it is much simpler to write the contribution in from the k^{th} element to the i^{th} equation with local node number $m = \mathbb{G}_k(i)$ from the j^{th} degree of freedom with local node number $n = \mathbb{L}_k(j)$:

$$\begin{aligned} &\int_{\Omega_k} \sigma \lambda_{m,k} \lambda_{n,k} d\Omega \cdot \frac{\partial a_j}{\partial t} + \int_{\Omega_k} \nu_0 \nabla \times \lambda_{m,k} \cdot \nabla \times \lambda_{n,k} d\Omega \cdot a_j \\ &+ \int_{\Omega_k} \nabla \times \lambda_{m,k} \cdot \mathbf{M} d\Omega + \int_{\Omega_k} \lambda_{m,k} \sigma E_\phi d\Omega - \int_{\Gamma_k} \lambda_{m,k} H_t d\Gamma = 0 \end{aligned} \tag{4.19}$$

Notice that the integrals in the first two terms of (4.19) depend only on the basis functions. The integral in the third term depends on the nonlinear function $\mathbf{M}$. In the special case of first–order elements, the approximation of the flux density in an element is constant. If the magnetization is modeled as a function of $\mathbf{B}$, $\mathbf{M}$ will also be constant and can be moved outside of the integral. In all other cases, it is not possible to simplify this expression any further without great difficulty. These integrals also implicity depend on the domain mapping functions $\mathfrak{f}\left(\Omega_T\right)$, which make it difficult to evaluate them exactly unless all of the $\mathfrak{f}$ are limited to affine transformations.

The solution to this problem is the use numerical quadrature for evaluation of the integrals. For polynomial basis functions and elements limited to affine transformations of the master element, the minimum degree of the quadrature rule for computing the integrals of the first two terms exactly is prescribed. The rules determined by this sort of analysis can serve well in practice for more complex cases.

The quadrature rules consist of a set of points $\{\mathbf{p}_q\}$ and weights $\{w_q\}$ defined on the master element Ω_T. The integrals in (4.19), written in terms of $\mathbf{x}$ coordinates in the physical space, are transformed into integrals over Ω_T in η coordinate space. The differential volume element, when written in terms of the master element, depends on the determinant of the Jacobian of the transformation function:

$$d\Omega_{\mathbf{x}} = \left|\det \bar{F}\left(\mathbf{p}_q\right)\right| d\Omega_\eta. \tag{4.20}$$

When derivatives are involved in the integral, it is necessary to express the derivatives in terms of the η coordinates,

$$\nabla^\eta \times \lambda = \begin{bmatrix} +\frac{\partial\lambda}{\partial\zeta} \\ \\ -\frac{\partial\lambda}{\partial\eta} \end{bmatrix} = \begin{bmatrix} +\frac{\partial\lambda}{\partial x}\frac{\partial x}{\partial\zeta} + \frac{\partial\lambda}{\partial y}\frac{\partial y}{\partial\zeta} \\ \\ -\frac{\partial\lambda}{\partial x}\frac{\partial x}{\partial\eta} - \frac{\partial\lambda}{\partial y}\frac{\partial y}{\partial\eta} \end{bmatrix} = \begin{bmatrix} +\frac{\partial y}{\partial\zeta} & -\frac{\partial x}{\partial\zeta} \\ \\ -\frac{\partial y}{\partial\eta} & +\frac{\partial x}{\partial\eta} \end{bmatrix} \begin{bmatrix} +\frac{\partial\lambda}{\partial y} \\ \\ -\frac{\partial\lambda}{\partial x} \end{bmatrix} = \breve{F} \cdot \nabla^{\mathbf{x}} \times \lambda, \tag{4.21}$$

which gives the following expression for the curl of the basis functions in the physical coordinates:

$$\nabla^{\mathbf{x}} \times \lambda = \breve{F}^{-1} \cdot \left(\nabla^{\eta} \times \lambda\right). \tag{4.22}$$

The matrices $\breve{F}$ and $\bar{F}$ are related by a 90 degree rotation:

$$\breve{F} = R^T \bar{F} R \tag{4.23}$$

$$R = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{4.24}$$

Approximation of the integrals in (4.19) using numerical quadrature and the previously outlined transformations gives us the overall contribution to the finite–element matrices from each pair of basis functions in each element. The first integral is related to the material conductivity and dynamics of the problem, and the corresponding matrix will be called the conductivity matrix with elemental contributions given by

$$C^k_{i,j} = \sum_q w_q \sigma \lambda_m(\mathbf{p}_q) \lambda_n(\mathbf{p}_q) \left|\det \bar{F}_k\left(\mathbf{p}_q\right)\right|. \tag{4.25}$$

The second integral is related to the reluctivity of free space and the corresponding matrix will be called the reluctivity matrix K, with contributions given by

$$\begin{aligned} K^k_{i,j} &= \nu_0 \sum_q w_q \left(\breve{F}_k^{-1} \cdot \nabla \times \lambda_m(\mathbf{p}_q)\right) \cdot \left(\breve{F}_k^{-1} \cdot \nabla \times \lambda_n(\mathbf{p}_q)\right) \left|\det \bar{F}_k\left(\mathbf{p}_q\right)\right| \\ &= \nu_0 \sum_q w_q \left(\bar{F}_k^{-1} \cdot \nabla \lambda_m(\mathbf{p}_q)\right) \cdot \left(\bar{F}_k^{-1} \cdot \nabla \lambda_n(\mathbf{p}_q)\right) \left|\det \bar{F}_k\left(\mathbf{p}_q\right)\right|. \end{aligned} \tag{4.26}$$

In the second line of (4.26), we have used the unitary nature of R to remove it from the vector dot–product. The third integral represents the nonlinearity, permanent magnets, and any linear materials with relative permeability greater than one. Generally, we will refer to

g as the nonlinearity, with contributions given by:

$$\begin{aligned} g_i^k &= \sum_q w_q \left(\breve{F}_k^{-1} \cdot \nabla \times \lambda_m(\mathbf{p}_q) \right) \cdot \mathbf{M} \left| \det \bar{F}_k \left(\mathbf{p}_q \right) \right| \\ &= \sum_q w_q \left(\bar{F}_k^{-1} \cdot \nabla \lambda_m(\mathbf{p}_q) \right) \times \mathbf{M} \left| \det \bar{F}_k \left(\mathbf{p}_q \right) \right| \end{aligned} \tag{4.27}$$

In the second line of (4.27), we have used the fact that the dot–product with R results in taking the cross product of the two vectors. The fourth integral represents the forced conduction due to E_ϕ and will be called the forcing function:

$$f_i^k = \sum_q w_q \sigma \lambda_m(\mathbf{p}_q) E_\phi \left| \det \bar{F}_k \left(\mathbf{p}_q \right) \right| . \tag{4.28}$$

We leave the boundary integral term in its exact form since we will employ a different approximation in the next section,

$$h_i^k = - \int_{\Gamma_k} \lambda_{m,k} H_t d\Gamma. \tag{4.29}$$

The complete matrices and vectors are formed by summing up all the contributions. If we consider each contribution as a matrix with one nonzero entry in the (i, j) position, then we can write

$$C = \sum_{k,m,n} C_{i,j}^k \tag{4.30}$$

$$K = \sum_{k,m,n} K_{i,j}^k \tag{4.31}$$

$$g = \sum_{k,m} g_i^k \tag{4.32}$$

$$f = \sum_{k,m} f_i^k \tag{4.33}$$

$$h = \sum_{k,m} h_i^k \tag{4.34}$$

and the unknown vector with components a_i as simply a. The model equation is then written as

$$C\frac{\partial a}{\partial t} + Ka + g\left(a\right) + h\left(t\right) = f\left(t\right). \tag{4.35}$$

To obtain a static (non–dynamic) model from (4.35), one simply sets $C = 0$.

Later, Newton's method will be applied to (4.35) after it has been discretized in time and will require the linearization of g. We will write this as the matrix G;

$$G = \frac{\partial g}{\partial a}. \tag{4.36}$$

An elemental expression for G can be derived by calculating $G^k_{i,j} = \frac{\partial g^k_i}{\partial a_j}$ using (4.27). Applying the chain rule to $\mathbf{M}$ we find

$$\frac{\partial \mathbf{M}}{\partial a_j} = \frac{\partial \mathbf{M}}{\partial \mathbf{B}} \cdot \frac{\partial \mathbf{B}}{\partial a_j} = \frac{\partial \mathbf{M}}{\partial \mathbf{B}} \cdot \left(\breve{F}_k^{-1} \cdot \nabla \times \lambda_n(\mathbf{p}_q)\right) \tag{4.37}$$

where $\frac{\partial \mathbf{M}}{\partial \mathbf{B}}$ is the matrix

$$\frac{\partial \mathbf{M}}{\partial \mathbf{B}} = \begin{bmatrix} \frac{\partial M_x}{\partial B_x} & \frac{\partial M_x}{\partial B_y} \\ \frac{\partial M_y}{\partial B_x} & \frac{\partial M_y}{\partial B_y} \end{bmatrix} \tag{4.38}$$

evaluated at the quadrature point $\mathbf{p}_q$. The expression $\frac{\partial \mathbf{B}}{\partial a_j}$ represents the contribution to the flux density due to a_j in element k at quadrature point $\mathbf{p}_q$. This gives the following value for $G^k_{i,j}$,

$$\begin{aligned} G^k_{i,j} &= \sum_q w_q \left(\breve{F}_k^{-1} \cdot \nabla \times \lambda_m(\mathbf{p}_q)\right) \cdot \frac{\partial \mathbf{M}}{\partial \mathbf{B}} \cdot \left(\breve{F}_k^{-1} \cdot \nabla \times \lambda_n(\mathbf{p}_q)\right) \left|\det \bar{F}_k\left(\mathbf{p}_q\right)\right| \\ &= \sum_q w_q \left(\bar{F}_k^{-1} \cdot \nabla \lambda_m(\mathbf{p}_q)\right) \times \frac{\partial \mathbf{M}}{\partial \mathbf{B}} \times \left(\bar{F}_k^{-1} \cdot \nabla \lambda_n(\mathbf{p}_q)\right) \left|\det \bar{F}_k\left(\mathbf{p}_q\right)\right|, \end{aligned} \tag{4.39}$$

and G,

$$G = \sum_{k,m,n} G^k_{i,j}. \tag{4.40}$$

On occasion, we will need to calculate the discrete approximation of the continuous L^2 norm of the magnetic vector potential:

$$\|A\left(t\right)\|_2 = \left(\int_\Omega A^2\left(t\right)d\Omega\right)^{\frac{1}{2}}. \tag{4.41}$$

For the discrete problem, this can be written as

$$\|a\left(t\right)\|_{W_a} = \left(a^T \cdot W_a \cdot a\right)^{\frac{1}{2}}. \tag{4.42}$$

W_a is a weighting matrix with elemental components given by

$$W_{i,j}^k = \sum_q w_q \lambda_m(\mathbf{p}_q)\lambda_n(\mathbf{p}_q)\left|\det \bar{F}_k\left(\mathbf{p}_q\right)\right|, \tag{4.43}$$

and overall matrix

$$W_a = \sum_{k,m,n} W_{i,j}^k. \tag{4.44}$$

Notice that, ultimately, the matrices can be equivalently expressed in terms of either the gradient or the curl of the basis functions. This equivalence is limited to the two–dimensional case since the magnetic vector potential (and basis functions) are scalars. In three dimensions, the unknown is a vector field, requiring vector basis functions. The benefit of the preceding analysis is that it is more closely related to the three–dimensional case and the weak-form in (4.19) can be used with less severe modifications.

4.5 Finite–Element Analytic Equation Coupling

The previous sections have dealt with situations for which each component/material in the physical domain Ω under consideration is stationary. For example, in a simple single stator single rotor machine, the stator is modeled in a stationary reference frame and the rotor is modeled in a reference frame that is rotating at its mechanical rotational velocity relative to the stator. This is equivalent to saying that, in the rotating reference frame, the rotor appears fixed and the stator appears to be rotating at the rotor mechanical speed with opposite sign. One could also take an Eulerian approach, modeling the stator and rotor in the same reference frame. This would introduce a velocity dependence into Maxwell's equations due to the $\frac{\partial \mathbf{B}}{\partial t}$ term in Faraday's law and can introduce numerical difficulties to the Péclet effect. This is a common concern in computational fluid dynamics. This is sometimes cited as a reason for the use of the previously described Lagrangian approach. However, the largest difficulty faced when employing a single reference frame is modeling the underlying rotational geometry on a fixed discretization. This is an impossible task in general, and could only be attempted if the boundaries of the underlying geometry lay strictly along lines of constant r or constant θ. These restrictions would severely limit modeling flexibility.

One of the most influential innovations for finite–element simulation of electric machines with rotational effects was the idea of remeshing [23, 24]; at each rotor position, a new mesh is generated in the airgap region to couple the stator and rotor domains. The difficulties associated with remeshing are manifold. They are partly due to the poor aspect ratio of the airgap and partly because of accuracy issues associated with a discretization that appears to be time–varying. The numerical sensitivity of torque calculations has been well documented [108, 109, 110].

However, the use of a rotating reference frame solves many issues associated with modeling complex rotating geometric configurations. This motivated the development of several different approaches aimed at coupling the finite–element solutions to the analytic solution to Laplace's equation within a cylindrical annulus inside the airgap [25, 26, 27]. The motional information appears in terms of time–varying boundary conditions imposed on the airgap by a pair of finite–element models, one for the rotor and one for the stator. The problem of modeling unbounded domains, which has a similar aspect ratio issue for the inverse reason, has also received considerable attention [28, 29, 30, 31]. Many of the techniques are similar.

Dealing with a nonuniform and nonconforming boundary discretization is the main difficulty of this approach, which we deal with in the next section. The main drawback of the analytical solution coupling method is that it introduces dense submatrices into the erstwhile sparse finite–element formulation. This is not as bad as it first appears: a meshed airgap requires a much finer discretization than the rest of the problem which effects similar computational difficulties such as an increased number of matrix entries and factorization fill in, even though the matrix is still "sparse" in the traditional sense.

4.5.1 Small Angle Approximation

When we introduced the analytical equation coupling method, we used the Fourier integrals in Section 2.7.2 to represent the θ-dependence of A in terms of spatial harmonics. In contrast, the finite-element discretization introduced in this chapter are triangles with straight boundaries. Obviously, it is impossible to represent a circular boundary exactly using a finite number of triangles.[2] In fact, an element "touching" the airgap only coincides with the exact (r,θ) coordinates of the boundary at two corners.

For an element adjacent to the airgap, assume the edge of the master-element

$$\Gamma_{k,\alpha} = \{\eta \quad s.t. \quad 0 <= \eta <= 1, \quad \zeta = 0\} \tag{4.45}$$

is mapped by the transformation f_k onto a subsegment of the boundary Γ_α in the sense that

$$\mathfrak{f}_k\left(0,0\right) = \left(r_\alpha, \theta_1\right) \tag{4.46}$$

and

$$\mathfrak{f}_k\left(1,0\right) = \left(r_\alpha, \theta_2\right) \tag{4.47}$$

where θ_1 and θ_2 are the θ coordinates of nodes $\mathbb{G}_k(1)$ and $\mathbb{G}_k(2)$, respectively. The labeling of the η coordinates of the master element is arbitrary, so we can always reassign the coordinates

[2] A function mapping a triangular master element into an element in the physical space with a perfectly circular arc as one boundary would necessarily be a rational transformation [111].

and rewrite $\mathfrak{f}_k$ using a series of linear transformations so that the η–axis corresponds to the airgap edge approximation. The exact function describing the finite-element edge is given by

$$\mathfrak{f}_k(\eta, 0) = \left(\left[(x_1(1-\eta) + x_2\eta)^2 + (y_1(1-\eta) + y_2\eta)^2\right]^{\frac{1}{2}}, \tan^{-1}\left[\frac{y_1(1-\eta) + y_2\eta}{x_1(1-\eta) + x_2\eta}\right]\right). \quad (4.48)$$

where (x_1, y_1) and (x_2, y_2) are the coordinates of nodes $\mathbb{G}_k(1)$ and $\mathbb{G}_k(2)$, respectively. In contrast, the exact edge is described by the following uniform parameterization:

$$(r, \theta) = (r_\alpha, \theta_1(1-\eta) + \theta_2\eta) \quad (4.49)$$

We would like to use the exact parameterizations of the edge in the Fourier expansion in (2.76) instead of the more complicated expression in (4.48), which would require reformulating the harmonic expansion to account for the dependence on the r coordinate.

In terms of Cartesian distance, the maximum error between these two approximations occurs at $\eta = \frac{1}{2}$. At this point, the error vector points in the r direction. If we normalize the error to the radius r_α, the relative error depends only on the angle tended by the edge:

$$\epsilon_r = 1 - cos\left(\frac{\theta_2 - \theta_1}{2}\right) \quad (4.50)$$

The Taylor expansion of ϵ_r to second order is

$$\epsilon_r \approx \frac{1}{4}(\theta_2 - \theta_1)^2. \quad (4.51)$$

If we want to control the discretization error to 1%, an estimate of the maximum allowable edge angle is

$$\theta_{max} = 0.04^{\frac{1}{2}} = 0.2\text{rad} \quad (4.52)$$

or a minimum of $\frac{2\pi}{0.2} \approx 32$ evenly spaced edges, which is an eminently reasonable requirement.

The other source of discretization error occurs because the parameterizations of θ for the straight–edge triangle is nonuniform. The error in theta is given by

$$\epsilon_\theta = \theta_1 (1-\eta) + \theta_2 \eta - \tan^{-1}\left[\frac{y_1 (1-\eta) + y_2 \eta}{x_1 (1-\eta) + x_2 \eta}\right]. \tag{4.53}$$

The Taylor expansion to first order is

$$\epsilon_\theta \approx \left(\frac{x_1 y_2 - x_2 y_1}{x_1^2 + y_1^2} - 1\right)\eta. \tag{4.54}$$

Since the angle error goes to zero at $\eta = 0, \frac{1}{2}, 1$, we can approximate the maximum occurring at $\eta \approx \frac{1}{4}$. In addition, rewriting the Cartesian coordinates in terms of the polar coordinate variables, we find

$$\epsilon_\theta \approx \frac{1}{4}\left(cos(\theta_1)sin(\theta_2) - cos(\theta_2)sin(\theta_1)\right), \tag{4.55}$$

or equivalently,

$$\epsilon_\theta \approx \frac{1}{4} sin(\theta_2 - \theta_1) \approx \frac{1}{4}\left(\theta_2 - \theta_1\right) \tag{4.56}$$

and to achieve a 1% error the maximum tended angle would be limited to

$$\theta_{max} = 0.08\pi \approx 0.25\text{rad} \tag{4.57}$$

This corresponds to a minimum of $\frac{2\pi}{0.25} \approx 25$ evenly spaced edges, which is the same order of magnitude as the previous approximation.

These estimates are very small compared to the number of edges found in a typical electric machine discretization. Electric machines, and their stators in particular, have a multi-toothed structure requiring at least two edges for each tooth. Usually many more are required to accurately capture the significant spatial harmonics introduced by the slotted nature of the stator. In typical cases, the extra discretization error introduced by a small angle approximation can be considered negligible. Care must be taken when analyzing a machine with a small number of teeth and in the discretization of components (e.g. rotors) that do not exhibit the same extreme spatial variability in the θ direction. In these cases, standard discretization algorithms may produce too few edges using default parameters.

4.5.2 Subspace Projections

The results of the previous section are used to approximate the magnetic vector potential A on the boundary as

$$A\left(r_\alpha, \theta\right) = A\left(r_\alpha, \theta_1\left(1-\eta\right) + \theta_2\eta\right) = a_1\left(1-\eta\right) + a_2\eta, \tag{4.58}$$

where a_1 and a_2 are the interpolated values at nodes $\mathbb{G}_k(1)$ and $\mathbb{G}_k(2)$, respectively. This approximation gives the partial contribution to the Fourier integrals in (2.76) from one edge as

$$\frac{\theta_2-\theta_1}{2\pi}\int_0^1 e^{-jp[\theta_1(1-\eta)+\theta_2\eta]}\left[a_1\left(1-\eta\right)+a_2\eta\right]d\eta \tag{4.59}$$

The total expression for harmonic p is

$$\tilde{A}_p\left(r_\alpha\right) = \sum_k \frac{\theta_{m(k)}-\theta_{n(k)}}{2\pi}\int_0^1 e^{-jp\left[\theta_{m(k)}(1-\eta)+\theta_{n(k)}\eta\right]}\left[a_{m(k)}\left(1-\eta\right)+a_{n(k)}\eta\right]d\eta \tag{4.60}$$

where the sum over the elements k is restricted to those with edges on the boundary Γ_α. The expressions $n(k) = \mathbb{G}_k(1)$ and $m(k) = \mathbb{G}_k(2)$ indicate the dependence of the node numbers on the element in the summation. For a single p, (4.60) can be written as an inner product with the entire a vector by using an appropriate zero–padding:

$$\tilde{A}_p\left(r_\alpha\right) = d_{\alpha,p}\cdot a. \tag{4.61}$$

An obvious problem is that the number of harmonics that can be generated this way is countably infinite. If we truncate the total number of harmonics so that $|l| \leq l_{max}$, then the linear operator D_α can be represented as a matrix in the form

$$\mathcal{D}_\alpha = \begin{bmatrix} d_{\alpha,-p_{max}} \\ d_{\alpha,-p_{max+1}} \\ \vdots \\ d_{\alpha,p_{max-1}} \\ d_{\alpha,p_{max}} \end{bmatrix}. \tag{4.62}$$

The choice of p_{max} depends on a tradeoff between accuracy of the representation and storage and computational costs on the other. For the transformation to be pseudo–invertible, $2p_{max}+1$ should be at least as large as the number of nodes on the boundary Γ_α. In practice, the spatial harmonics corresponding to some p are known to be zero due to symmetry arguments and it is possible to select the expansion more carefully.[3] Using at least $2p_{max}+1$ nonzero harmonics will give better results. The mapping

$$\tilde{A}_\alpha = \mathcal{D}_\alpha a \tag{4.63}$$

now expresses a relationship between the finite number of interpolated magnetic vector potential values in a and a truncated set of harmonics $\tilde{A}_\alpha$. This represents a projection of the interpolation onto a subspace spanned by the trigonometric basis functions. The diagonal scaling operators $\mathcal{F}$ and $\mathcal{G}$ and the rotation operator $\mathcal{R}$ of section 2.7.3 can now be represented as simple diagonal matrices as well.

Finally, we must substitute the discrete approximation of H_t into the boundary integral term of (4.19). In particular, we are interested in evaluating the term h_i^k in Section 4.29. Using our approximations, on a particular edge we find $d\Gamma = r_\alpha d\theta = r_\alpha\left(\theta_m - \theta_n\right) d\eta$. When

[3] For example, using antiperiodic boundary conditions while modeling a single pole, only harmonics that have a spatial frequencies at an odd integer multiple of $\frac{N_p}{2}$ are nonzero.

$\lambda_{m,k} = 1 - \eta$ we have

$$h_i^k = \int_{\Gamma_k} \lambda_{m,k} H_t d\Gamma = \int_0^1 r_\alpha \left(\theta_m - \theta_n\right) H_t \left(1 - \eta\right) d\eta. \tag{4.64}$$

A similar statement can be made for $\lambda_{n,k} = \eta$. When the expression for $\mathcal{P}_\alpha(\theta)$ is substituted into 4.64, we find the integral takes the same form as the one found in (4.60), differing only by a factor of $2\pi r_\alpha$. In fact, after evaluating the summation over all h_i^k, we find

$$h_\alpha = 2\pi r_\alpha \mathcal{D}_\alpha^H \mathcal{F}_{\beta,\alpha} \mathcal{D}_\alpha a_\alpha + 2\pi r_\alpha \mathcal{D}_\alpha^H \mathcal{R} \mathcal{G}_{\beta,\alpha} \mathcal{D}_\beta a_\beta, \tag{4.65}$$

$$h_\beta = 2\pi r_\beta \mathcal{D}_\beta^H \mathcal{G}_{\alpha,\beta} \mathcal{R}^H \mathcal{D}_\alpha a_\alpha + 2\pi r_\beta \mathcal{D}_\beta^H \mathcal{F}_{\alpha,\beta} \mathcal{D}_\beta a_\beta. \tag{4.66}$$

For two domains, the coupled finite–element equations become

$$\begin{aligned} &\begin{bmatrix} C_\alpha & 0 \\ 0 & C_\beta \end{bmatrix} \begin{bmatrix} \frac{\partial a_\alpha}{\partial t} \\ \frac{\partial a_\beta}{\partial t} \end{bmatrix} \\ &+ \begin{bmatrix} K_\alpha + 2\pi r_\alpha \mathcal{D}_\alpha^H \mathcal{F}_{\beta,\alpha} \mathcal{D}_\alpha & 2\pi r_\alpha \mathcal{D}_\alpha^H \mathcal{R} \mathcal{G}_{\beta,\alpha} \mathcal{D}_\beta \\ 2\pi r_\beta \mathcal{D}_\beta^H \mathcal{G}_{\alpha,\beta} \mathcal{R}^H \mathcal{D}_\alpha & K_\beta + 2\pi r_\beta \mathcal{D}_\beta^H \mathcal{F}_{\alpha,\beta} \mathcal{D}_\beta \end{bmatrix} \begin{bmatrix} a_\alpha \\ a_\beta \end{bmatrix} + \begin{bmatrix} g_\alpha(a_\alpha) \\ g_\beta(a_\beta) \end{bmatrix} = \begin{bmatrix} f_\alpha(t) \\ f_\beta(t) \end{bmatrix}. \end{aligned} \tag{4.67}$$

This procedure can be followed recursively to couple more than two domains. To keep the notation as simple as possible, we will rewrite the previous expression as

$$C \frac{\partial a}{\partial t} + K(t) a + g(a) = f(t) \tag{4.68}$$

by concatenating the various matrix and vector components. Notice that K now depends on time, which comes from the implicit time–dependence of the rotation matrix $\mathcal{R}$ due to the fixed nonzero rotor velocity. A careful examination of the definition of $\mathcal{G}$ in (2.73) reveals that

$$r_\alpha \mathcal{G}_{\beta,\alpha} = r_\beta \mathcal{G}_{\alpha,\beta}, \tag{4.69}$$

which implies

$$2\pi r_\alpha \mathcal{D}_\alpha^H \mathcal{R} \mathcal{G}_{\beta,\alpha} \mathcal{D}_\beta = \left[2\pi r_\beta \mathcal{D}_\beta^H \mathcal{G}_{\alpha,\beta} \mathcal{R}^H \mathcal{D}_\alpha\right]^H . \tag{4.70}$$

That is, the approximations we have employed in evaluating the boundary integral using the analytical solution to Laplace's equation has preserved the symmetry of the underlying matrices.

4.6 Finite-Element Field-Circuit Coupling

Two changes occur in the field–circuit coupling equations after the domain is discretized. First, the discretized domain shape is not that same as the continuous domain. In particular, the cross sectional area of domain k, $|\Omega_k|$, is different. This term appears in several places throughout Chapter 3. Because the underlying field equations are based on the shape of the discrete domain, it is important to use the discrete area for this value in the equations. Second, the integral of $\frac{\partial A}{\partial t}$ over each Ω_k must be converted into discrete form wherever it occurs. This is accomplished in the same way as in Section 4.4. Chapter 3 lays out a very general framework for coupling the field equations to external circuits. It would be tedious and not very insightful to explicitly list all of the potential combinations of discrete equations that are possible. Instead, we offer a prototype example which is typical after concatenation of many variables and matrices. More details can be found in books on the subject [112, 113].

In the voltage–driven case, a typical formulation, ignoring airgap coupling, looks as follows:

$$\begin{bmatrix} C_{f,f} & 0 & 0 \\ C_{i,f} & 0 & 0 \\ 0 & 0 & C_{v,v} \end{bmatrix} \begin{bmatrix} a' \\ E'_\phi \\ i' \end{bmatrix} + \begin{bmatrix} K_{f,f} & K_{f,i} & 0 \\ 0 & K_{i,i} & K_{i,v} \\ 0 & K_{v,i} & K_{v,v} \end{bmatrix} \begin{bmatrix} a \\ E_\phi \\ i \end{bmatrix} + \begin{bmatrix} g\left(a\right) \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ v\left(t\right) \end{bmatrix} \tag{4.71}$$

The first row of 4.71 represents the finite-element equations. When external circuits are included, E_ϕ moves from an input in the forcing function to an unknown. The second row of equations are the actual field-circuit coupling equations. The unknowns i represent the currents, either strand currents or bundle currents depending on the model used. The final row is a voltage equation, relating the voltage drop represented by E_ϕ and the resistive and inductive drop of external circuit elements due to the currents i to the input voltage $v\left(t\right)$. Following the development of Chapter 3, the matrices are symmetric in the sense that the diagonal blocks are symmetric and $K_{v,i} = K_{i,v}^T$ and $K_{f,i} = C_{i,f}^T$.

4.7 Model Equation

To keep the notation as simple as possible, we will concatenate the results of this chapter into a single equation:

$$Cx' + g\left(t,x\right) = f\left(t\right) \tag{4.72}$$

where we have combined the linear–time varying term $K\left(t\right)$ and the nonlinearity $g(x)$ into a single term:

$$g\left(t,x\right) = K\left(t\right)x + g\left(x\right). \tag{4.73}$$

We have changed the unknown vector from a to x to reflect the fact that the variables represent more than just nodal magnetic vector potential values. This is the model equation that will be studied in the remainder of this dissertation. It is composed of three parts:

1. Finite–element equations for each subdomain with airgap coupling terms for subdomain pairs

2. Field–circuit coupling equations

3. Circuit (bulk parameter) equations

Newton's method will be used to solve (4.72) after it has been discretized in time. This will require calculating $G\left(t,x\right) = \frac{\partial g(t,x)}{\partial x}$, which is given by

$$G\left(t,x\right) = K\left(t\right) + \frac{\partial g\left(x\right)}{\partial x}, \tag{4.74}$$

where $\frac{\partial g(x)}{\partial x}$ has one nonzero block associated with the finite-element unknowns given by $\frac{\partial g}{\partial a}$ as defined in (4.39),

$$\frac{\partial g\left(x\right)}{\partial x}=\begin{bmatrix}\frac{\partial g}{\partial a} & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}. \tag{4.75}$$

The L_2 norm of the magnetic vector potential will be written as

$$\|a\|_{W_a}=\|x\|_W=\left(x^T\cdot W\cdot x\right)^{\frac{1}{2}} \tag{4.76}$$

where W is the matrix

$$W=\begin{bmatrix}W_a & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix} \tag{4.77}$$

and $\|\cdot\|_W$ is the corresponding seminorm.

Chapter 5

Time-Domain Numerical Integration

The discretized electric machine model becomes time–varying when motion of the rotor is considered. Some sort of numerical integration is required to capture the resulting rotational motion. There was little theoretical– or application– oriented work being done to apply numerical integration techniques to any dynamic, nonlinear, magnetic field analysis problems during the early years of electric machine finite–element analysis. The first applications of time-domain numerical integration used the implicit-Euler method [32] and the Crank-Nicholson method [33]. There was little interest in this approach until several years later [34, 35, 36, 37, 38]. This period coincides with the first analysis of a nonlinear, rotational, eddy–current problem appearing in the mid 1980s [39, 40].

Runge–Kutta methods are widely used for the numerical integration of ordinary differential equations. The most obvious benefit of Runge–Kutta methods is that they are more accurate than the implicit–Euler method for a given step–size. Runge–Kutta methods also facilitate the use of adaptive step size selection methods, which coarsen and refine the step-size as dictated by some estimate of the solution error [96, 97, 98]. Related techniques can be used to extrapolate the solution to generate initial conditions for the integration procedure in the next interval [99] and more precisely determine switching events causing discontinuities in the solution or its derivatives [100, 101, 102]. In terms of steady–state analysis, the use of

Runge–Kutta methods, and adaptive methods in particular, can be used to accelerate the transient–analysis procedure toward the steady–state solution [103].

In this chapter we introduce the numerical integration techniques that will be used in three of the four steady–state simulation algorithms in this dissertation. We start by introducing the explicit–Euler method and subsequently examine some key properties of the model equation (4.72) that prevent us from using explicit techniques in general. We then derive the implicit–Euler method as the simplest technique that can be successfully applied to our problem. The idea of implicit integration is extended by introducing the family of Runge–Kutta methods and the associated conditions required of the coefficients to achieve a given order and certain stability properties. Finally, we discuss the special case of diagonally–implicit Runge–Kutta (DIRK) methods, which is the family of techniques preferred in this dissertation [93, 94, 95].

5.1 Explicit–Euler

Given an initial condition for (4.72) at time t_0, we can find an approximate solution at time $t_1 > t_0$ using numerical integration. In their simplest form, numerical integration techniques are based on truncated Taylor series expansions: the higher the degree of the polynomial approximation, the more accurate the solution for a given step size. By step size, we mean the difference $h_k = t_k - t_{k-1}$. The exact solution over the interval $[t_{k-1}, t_k]$ can be written as

$$x_k = x_{k-1} + \int_{t_{k-1}}^{t_k} x' dt \tag{5.1}$$

where the subscript notation is used to indicate the sample time

$$x_k = x\left(t_k\right). \tag{5.2}$$

The most well known numerical integration technique is the explicit–Euler method. It is derived by performing a first–order Taylor series expansion of $x(t)$ at $t = t_{k-1}$:

$$x(t) = x_{k-1} + x'_{k-1}(t - t_{k-1}). \tag{5.3}$$

The value of x_k is approximated by evaluating (5.3) at $t = t_k$. We can solve for x'_{k-1} to find

$$x'_{k-1} = \frac{x_k - x_{k-1}}{t_k - t_{k-1}}, \tag{5.4}$$

which is a simple finite–difference approximation of the time–derivative. These results can be substituted into our model equation to find

$$C\frac{x_k - x_{k-1}}{h_k} + g_{k-1} = f_{k-1}. \tag{5.5}$$

Under certain conditions it is possible to determine x_k as

$$x_k = x_{k-1} + h_k C^{-1}\left[f_{k-1} - g_{k-1}\right]. \tag{5.6}$$

Comparing (5.6) to (5.1), we evidently have

$$\int_{t_{k-1}}^{t_k} x' dt = h_k C^{-1}\left[f t_{k-1} - g_{k-1}\right]. \tag{5.7}$$

The same conclusion could have been reached by transforming (4.72) into the explicit form

$$x' = C^{-1}\left[f(t) - g(t, x)\right] \tag{5.8}$$

and using a one–point rectangular integration rule evaluated at $t = t_{k-1}$ to approximate the integral.

The form of (5.6) is attractive for several reasons. The bracketed portion on the right-hand side is dependent only on information at $t = t_{k-1}$, which means only one function evaluation is required per time step. Furthermore, when the matrix C is linear and time–invariant and C^{-1} exists, it can be calculated (or factored) once at the outset and that initial cost is heavily amortized over many time steps.

5.2 Index of the Model Equation

What are the conditions under which it is valid to write (5.6)? A necessary condition is the existence of C^{-1}. Unfortunately, the matrix C is singular in general when simulating electric machines. It was previously mentioned that C is dependent on the underlying material conductivities. Whenever insulators, materials with zero or effectively zero conductivity, are included in the problem domain, the corresponding rows and columns of C are zero. Air is modeled with zero–conductivity and airgaps are necessary in electric machines for torque production and to allow relative motion. Insulation between stranded conductors are also modeled using a conductivity of zero. In two dimensions, steel laminations are modeled using a conductivity of zero because, in a macroscopic sense, the net current conducted in the z–direction is zero.

Because C is singular, our model equation (4.72) falls into a special class of differential equations called differential–algebraic equations (DAEs). The terminology arises as follows. A subset of the equations represented by (4.72) are "true" differential equations. If we consider E_ϕ the input to the problem so that the vector of nodal magnetic vector potential values is the only unknown, the number of pure differential equations are equal to the rank of $C_{f,f}$ in (4.71). The number of algebraic constraint is equal to the number of zero eigenvalues of $C_{f,f}$, represented by the remaining equations. In this case, our model equation is an "easy" DAE in the sense that it is trivial to put it into the semi–explicit form

$$\begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a'_d \\ a'_a \end{bmatrix} + \begin{bmatrix} C_{f,f}^{-1} g_d \\ g_a \end{bmatrix} + \begin{bmatrix} C_{f,f}^{-1} K_{f,i} \\ 0 \end{bmatrix} E_\phi = 0 \tag{5.9}$$

using a permutation. In this form, the nature of the algebraic constraints is readily apparent since the second row of equations does not depend on a'_a. Differentiating the second row of equations with respect to t reveals that

$$\frac{\partial g_a}{\partial a_d} a'_d + \frac{\partial g_a}{\partial a_a} a'_a = 0. \tag{5.10}$$

When $\frac{\partial g_a}{\partial a_a}$ is invertible, we can rearrange (5.10) and substitute a'_d from (5.9) to find

$$a'_a = \frac{\partial g_a}{\partial a_a}^{-1} \frac{\partial g_a}{\partial a_d} C_{f,f}^{-1} \left[K_{f,i} E_\phi + g_d \right], \tag{5.11}$$

giving us an explicit ODE governing the evolution of the algebraic variables a_a. For two dimensional problems, $\frac{\partial g_a}{\partial a_a}$ is always invertible and (5.9) is an index–1 DAE.

Returning to (4.71), and now fixing i as the input, we can permute the equations into the form

$$\begin{bmatrix} I & 0 & 0 \\ 0 & 0 & 0 \\ C_{i,f} & 0 & 0 \end{bmatrix} \begin{bmatrix} a'_d \\ a'_a \\ E'_\phi \end{bmatrix} + \begin{bmatrix} C_{f,f}^{-1} g_d \\ g_a \\ 0 \end{bmatrix} + \begin{bmatrix} C_{f,f}^{-1} K_{f,i} \\ 0 \\ K_{i,i} \end{bmatrix} E_\phi + \begin{bmatrix} 0 \\ 0 \\ K_{i,v} i \end{bmatrix} = 0. \tag{5.12}$$

By the same procedure, we can show that a_a are index–1 variables under the previously described conditions. If we substitute the value of a'_d into the last equation and differentiate, we find

$$\left[K_{i,i} - C_{i,f} C_{f,f}^{-1} K_{f,i} \right] E'_\phi = C_{i,f} C_{f,f}^{-1} g'_d - K_{i,v} i'. \tag{5.13}$$

Now, E_ϕ is an index–1 variable only if the matrix $K_{i,i} - C_{i,f} C_{f,f}^{-1} K_{f,i}$ is invertible. In fact, it turns out this matrix is the zero matrix. Because $C_{f,i}$ is the result of integral constraints, each row of $C_{f,i}$ can be written as the sum of rows of $C_{f,f}$ as

$$C_{i,f} = S^T C_{f,f} \tag{5.14}$$

where S is a matrix having orthogonal columns consisting of ones and zeros. Recalling from the previous chapter that $C_{i,f} = K_{f,i}^{T}$, we can then write

$$C_{i,f} C_{f,f}^{-1} K_{f,i} = S^{T} C_{f,f} S \tag{5.15}$$

The matrix $K_{i,i}$ is diagonal with entries equal to the material conductivity multiplied by the conductor cross sectional area, which can also be written as $K_{i,i} = S^{T} C_{f,f} S$.

Using the previously derived quantities, it is possible to show that g_d' can be expressed as

$$\begin{aligned} g_d' &= \left[\frac{\partial g_d}{\partial a_d} a_d' + \frac{\partial g_d}{\partial a_a} a_a' \right] \\ &= \left[\frac{\partial g_d}{\partial a_d} C_{f,f}^{-1} \left(-K_{f,i} E_\phi - g_d \right) + \frac{\partial g_d}{\partial a_a} \frac{\partial g_a}{\partial a_a}^{-1} \frac{\partial g_a}{\partial a_d} C_{f,f}^{-1} \left(K_{f,i} E_\phi + g_d \right) \right] \\ &= - \left[\frac{\partial g_d}{\partial a_d} - \frac{\partial g_d}{\partial a_a} \frac{\partial g_a}{\partial a_a}^{-1} \frac{\partial g_a}{\partial a_d} \right] C_{f,f}^{-1} \left[K_{f,i} E_\phi + g_d \right]. \end{aligned} \tag{5.16}$$

This theoretically allows us to rewrite (5.13) in terms of E_ϕ. Proving that E_ϕ is an index–2 variable amounts to differentiating (5.13) once more, and showing that the coefficient matrix of E_ϕ is invertible. This equivalent to showing that the matrix

$$C_{i,f} C_{f,f}^{-1} \left[\frac{\partial g_d}{\partial a_d} - \frac{\partial g_d}{\partial a_a} \frac{\partial g_a}{\partial a_a}^{-1} \frac{\partial g_a}{\partial a_d} \right] C_{f,f}^{-1} K_{f,i} = S^{T} \left[\frac{\partial g_d}{\partial a_d} - \frac{\partial g_d}{\partial a_a} \frac{\partial g_a}{\partial a_a}^{-1} \frac{\partial g_a}{\partial a_d} \right] S \tag{5.17}$$

is invertible. This is true in two–dimensions since the columns of S are orthogonal and the matrix $\left[\frac{\partial g_d}{\partial a_d} - \frac{\partial g_d}{\partial a_a} \frac{\partial g_a}{\partial a_a}^{-1} \frac{\partial g_a}{\partial a_d} \right]$ is symmetric and positive–definite.

5.3 Implicit–Euler

The most well–known implicit numerical integration is the implicit-Euler method. It is derived in a similar manner to the explicit-Euler method by performing a Taylor expansion of x about t_k:

$$x\left(t\right) = x_{k-1} + x_k' \left(t - t_{k-1}\right). \tag{5.18}$$

Evaluating (5.18) at t_k and solving for x'_k yields the finite-difference approximation

$$x'_k = \frac{x_k - x_{k-1}}{t_k - t_{k-1}}. \tag{5.19}$$

Substitution into (4.72) gives

$$\frac{1}{h_k} C x_k + g_k = f_k + \frac{1}{h_k} C x_{k-1}. \tag{5.20}$$

The form of (5.20) is materially different than the result of (5.6). Equation (5.20) is a nonlinear equation and evaluating the integral in (5.1) requires its solution. Unlike the explicit–Euler case, it is not possible to write an explicit form for the integral in (5.1). The best we can say is that, given x_{k-1},

$$\int_{t_{k-1}}^{t_k} x' dt = x_k - x_{k-1} \tag{5.21}$$

where x_k is the solution to (5.20).

5.4 Fully Implicit Runge–Kutta Methods

There are several approaches for developing higher–order numerical integration methods. The family of linear multistep methods is one possibility. However, these methods have an inherent drawback in that their order is limited to 2. We prefer to focus on the class of Runge–Kutta methods, which are very flexible in terms of their design for stability and accuracy.

Application of an s–stage Runge-Kutta method is performed by solving a set of equations of the form

$$C y'_{i,k} + g\left(t_{k-1} + c_i h_k, x_{k-1} + h_k \sum_{j=1}^{s} a_{i,j} y'_{j,k} \right) = f\left(t_{k-1} + c_i h_k\right). \tag{5.22}$$

where the values $y'_{i,k}$ are called the stage–derivatives and the integration formula of (5.1) is

Table 5.1: Generalized Butcher tableau

$\{c_i\}$	$\{a_{i,j}\}$
	$\{b_j\}$

Table 5.2: Butcher Tableau for the Explicit-Euler Method

0	0
	1

approximated as

$$x_k = x_{k-1} + h_k \sum_{j=1}^{s} b_j y'_{j,k}. \tag{5.23}$$

To simplify the notation, we introduce the stage–values

$$y_{i,k} = x_{k-1} + h_k \sum_{j=1}^{s} a_{i,j} y'_{j,k}, \tag{5.24}$$

and stage–times

$$t_{i,k} = t_{k-1} + c_i h_k. \tag{5.25}$$

where i is the stage index and k is the time index. An s–stage method is characterized set of $s^2 + 2s$ coefficients $\{a_{i,j}\}$, $\{b_i\}$, and $\{c_j\}$. Table 5.1 demonstrates how these coefficients are concisely summarized using a Butcher tableau.

Both the explicit– and implicit–Euler methods fit within the Runge–Kutta framework as single stage ($s = 1$) methods. Their Butcher tableaus are listed in Tables 5.2 and 5.3. An example of a two–stage method is given in Table 5.4. The method in Table 5.4 is based on the two–point Gaussian quadrature rule, where the c_i correspond to the abscissae and b_j correspond to the quadrature weights for approximating an integral on the interval $[0, 1]$.

The form of (5.22) and (5.23) give a very intuitive notion of the Runge–Kutta process:

Table 5.3: Butcher Tableau for the Implicit-Euler Method

1	1
	1

Table 5.4: Butcher Tableau for the Two-Stage Gauss Method

$\frac{1}{2} - \frac{\sqrt{3}}{6}$	$\frac{1}{4}$	$\frac{1}{4} - \frac{\sqrt{3}}{6}$
$\frac{1}{2} + \frac{\sqrt{3}}{6}$	$\frac{1}{4} + \frac{\sqrt{3}}{6}$	$\frac{1}{4}$
	$\frac{1}{2}$	$\frac{1}{2}$

generate approximations of x' (the stage–derivatives) at several times and use an appropriately weighted sum to estimate x_{k+1}. The problem with (5.22) as written is that it depends on both the stage–derivatives and the stage–values. Equation (5.24) reveals that the stage–values and the stage–derivatives are affinely–dependent. It is possible to rewrite (5.24) as

$$y'_{i,k} = -\frac{p_i}{h_k} x_k + \frac{1}{h_k} \sum_{j=1}^{s} d_{i,j} y_{j,k} \tag{5.26}$$

where $\{d_{i,j}\}$ is the matrix defined by

$$\{d_{i,j}\} = \{a_{i,j}\}^{-1} \tag{5.27}$$

and $\{p_i\}$ is the vector defined by

$$p_i = \sum_{j=1}^{s} d_{i,j}. \tag{5.28}$$

Substituting this relationship into (5.22) we arrive at an alternative representation that depends only on the stage–values:

$$\sum_{j=1}^{s} \frac{d_{i,j}}{h_k} C y_{j,k} + g\left(t_{i,k}, y_{i,k}\right) = f\left(t_{i,k}\right) + \frac{p_i}{h_k} C x_k. \tag{5.29}$$

Even though Runge–Kutta methods generate approximations of x' during the numerical integration processes, none of the stage–derivatives or any linear combinations there–of necessarily approximate x' consistently in the sense that the pair (x_k, x'_k) satisfies

$$C x'_k + g\left(t_k, x_k\right) = f\left(t_k\right). \tag{5.30}$$

Having consistent derivative approximations is important when analyzing quantities such as conduction losses, which depend on both the solution and its derivative. There is, however, a subset of Runge–Kutta methods that will generate consistent derivative approximations. If we restrict the coefficients so that

$$\begin{aligned} c_s &= 1 \\ b_j &= a_{s,j} \end{aligned} \tag{5.31}$$

then the final stage time is equal to the time at the end of the integration interval,

$$t_{s,k} = t_k \tag{5.32}$$

and the final stage–value is equal to the solution output at the end of the interval,

$$y_{s,k} = x_k. \tag{5.33}$$

Substituting these values into (5.22) for $i = s$ we find

$$Cy'_{s,k} + g\left(t_k, x_k\right) = f\left(t_k\right). \tag{5.34}$$

Evidently, a Runge–Kutta method with these coefficient restrictions will produce a consistent derivative approximation $x'_k = y'_{s,k}$ as the final stage–derivative. We will limit ourselves to methods obeying these restrictions for the remainder of this chapter.

5.5 Stability

The stability function of a Runge–Kutta method arises from its application to the scalar equation

$$\frac{\partial x}{\partial t} = \lambda x. \tag{5.35}$$

With step size h and initial value x_0 at time t_0, the solution x_1 at time $t_1 = t_0 + h$ can be written as

$$x_1 = R\left(h\lambda\right) x_0 \tag{5.36}$$

where the stability function $R\left(z\right)$ is given by

$$R\left(z\right) = \frac{\det\left(I - z\{a\} + z\mathbf{1}\{b\}\right)}{\det\left(I - z\{a\}\right)} \tag{5.37}$$

and $\mathbf{1}$ is a vector of ones. The analytic solution to (5.35) is $x\left(t\right) = x_0 e^{\lambda t}$ and the stability function is an approximation of the exponential function;

$$R\left(z\right) \approx e^{z}. \tag{5.38}$$

The stability domain of a method is the set of all complex z such that $|R\left(z\right)| \leq 1$. The method is said to be A–stable if the stability domain includes the entire left–half plane. This is equivalent to requiring that $|R\left(iz\right)| \leq 1$ for all real z and that $R\left(z\right)$ has no poles in the left–half plane. A method is said to be L–stable if it is A–stable and $\lim_{z\to\infty} R\left(z\right) = 0$. L–stability implies that high–frequency oscillations relative to the step size are filtered out of the numerical solution. This is a desirable property because oscillations occurring at frequencies higher than the Nyquist rate implied by the step size may otherwise show up as spurious oscillations. The trapezoidal method is A–stable but not L–stable and is known to cause difficulties for stiff and algebraic differential equations because of this phenomenon [114].

5.6 Order Conditions

A Runge–Kutta method is said to be order p if the local error of the solution is $\mathcal{O}\left(h^{p+1}\right)$. More precisely, a method is order p if

$$\|x\left(t_k + h\right) - x_k\| \leq K h^{p+1} \tag{5.39}$$

for some constant K and $x\left(t_k + h\right)$ represents the exact solution of (4.72) at $t = t_k + h$ using x_k as an initial condition at $t = t_k$. There is a rich theory of order conditions for Runge–Kutta methods providing sets of sufficient conditions in the form of algebraic constraints on the coefficients that guarantee a method will achieve a given order. Generally, more complex differential equations must satisfy more order conditions. For example, nonlinear ordinary differential equations require more stringent conditions on the Runge–Kutta coefficients than linear ODEs. By the same token, methods for DAEs require even more conditions than those for nonlinear ODEs. The reason for this becomes apparent when one begins to perform Taylor expansions of the solution. More general forms of differential equations lead to increasingly complex Taylor series expansions and more conditions on the coefficients are required for the exact and numerical solution to achieve agreement up to a given order.

A set of order conditions for index–2 DAEs was developed in [114]. In general, the method must satisfy all the order conditions for an index–1 DAE in order for the differential and algebraic variables to obtain a given order. Additional order conditions are required for the index–2 variables to obtain a given order. The presentation in [114] gives order conditions for methods obtaining order 3 in the index–1 components order 2 in the index–2 components. In fact, because our problem is linear in x', the number of order conditions that must be satisfied are somewhat less than the full number of conditions derived therein. The necessary conditions are repeated here in Tables 5.5 and 5.6 for the index–1 and index–2 components, respectively.

In general, the index–2 components can only achieve an order one less than the index–1 components. In our case, this means the external circuit variables will obtain a reduced order compared to the magnetic vector potential variables. This is somewhat intuitive, as current and voltage are both directly affected by the time rate of change of the magnetic flux–density through Faraday's law.

Table 5.5: Order conditions for the index–1 components

Order	Condition
1	$\sum_i b_i = 1$
2	$2\sum_i b_i c_i = 1$
2	$\sum_i b_i d_{i,j} c_j^2 = 1$
3	$3\sum_{i,j} b_i c_i^2 = 1$
3	$6\sum_{i,j} b_i a_{i,j} c_j = 1$
3	$\sum_{i,j} b_i d_{i,j} c_j^3 = 1$
3	$2\sum_{i,j,k} b_i d_{i,j} c_j a_{j,k} c_k = 1$

Table 5.6: Additional order conditions for the index–2 components

Order	Condition
1	$\sum_{i,j,k} b_i d_{i,j} d_{j,k} c_k^2 = 2$
2	$\sum_{i,j,k} b_i d_{i,j} d_{j,k} c_k^3 = 3$
2	$2\sum_{i,j,k,l} b_i d_{i,j} d_{j,k} c_k a_{k,l} c_l = 3$

5.7 Runge–Kutta Methods with an Explicit First Stage

An important class of implicit Runge–Kutta methods for index–2 DAEs are those having an $\{a_{i,j}\}$ matrix with $a_{1,j} = 0$, i.e., an explicit first stage, and having an invertible submatrix $\hat{a}_{i,j} = a_{i+1,j+1}$ for $1 \leq i, j \leq s-1$. In terms of the order conditions of the previous section, this class of methods can be viewed has having an $\{a_{i,j}\}$ matrix with $a_{1,1} = c_1$, $a_{1,j} = 0$ for $j > 1$, and satisfying the necessary order conditions in the limit as $c_1 \to 0$.

In this case, the stage–values are given by

$$\begin{aligned} y_{1,k} &= y_{s,k-1} \\ y_{i,k} &= y_{s,k-1} + h_k a_{i,1} y'_{s,k-1} + h_k \sum_{j=1}^{s-1} \hat{a}_{i,j} y'_{j+1,k}. \end{aligned} \tag{5.40}$$

Table 5.7: Modified Butcher table for methods with an explicit first stage

$\{\hat{c}_i\}$	$\{\hat{q}_i\}$	$\{\hat{a}_{i,j}\}$
	b_1	$\{\hat{b}_j\}$

The stage–derivatives can be written in terms of the stage–values as

$$\begin{aligned} y'_{1,k} &= y'_{s,k-1} \\ y'_{i+1,k} &= -\hat{q}_i y'_{s,k-1} - \frac{\hat{p}_i}{h_k} y_{s,k-1} + \frac{1}{h_k} \sum_{j=1}^{s-1} \hat{d}_{i,j} y_{j+1,k}, \end{aligned} \tag{5.41}$$

where

$$\{d_{i,j}\} = \{\hat{a}_{i,j}\}^{-1} \tag{5.42}$$

$$\hat{p}_i = \sum_{j=1}^{s-1} \hat{d}_{i,j} \tag{5.43}$$

$$\hat{q}_i = \sum_{j=1}^{s-1} \hat{d}_{i,j} a_{j+1,1}. \tag{5.44}$$

An important consequence of L–Stability is that $\hat{q}_{s-1} = 0$, implying that $y'_{s,k}$ can be calculated independently of $y'_{s,k-1}$.[1]

Alternatively, we may describe methods satisfying these conditions using a modified set of coefficients as in Table 5.7 where $\hat{c}_i = c_{i+1}$ and $\hat{b}_i = b_{i+1}$. Any standard Runge–Kutta method can be expressed in this form with $\hat{q}_i = 0$ and $b_1 = 0$. From now on, we assume all methods have coefficients in the form of Table 5.7 and drop the "hats" from the coefficient matrices and vectors for convenience. This is done with the understanding that when referring to an "s–stage" method, s refers to the number of implicit stages in the standard form. The vector $\{b_i\}$ is now considered to have length $s + 1$.

[1] The trapezoidal method has $q_{s-1} = 1$, which goes some way in explaining the difficulty of the strategy when applied to stiff problems.

It is now possible to rewrite (5.29) as

$$\sum_{j=1}^{s} \frac{d_{i,j}}{h_k} C y_{j,k} + g\left(t_{i,k}, y_{i,k}\right) = f\left(t_{i,k}\right) + \frac{p_i}{h_k} C y_{s,k-1} + q_i C y'_{s,k-1}. \tag{5.45}$$

Because the pair $(y_{s,k-1}, y'_{s,k-1})$ satisfies the DAE at time t_{k-1}, we can write

$$C y'_{s,k-1} = f\left(t_{k-1}\right) - g\left(t_{k-1}, y_{s,k-1}\right), \tag{5.46}$$

which, after substitution into (5.45), yields an expression of (5.29) using only the stage–values:

$$\sum_{j=1}^{s} \frac{d_{i,j}}{h_k} C y_{j,k} + g\left(t_{i,k}, y_{i,k}\right) = f\left(t_{i,k}\right) + \frac{p_i}{h_k} C y_{s,k-1} + q_i \left[f\left(t_{k-1}\right) - g\left(t_{k-1}, y_{s,k-1}\right)\right]. \tag{5.47}$$

The final issue that must be addressed is the interdependence of the stage–values. That is to say, (5.47) actually represents a set of s equations, one for each stage. Each stage–equation is dependent on all of the stage values, so they must be solved simultaneously. This can be expressed in a single equation by concatenating the unknowns for all the stages as

$$\frac{1}{h_k} \bar{C}_d \bar{y}_k + \bar{g}_k = \bar{f}_k + \frac{1}{h_k} \bar{C}_p \bar{y}_{k-1} + \bar{I}_q \left[\bar{f}_{k-1} - \bar{g}_{k-1}\right], \tag{5.48}$$

where the overbars denote the one–dimensional (row–wise) concatenation of the vectors $y_{i,k}$, $g\left(t_{i,k}, y_{i,k}\right)$, and $f\left(t_{i,k}\right)$:

$$\{\bar{y}_k\}_i = y_{i,k} \tag{5.49}$$

$$\{\bar{g}_k\}_i = g\left(t_{i,k}, y_{i,k}\right) \tag{5.50}$$

$$\{\bar{f}_k\}_i = f\left(t_{i,k}\right) \tag{5.51}$$

The matrices $\bar{C}_p$ have entries given by

$$\{\bar{C}_p\}_{i,j} = \begin{cases} p_i C & j = s \\ \\ 0 & j \neq s, \end{cases} \tag{5.52}$$

and

$$\{\bar{I}_q\}_{i,j} = \begin{cases} q_i I & j = s \\ \\ 0 & j \neq s, \end{cases} \tag{5.53}$$

respectively. The fact that only the last columns of $\bar{C}_p$ and $\bar{I}_q$ are nonzero is a direct consequence of the coefficient restriction assumptions (5.31) and the resulting property of (5.33). The matrix $\bar{C}_d$ is constructed similarly, with entries obtained by taking the Kroenecker product of $\{d_{i,j}\}$ with C:

$$\{\bar{C}_d\}_{i,j} = d_{i,j} C. \tag{5.54}$$

When applying Newton's method to solve (5.48), the gradient of $\bar{g}$ will be written as $\bar{G}$, and is the block-diagonal matrix with nonzero entries given by

$$\{\bar{G}_k\}_{i,i} = G\left(t_{i,k}, y_{i,k}\right). \tag{5.55}$$

Historically, the standard statement of Runge–Kutta methods as presented in the beginning of this section is motivated by their application to time–marching problems. We suspect this bias in presentation poses some conceptual difficulties when attempting to apply these numerical integration techniques to steady–state algorithms and have found no evidence of any attempts to do so. The steps and assumptions required to move from (5.22) to (5.48) represent part of this conceptual barrier. It should be now be evident, however, that higher–order Runge–Kutta techniques are applicable wherever one can derive an implicit–Euler based algorithm.

5.8 Diagonally Implicit Runge–Kutta Methods

One issue with fully–implicit Runge-Kutta methods is that they lead to a set of simultaneous equations for each time–step. That is to say, all of the stage–values must be determined at once. For scalar ODEs and small vector DAEs, this is may or may not be an issue. For problems arising from the discretization of nonlinear PDEs such as electric machine simulations, the resulting DAE is quite large. The concatenation procedure previously outlined increases the problem size linearly with the number of stages. It is well known, however, that the computational expense of solving linear systems grows greater than linearly with problem size, even for sparse problems. Added to this is the fact that most $\{a_{i,j}\}$ matrices are not symmetric, which destroys a desirable property of the matrix equations once concatenated.

The family of diagonally–implicit Runge–Kutta (DIRK) methods offers a solution to this problem. These methods are characterized by a lower–triangular matrix:

$$a_{i,j} = 0, \quad j > i. \tag{5.56}$$

DIRK methods have the property that the i^{th} stage–value $y_{i,k}$ depends only on previous stage–values $y_{j,k}$ for $j < i$. This can be seen by rewriting (5.47) using (5.56) as

$$\frac{d_{i,i}}{h_k}Cy_{i,k} + g\left(t_{i,k}, y_{i,k}\right) = f\left(t_{i,k}\right) + \frac{p_i}{h_k}Cy_{s,k-1} - \sum_{j=1}^{i-1}\frac{d_{i,j}}{h_k}Cy_{j,k} + q_i\left[f\left(t_{k-1}\right) - g\left(t_{k-1}, y_{s,k-1}\right)\right]. \tag{5.57}$$

The lower triangularity of $\{a_{i,j}\}$ means its inverse $\{d_{i,j}\}$ is lower triangular as well. Using a DIRK method, $x_k = y_{s,k}$ can be calculated by sequentially solving (5.57) for $i = 1, \ldots, s$. Calculating each stage–value requires determining the solution to a nonlinear equation of the same size and symmetry properties as the original problem. Therefore, the computation time grows linearly with the number of stages. The tradeoff is that the order of a DIRK method will generally be less than a fully–implicit method with the same number of stages.

Table 5.8: Diagonally Implicit Runge–Kutta method of order 2/1 (ESDIRK2).

0	0	0	0
$2-\sqrt{2}$	$1-\frac{\sqrt{2}}{2}$	$1-\frac{\sqrt{2}}{2}$	0
1	$\frac{\sqrt{2}}{4}$	$\frac{\sqrt{2}}{4}$	$1-\frac{\sqrt{2}}{2}$
	$\frac{\sqrt{2}}{4}$	$\frac{\sqrt{2}}{4}$	$1-\frac{\sqrt{2}}{2}$

Table 5.9: Diagonally Implicit Runge-Kutta method of order 3/2 (ESDIRK3).

0	0	0	0	0
2γ	γ	γ	0	0
c_3	$c_3-a_{3,2}-\gamma$	$a_{3,2}$	γ	0
1	b_1	b_2	b_3	γ
	b_1	b_2	b_3	γ

The ubiquitous implicit–Euler method is a 1 stage DIRK method.

For a DIRK method to satisfy any of the order conditions for index–2 DAEs, it is necessary that the first row of the $\{a_{i,j}\}$ matrix be zero. For methods of orders 2/1 and 3/2, there end up being 1 and 2 free parameters, respectively. A typical way of constraining the parameters is to require the diagonal entries of $\{a_{i,j}\}$ to be equal. This gives the method of order 2/1 in Table 5.8 [115]. The method given in Table 5.9 is of order 3/2 [116]. The parameter γ is the root of the polynomial

$$-\gamma^3+\frac{3}{2}\gamma^2-\frac{1}{2}\gamma+\frac{1}{24}=0, \tag{5.58}$$

in the interval $\left[\frac{1}{3},1\right]$, and the remaining coefficients are given by

$$c_3=\frac{2\gamma\left(\gamma-\frac{1}{4}\right)\left(\gamma-1\right)}{\left(\gamma-\frac{1}{2}\right)^2-\frac{1}{12}} \tag{5.59}$$

$$a_{3,2}=\frac{c_3^2-2c_3\gamma}{4\gamma} \tag{5.60}$$

$$b_1=\frac{\frac{1}{6}+\gamma^2-\gamma^2c_3+\frac{3}{2}\gamma c_3-\gamma-\frac{1}{4}c_3}{\gamma c_3} \tag{5.61}$$

$$b_2=\frac{\frac{1}{3}-\gamma+\frac{1}{2}c_3+\gamma c_3}{2\gamma\left(2\gamma-c_3\right)} \tag{5.62}$$

$$b_3=\frac{\frac{1}{3}-2\gamma+2\gamma^2}{c_3\left(c_3-2\gamma\right)}. \tag{5.63}$$

Algorithm 5.1 Diagonally Implicit Runge-Kutta Newton Method

```
function DIRKNewton($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$, $i$, $k$)
    $h_k \leftarrow t_k - t_{k-1}$
    $y'_{i,k} \leftarrow -p_i y_{s,k-1}$
    for $j = 1 : i - 1$ do
        $y'_{i,k} \leftarrow y'_{i,k} + d_{i,j} y_{j,k}$
    end for
    $y'_{i,k} \leftarrow h_k^{-1} y'_{i,k} - q_i y'_{s,k-1}$
    $f_{i,k} \leftarrow f(t_{i,k}) - C y'_{i,k}$

    $r_{i,k} \leftarrow \frac{d_{i,i}}{h_{i,k}} C y_{i,k} + g(t_{i,k}, y_{i,k}) - f_{i,k}$
    while $\|r_{i,k}\| > \|f_{i,k}\| \epsilon$ do
        $J_{i,k} \leftarrow h_k^{-1} d_{i,i} C + G(t_{i,k}, y_{i,k})$
        $y_{i,k} \leftarrow y_{i,k} - J_{i,k}^{-1} r_{i,k}$
        $r_{i,k} \leftarrow h_k^{-1} d_{i,i} C y_{i,k} + g(t_{i,k}, y_{i,k}) - f_{i,k}$
    end while
    $y'_{i,k} \leftarrow y'_{i,k} + h_k^{-1} d_{i,i} y_{i,k}$
    return $y_{i,k}$, $y'_{i,k}$
end function
```

Both methods are L–stable.

Algorithm 5.1 presents Newton's method for solving 5.48 when the Runge–Kutta method is diagonally–implicit. In particular, it is a function for calculating the i^{th} stage–value for the k^{th} time step. It is assumed that the stages $j < i$ have already been determined along with the solution for the previous time step $y_{s,k-1}$.

5.9 Interpolation

The order conditions in Section 5.6 are normalized to the step–size h_k. Letting u be a free parameter, the requirements for an approximate solution at time $t(u) = t_{k-1} + u h_k$ to obtain a given order can be written as in Tables 5.10 and 5.11. Using these conditions, it is possible to construct a parametric quadrature weight vector $\{\bar{b}_j(u)\}$ which, given the stage–derivatives, interpolates the solution over the interval $[t_{k-1}, t_k]$. If a method is of order p, it is generally not possible to construct $\{\bar{b}_j(u)\}$ to satisfy the same order conditions for all u. The goal, then, is to satisfy as many possible order conditions for all u, starting with the lowest order

Table 5.10: Interpolation order conditions for the index–1 components

Order	Condition
1	$\sum_i b_i = u$
2	$2\sum_i b_i c_i = u^2$
2	$\sum_i b_i d_{i,j} c_j^2 = u^2$
3	$3\sum_{i,j} b_i c_i^2 = u^3$
3	$6\sum_{i,j} b_i a_{i,j} c_j = u^3$
3	$\sum_{i,j} b_i d_{i,j} c_j^3 = u^3$
3	$2\sum_{i,j,k} b_i d_{i,j} c_j a_{j,k} c_k = u^3$

Table 5.11: Additional order conditions for the index–2 components

Order	Condition
1	$\sum_{i,j,k} b_i d_{i,j} d_{j,k} c_k^2 = 2u$
2	$\sum_{i,j,k} b_i d_{i,j} d_{j,k} c_k^3 = 3u^2$
2	$2\sum_{i,j,k,l} b_i d_{i,j} d_{j,k} c_k a_{k,l} c_l = 3u^2$

and moving up from there.

We will write the value approximated by interpolation to time $t_k(u) = t_k + uh_k$ as

$$\hat{y}(t_{k-1} + uh_k) = y_{s,k-1} + h_k \sum_{i=1}^{n} u^i \left[\bar{b}_{i,1} y'_{s,k-1} + \sum_{j=1}^{s} \bar{b}_{i,j+1} y'_{j,k} \right] \qquad (5.64)$$

where the matrix $\{\hat{b}_{i,j}\}$ is an n by $s+1$ by matrix. Algorithm 5.2 contains a pseudo-code implementation of the interpolation procedure for the solution over the interval $[t_{k-1}, t_k]$ to the time $t_{k-1} + uh_k$. For the implicit–Euler method, the matrix is 2 by 1 with a single non–zero entry $\bar{b}_{i,2} = u$. For the ESDIRK2 method, it is actually possible to solve all the conditions to achieve order 2/1 for arbitrary u. This results in the matrix of interpolation coefficients given in Table 5.12.

For the ESDIRK3 method, it is not possible to satisfy all order conditions to obtain order 3/2 for all u. It is possible, however, to satisfy the conditions to obtain order 2/1 and the first two conditions for order 3 list in Table 5.10. The two extra conditions were chosen to be satisfied on the basis that they must necessarily be satisfied for ODEs (in addition to

Algorithm 5.2 Runge-Kutta Interpolator

```
function RKInterpolate($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$, $k$, $u$)
    $h_k \leftarrow t_k - t_{k-1}$
    $\hat{y} \leftarrow y_{s,k-1}$
    for $i = 1 : s+1$ do
        $\hat{y} \leftarrow \hat{y} + h_k u^i \bar{b}_{i,1} y'_{s,k-1}$
        for $j = 1 : s$ do
            $\hat{y} \leftarrow \hat{y} + h_k u^i \bar{b}_{i,j+1} y'_{j,k}$
        end for
    end for
    return $\hat{y}$
end function
```

Table 5.12: ESDIRK2 Interpolation Coefficients

	$\bar{b}_{:,1}$	$\bar{b}_{:,2}$	$\bar{b}_{:,3}$
$\bar{b}_{1,:}$	$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{2}}{2}$	$1-\sqrt{2}$
$\bar{b}_{2,:}$	$-\frac{\sqrt{2}}{4}$	$-\frac{\sqrt{2}}{4}$	$\frac{\sqrt{2}}{2}$

DAEs). This gives the matrix of interpolation coefficient in Table 5.13, written in terms of the parameter γ.

5.10 Error Estimation

The classical error estimation technique for Runge–Kutta methods employs an auxiliary weight vector $\{\hat{b}_j\}$ that gives a solution $\hat{y}_{s,k}$ with order less than the solution obtained using the weight vector from the method proper. If the less accurate method obtains order $p-1$, the difference between the two solutions $y_{s,k} - \hat{y}_{s,k}$ is $\mathcal{O}\left(h^p\right)$. This gives an error estimate of the lower order method. We focus on estimating the error of the index–1 components, that

Table 5.13: ESDIRK3 Interpolation Coefficients

	$\bar{b}_{:,1}$	$\bar{b}_{:,2}$	$\bar{b}_{:,3}$	$\bar{b}_{:,4}$
$\bar{b}_{1,:}$	$\frac{-24\gamma^3+34\gamma^2-18\gamma+3}{4(3\gamma^2-3\gamma+1)(2\gamma^2-4\gamma+1)}$	$\frac{6\gamma^2-6\gamma+1}{4\gamma^2-8\gamma+2}$	$-\frac{6\gamma^2-4\gamma+1)^2}{4(3\gamma^2-3\gamma+1)(2\gamma^2-4\gamma+1)}$	$\frac{2\gamma^2 u}{2\gamma^2-4\gamma+1}$
$\bar{b}_{2,:}$	$\frac{24\gamma^4-28\gamma^3+10\gamma^2+2\gamma-1}{8\gamma(3\gamma^2-3\gamma+1)(2\gamma^2-4\gamma+1)}$	$-\frac{(6\gamma^2-1)(\gamma-1)}{2(2\gamma-1)(2\gamma^2-4\gamma+1)}$	$\frac{(2\gamma^2-1)(6\gamma^2-4\gamma+1)^2}{8\gamma(2\gamma-1)(3\gamma^2-3\gamma+1)(2\gamma^2-4\gamma+1)}$	$-\frac{2\gamma}{2\gamma^2-4\gamma+1}$
$\bar{b}_{3,:}$	$\frac{-(3\gamma-1)(2\gamma^2-2\gamma+1)}{12\gamma(3\gamma^2-3\gamma+1)(2\gamma^2-4\gamma+1)}$	$\frac{6\gamma^2-10\gamma+3}{6(2\gamma-1)(2\gamma^2-4\gamma+1)}$	$-\frac{(\gamma-1)(6\gamma^2-4\gamma+1)^2}{12\gamma(2\gamma-1)(3\gamma^2-3\gamma+1)(2\gamma^2-4\gamma+1)}$	$\frac{1}{6\gamma^2-12\gamma+3}$

Table 5.14: ESDIRK2 auxiliary weight vector of order 1/1

$\hat{b}_1$	$\hat{b}_2$	$\hat{b}_3$
$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{2}}{2}$	$1-\sqrt{2}$

is, the magnetic vector potential variables, since they make up the majority of the unknowns and are directly related to the loss calculations.

The notion that $\{\hat{b}_j\}$ not satisfy as many order conditions as $\{b_j\}$ is somewhat vague. There are many ways to fail to satisfy a set of order conditions that, for the purpose of error estimation, are practically useless. This occurs, for example, when the coefficients just barely fail to satisfy some conditions and thus give unreasonably small estimates. One choice is to select $\{\hat{b}_j\}$ such that the coefficient of z^p is annihilated in the Taylor expansion of the stability function;

$$\hat{R}\left(z\right) = \frac{\det\left(I - z\{a\} + z\mathbf{1}\{\hat{b}\}\right)}{\det\left(I - z\{a\}\right)}, \tag{5.65}$$

resulting in a Taylor expansion equal to

$$\hat{R}\left(z\right) = \sum_{m=0}^{p-1} \frac{z^m}{m!} + \mathcal{O}\left(z^{p+1}\right). \tag{5.66}$$

The difference between the stability functions R and $\hat{R}$ for the method using b and $\hat{b}$ will be equal to $\frac{1}{p!}z^p + \mathcal{O}\left(z^{p+1}\right)$. This is actually equivalent to requiring that $\{\hat{b}_j\}$ satisfy the condition

$$\sum_{j=1}^{s} \hat{b}_j c_j^{p-1} = 0, \tag{5.67}$$

which says that the quadrature formula with weights $\{\hat{b}_j\}$ and abscissae $\{c_i\}$ will give a value of zero when integrating the polynomial t^{p-1}. The auxiliary vectors derived in this manner for ESDIRK2 and ESDIRK3 are given in Tables 5.14 and 5.15 respectively.

Additionally, with this choice we can write the error estimate for the index–1 components

Table 5.15: ESDIRK3 auxiliary weight vector of order 2/1

$\hat{b}_1$	$\hat{b}_2$	$\hat{b}_3$	$\hat{b}_4$
$-\frac{(2\gamma-1)^2(6\gamma^2-4\gamma+1)}{8\gamma(3\gamma^2-3\gamma+1)(2\gamma^2-4\gamma+1)}$	$\frac{3\gamma^2-1/2}{(2\gamma-1)(4\gamma^2-8\gamma+2)}+\frac{3}{4}$	$\frac{(6\gamma^2-4\gamma+1)^2(\gamma^2-\gamma+1/2)}{4(2\gamma^2-4\gamma+1)(-6\gamma^4+9\gamma^3-5\gamma^2+\gamma)}$	$\frac{2\gamma(\gamma-1)}{2\gamma^2-4\gamma+1}$

Algorithm 5.3 Runge-Kutta Error Indicators

```
function RKError($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$)
    $\kappa \leftarrow 0$
    for $k = 1 : N_t$ do
        $h_k \leftarrow t_k - t_{k-1}$
        $e_v \leftarrow \left(b_1 - \hat{b}_1\right) y'_{s,k-1}$
        for $i = 1 : s$ do
            $e_v \leftarrow e_v + \left(b_{i+1} - \hat{b}_{i+1}\right) y'_{i,k}$
        end for
        $\epsilon_k \leftarrow h_k \|e_v\|_W$
        $\kappa \leftarrow \max\left(\kappa, \|e_v\|_W\right)$
    end for
    for $k = 1 : N_t$ do
        $\epsilon_k \leftarrow \kappa^{-1} \epsilon_k$
    end for
    return $\{\epsilon_k\}$
end function
```

as

$$y_{s,k} - \hat{y}_{s,k} = \frac{h_k^{p-1}}{(p-1)!} x^{(p-1)}\left(t_k\right) + \mathcal{O}\left(h^p\right) \tag{5.68}$$

where $x^{(p)}$ is the p^{th} derivative of the true solution at t_k starting from $x\left(t_{k-1}\right) = y_{s,k-1}$. In fact, the error estimate in and of itself is not of much interest. One is typically most interested in the error relative to the solution. We will use the relative error ϵ_k as an indicator of the local discretization error for time step k where,

$$\epsilon_k = \frac{\|y_{s,k} - \hat{y}_{s,k}\|_W}{\max\limits_k \|y_{s,k}\|_W}, \tag{5.69}$$

and $\|\cdot\|_W$ is the seminorm described in Chapter 4. Algorithm 5.3 contains a pseudo–code implementation for calculating the error indicators.

One issue with this scheme is that the error estimates are actually error estimates of the lower order solution. In point of fact, the numerical integration procedure is nearly always advanced using the higher order solution and the term “error estimate” is somewhat of a misnomer in this context. The choice of continuing with the higher order solution is sometimes referred to as local extrapolation. In this sense, the ϵ_k are not strictly related to the error of the solution and are merely used to facilitate adaptive step size selection.

Chapter 6

Steady-State Analysis Algorithms

The general continuous time steady-state analysis problem can be stated in the following form:

$$\begin{aligned} \texttt{Solve}\ \ & Cx' + g(t,x) = f(t)\,, \\ \texttt{until}\ \ & x(t) = x(t+T)\,. \end{aligned} \tag{6.1}$$

We assume that f and g are periodic with period T and that a unique solution satisfying the constraint $x(t) = x(t+T)$ exists independent of $x(0)$. Under these conditions, the problem statement of (6.1) is equivalent to the problem

$$\begin{aligned} \texttt{Solve}\ \ & Cx' + g(t,x) = f(t)\,, \\ \texttt{for}\ \ & t \in [0,T]\,, \\ \texttt{s.t.}\ \ & x(0) = x(T)\,. \end{aligned} \tag{6.2}$$

The statement (6.1) views continuous periodicity as the fundamental characteristic of the steady–state solution without regard to any fixed interval. For stable systems, this suggests simulating an initial–value problem over a sufficiently long interval so that the contribution of the transients to the solution is negligible. This is the approach taken by the transient analysis algorithms in Section 6.1.1.

On the other hand, (6.2) is more closely related to a boundary–value problem. Shooting-methods are general strategies for solving boundary value problems and are applicable to problems like (6.2) having periodic boundary–value constraints. The shooting–Newton method of Section 6.1.2 formulates the discrete problem in terms of a single nonlinear boundary–value problem that is solved using Newton's method. For this reason, the strategy is sometimes referred to as the single shooting method. The direct multiple shooting method of Section 6.1.3 introduces degrees of freedom in the interior of the interval $[0, T]$ and solves a system of coupled boundary–value problems.

Returning to (6.1), the periodicity constraint on the solution suggests an approach different to transient analysis. By assuming the solution can be expressed as a weighted sum of sinusoidal basis functions, the periodicity of the solution can be satisfied a priori. The problem then becomes one of finding weights that approximate the solution in some best sense. This leads to the harmonic balance method of Section 6.2.

The order of presentation of the algorithms in this chapter has been chosen for a particular reason. Transient analysis is the de facto standard for steady–state analysis because of the ease of implementation and understanding, despite it being the least efficient of the methods. The inefficiency of transient analysis is the result of simulation time increasing with the ratio of excitation period to time–constant. Magnetic devices in particular are designed to be excited with signals having periods much shorter than the natural time constants. This implies that many periods of transient analysis are required to converge to the steady-state solution.

Algorithmically, the single shooting method is very similar to transient analysis. The main difference is that at the end of every transient analysis period a correction is made to the solution based on the periodic boundary–value formulation of the problem. The multiple shooting method takes the boundary–value view of the problem adopted by the single shooting method and pushes it to the limit. The introduction of internal degrees of freedom removes the need to perform transient analysis all together. The tradeoff is that it becomes necessary to deal with a much larger system of equations. Although perhaps not obvious at first, the harmonic balance method is similar to the multiple shooting method in most respects. Whereas the multiple shooting method uses numerical integration techniques to approximate the solution derivatives, the harmonic balance method appeals to the frequency–domain to perform derivative calculations. The frequency domain derivative approximation results in discrete solutions that are more accurate, but also endows the problem with a much more difficult structure.

The unifying difference between transient analysis and the other steady–state algorithms is that the later solved using Newton's method. This requires the solution of a linear system at each Newton iteration. Direct methods are generally unsuitable for the solution of these problems. Therefore, we use the generalized minimum residual method (GMRES) to solve linear equations resulting from the Newton iteration, which is briefly described in Appendix A.

6.1 Time–Domain Methods

6.1.1 Transient Analysis

Transient analysis refers to the direct application of numerical integration techniques to analyze (4.72) as an initial value problem with initial condition $x\left(0\right)$. Transient analysis can be used to determine the steady–state solution with arbitrary precision by letting $t \to \infty$. In reality, one monitors the norm of the error between the solution shifted in time by one period, $\|x\left(t\right) - x\left(t-T\right)\|$, and terminates the analysis when this error falls below a threshold ϵ. When n periods have been simulated, the approximate steady–state solution is taken to be the solution over the last period, $x\left(t\right)$ for $t \in \left[nT-T, nT\right]$. The tolerance ϵ determines how close the approximate solution is to the true steady–state solution. The initial condition $x\left(0\right)$ is an input to the problem, ideally chosen as close as possible to the true steady–state solution.

Algorithm 6.1 Fixed Step-Size Transient Analysis

```
function TAFixed($y_{s,0}$, $\{t_k\}$, $\epsilon$, $N_{max}$)
    $\{y_{i,k}\}$, $\{y'_{i,k}\}$ ←TAPeriod($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$)
    $N_{iter} \leftarrow 1$
    while $\|y_{s,N_t} - y_{s,0}\| \geq \epsilon \|y_{s,N_t}\|$ and $N_{iter} \leq N_{max}$ do
        $y_{s,0}$, $y'_{s,0} \leftarrow y_{s,N_t}$, $y'_{s,N_t}$
        $\{y_{i,k}\}$, $\{y'_{i,k}\}$ ←TAPeriod($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$)
        $N_{iter} \leftarrow N_{iter} + 1$
    end while
    return $\{y_{s,k}\}$, $\{y'_{s,k}\}$
end function
```

The simplest method to solve (6.1) using numerical integration is to choose a fixed step-size $h_k = \frac{T}{N_t}$ based on N_t steps over one period. More generally, we will assume a fixed set of N_t time-points $\{t_k\}$ are used to discretized the time axis with corresponding step sizes $\{h_k = t_k - t_{k-1}\}$ and initial time t_0. Given this set, a discrete time restatement of (6.2) is simply a matter of employing the generic Runge–Kutta notation of (5.48):

$$\begin{aligned} \texttt{Solve}\ \ & \frac{1}{h_k}\bar{C}_d\bar{y}_k + \bar{g}_k = \bar{f}_k + \frac{1}{h_k}\bar{C}_p\bar{y}_{k-1} + \bar{I}_q\left(\bar{f}_{k-1} - \bar{g}_{k-1}\right), \\ \texttt{for}\ \ & t_k \in [0, T], \\ \texttt{s.t.}\ \ & \bar{y}_0 = \bar{y}_{N_t}. \end{aligned} \tag{6.3}$$

After one period of analysis, we set $\bar{y}_0 \leftarrow \bar{y}_{N_t}$ so that only one period of the solution must be stored at a time. The discrete solution to (6.9) is formed by extracting the last stage–values from each time step, $x(t_k) = y_{s,k}$.

One subtlety occurring here is the specification of the initial condition. Because we have required the Runge–Kutta methods to satisfy $b_j = a_{s,j}$, the numerical integration starting from t_0 only depends on $y_{s,0}$ and not on the entire concatenated stage-value vector $\bar{y}_0$. This greatly simplifies matters, as it would otherwise be necessary, in effect, to specify a number of initial conditions equal to the number of Runge–Kutta stages.

Algorithm 6.1 presents a pseudo–code implementation of the fixed step–size transient analysis algorithm. We have separated the function for one period of transient analysis in

Algorithm 6.2 One Period of DIRK Transient Analysis

```
function TAPERIOD($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$)
    for $k = 1 : N_t$ do
        for $i = 1 : s$ do
            $y_{i,k} \leftarrow y_{s,k-1} + c_i h_k y'_{s,k-1}$        ▷ Initial guess based on linear extrapolation
            $y_{i,k}$, $y'_{i,k}$ ←DIRKNEWTON($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$, $i$, $k$)
        end for
    end for
    return $\{y_{i,k}\}$, $\{y'_{i,k}\}$
end function
```

Algorithm 6.2 because it is reused elsewhere. Algorithm 6.2 is constructed using the Runge–Kutta functions developed in Chapter 5 as building blocks.

6.1.2 Single Shooting Method

The single shooting method is a steady–state analysis algorithm closely related to transient analysis. For problems with a natural steady–state trajectory, the quantity $r\left(t\right) = x\left(t\right) - x\left(t - T\right)$ represents a measure of the distance of $x(t)$ from the steady-state solution. Forcing $r(t)$ to zero implies that $x\left(t\right) = x\left(t - T\right)$. That is, $x\left(t\right)$ is on the steady–state solution manifold. The single shooting method is simply the application of Newton's method to $r\left(T\right)$ with an unknown initial condition $x\left(0\right)$ and $x\left(T\right)$ implicitly defined by one period of transient analysis. In other words, we are searching for a solution $x\left(t\right)$ of (4.72) that obeys the constraint $x\left(0\right) = x\left(T\right)$. Whereas transient analysis is an initial value problem, the single shooting method solves a two–point boundary value problem on the interval $[0, T]$ with periodic boundary conditions.

For an arbitrary initial condition, $x\left(T\right)$ is implicitly a function of $x\left(0\right)$ through (4.72). For any two times t_1 and t_2, and a given solution value $x\left(t_1\right)$, we write the corresponding value of the solution $x\left(t_2\right)$ as

$$x\left(t_2\right) = \Phi\left[t_1, t_2, x\left(t_1\right)\right] = x\left(t_1\right) + \int_{t_1}^{t_2} x'\left(\tau\right) d\tau \tag{6.4}$$

where the function Φ is referred to as the nonlinear state–transition function. The function Φ outputs the solution to the initial value problem

$$\begin{aligned} \texttt{Solve} \;\; & Cx' + g\left(t, x\right) = f\left(t\right), \\ \texttt{for} \;\; & x\left(t_2\right), \\ \texttt{given} \;\; & x\left(t_1\right). \end{aligned} \tag{6.5}$$

One period of transient analysis over the interval $[0, T]$ can be expressed as

$$x\left(T\right) = \Phi\left[0, T, x\left(0\right)\right]. \tag{6.6}$$

If we substitute the periodic boundary constraint $x\left(T\right) = x\left(0\right)$ into (6.6), we arrive at a nonlinear equation in $x\left(0\right)$, which can be solved for the initial condition coincident with the steady–state solution,

$$x\left(0\right) = \Phi\left[0, T, x\left(0\right)\right]. \tag{6.7}$$

Of course, Φ is not available explicitly. Using numerical integration to perform one period of transient analysis implicitly defines a discrete approximation of $\Phi\left[0, T, x\left(0\right)\right]$.

For one step of numerical integration from t_{k-1} to t_k, we will write the solution using the nonlinear state–transition function as

$$y_{s,k} = \Phi\left[t_{k-1}, t_k, y_{s,k-1}\right] = y_{s,k-1} + h_k b_1 y'_{1,k-1} + h_k \sum_{j=1}^{s} b_{j+1} y'_{j,k}. \tag{6.8}$$

In analogy with the continuous time statement, Φ is now defined as producing the output of the initial–value problem

$$\begin{aligned} \texttt{Solve} \;\; & \frac{1}{h_k}\bar{C}_d \bar{y}_k + \bar{g}_k = \bar{f}_k + \frac{1}{h_k}\bar{C}_p y_{s,k-1} + \bar{I}_q\left(\bar{f}_{k-1} - \bar{g}_{k-1}\right), \\ \texttt{for} \;\; & y_{s,k}, \\ \texttt{given} \;\; & y_{s,k-1}. \end{aligned} \tag{6.9}$$

The numerical integration procedure gives a piecewise definition of Φ over an arbitrary interval spanned by multiple time steps by repeatedly composing (6.8). For example, over the interval $[t_m, t_n]$, the numerical integration procedure can be expanded recursively as

$$\begin{aligned} y_{s,n} &= \Phi\left[t_m, t_n, y_{s,m}\right] \\ &= \Phi\left[t_{n-1}, t_n, \Phi\left[t_m, t_{n-1}, y_{s,m}\right]\right] \\ &= \Phi\left[t_{n-1}, t_n, \Phi\left[t_{n-2}, t_{n-1}, \ldots \Phi\left[t_m, t_{m+1}, y_{m,s}\right]\right]\right]. \end{aligned} \tag{6.10}$$

One period of numerical integration can be expressed in a similar manner to the continuous case,

$$y_{s,N_t} = \Phi\left[0, T, y_{s,0}\right], \tag{6.11}$$

with the understanding that (6.11) masks the underlying recursive definition. The periodic-boundary condition is incorporated into this problem by substituting $y_{s,0}$ for y_{s,N_t}:

$$y_{s,0} = \Phi\left[0, T, y_{s,0}\right]. \tag{6.12}$$

We propose solving (6.12) using Newton's method. For an initial guess of $y_{s,0}$, the residual r is simply

$$r = y_{s,0} - \Phi\left[0, T, y_{s,0}\right] = y_{s,0} - y_{s,N_t}. \tag{6.13}$$

With a slight abuse of notation, we conflate the exact solution of (6.12), $y_{s,0}$, with an arbitrary Newton iterate. The Jacobian for this problem is obtained by taking the partial derivative of r with respect to $y_{s,0}$,

$$\begin{aligned} \frac{\partial r}{\partial y_{s,0}} &= I - \frac{\partial \Phi\left[0, T, y_{s,0}\right]}{\partial y_{s,0}} \\ &= I - \frac{\partial y_{s,N_t}}{\partial y_{s,0}}. \end{aligned} \tag{6.14}$$

The correction term δ is calculated as the solution to the equation

$$\left[I - \frac{\partial y_{s,N_t}}{\partial y_{s,0}}\right]\delta = r. \tag{6.15}$$

The term $\frac{\partial y_{s,N_t}}{y_{s,0}}$ can be expanded using the chain rule applied to the recursive definition of Φ in (6.10):

$$\begin{aligned} \frac{\partial \Phi\left[0, T, y_{s,0}\right]}{\partial y_{s,0}} &= \frac{\partial y_{s,N_t}}{\partial y_{s,0}} \\ &= \frac{\partial y_{s,N_t}}{\partial y_{s,N_t-1}} \cdot \frac{\partial y_{s,N_t-1}}{\partial y_{s,N_t-2}} \cdots \frac{\partial y_{s,1}}{\partial y_{s,0}} \\ &= \prod_{k=1}^{N_t} \frac{\partial y_{s,k}}{\partial y_{s,k-1}}. \end{aligned} \tag{6.16}$$

The matrix $\frac{\partial y_{s,k}}{\partial y_{s,k-1}}$ is implicitly defined by linearizing (5.48) around $y_{s,k-1}$ for fully–implicit methods,

$$\left[\frac{1}{h_k}\bar{C}_d + \bar{G}_k\right]\frac{\partial \bar{y}_k}{\partial y_{s,k-1}} = \left[\frac{1}{h_k}\bar{C}_p - \bar{I}_q\bar{G}_{k-1}\right]\frac{\partial \bar{y}_{k-1}}{\partial y_{s,k-1}}, \tag{6.17}$$

where

$$\left\{\frac{\partial \bar{y}_{k-1}}{\partial y_{s,k-1}}\right\}_i = \begin{cases} 0 & i < s \\ I & i = s \end{cases} \tag{6.18}$$

or (5.57) for diagonally–implicit methods,

$$\left[\frac{d_{i,i}}{h_k}C + G\left(t_{i,k}, y_{i,k}\right)\right]\frac{\partial y_{i,k}}{\partial y_{s,k-1}} = \frac{p_i}{h_k}C - q_i G\left(t_{k-1}, y_{s,k-1}\right) - \sum_{j=1}^{i-1}\frac{d_{i,j}}{h_k}C\frac{\partial y_{j,k}}{\partial y_{s,k-1}}. \tag{6.19}$$

The complex expressions for the single shooting Jacobian reveal a fundamental difficulty of the method; explicitly calculating $\frac{\partial r}{\partial y_{s,0}}$ using the expression in (6.16) requires the solution of $s \times N_t$ matrix–matrix equations. The cost of performing these calculations for even the smallest N_t quickly makes this approach abysmally inefficient, even compared to transient analysis. The key to resolving this issue is to use an iterative method such as GMRES that only requires calculating matrix–vector products with the Jacobian.

Let $z_{s,0}$ be an arbitrary vector of appropriate dimension. We are interested in calculating matrix–vector products of $z_{s,0}$ with the Jacobian as

$$\begin{aligned}\frac{\partial r}{\partial y_{s,0}} z_{s,0} &= z_{s,0} - \frac{\partial y_{s,N_t}}{\partial y_{s,0}} z_{s,0} \\ &= z_{s,0} - z_{s,N_t}.\end{aligned} \tag{6.20}$$

The vectors $z_{i,k}$ are defined recursively as

$$z_{i,k} = \frac{\partial y_{i,k}}{\partial y_{s,k-1}} z_{s,k-1}, \tag{6.21}$$

which can be calculated for fully–implicit methods by solving

$$\left[\frac{1}{h_k}\bar{C}_d + \bar{G}_k\right] \bar{z}_k = \left[\frac{1}{h_k}\bar{C}_p - I_q \bar{G}_{k-1}\right] \bar{z}_{k-1}, \tag{6.22}$$

or for diagonally–implicit methods by solving

$$\left[\frac{d_{i,i}}{h_k} C + G\left(t_{i,k}, y_{i,k}\right)\right] z_{i,k} = \left[\frac{p_i}{h_k} C - q_i G\left(t_k, y_{s,k-1}\right)\right] z_{s,k-1} - \sum_{j=1}^{i-1} \frac{d_{i,j}}{h_k} C z_{j,k}. \tag{6.23}$$

Compared to explicitly forming the Jacobian, one matrix–vector product requires the solution of $s \times N_t$ matrix–vector equations. The efficiency of the single shooting method depends on the total number of matrix–vector products required for the iterative method to converge. This number is quite small for many problems of interest.

Normally we would be concerned about the convergence rate of GMRES when applied to a unpreconditioned system. However, the single shooting Jacobian has a structure that typically results in fast GMRES convergence without preconditioning [86, 87, 117]. This can be understood by reading the expression for the Jacobian in (6.14) as the identity matrix plus a perturbation term. When the perturbation is small in the sense that its eigenvalues have modulus less than and bounded away from 1, the Jacobian is close to the identity

Algorithm 6.3 Fixed Step Size Single Shooting Method

```
function SNFixed($y_{s,0}$, $\{t_k\}$, $\epsilon$, $N_{max}$)
    $\{y_{i,k}\}$, $\{y'_{i,k}\}$ ←TAPeriod($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$)
    $r \leftarrow y_{s,0} - y_{s,N_t}$
    $N_{iter} \leftarrow 1$
    while $\|r\| \geq \epsilon \|y_{s,N_t}\|$ and $N_{iter} \leq N_{max}$ do
        $\delta \leftarrow \left[I - \frac{\partial y_{s,N_t}}{\partial y_{s,0}}\right]^{-1} r$        ▷ Solve using GMRES, SSMVP
        $y_{s,0} \leftarrow y_{s,0} - \delta$

        $\{y_{i,k}\}$, $\{y'_{i,k}\}$ ←TAPeriod($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$)
        $r \leftarrow y_{s,0} - y_{s,N_t}$
        $N_{iter} \leftarrow N_{iter} + 1$
    end while
    return $\{y_{s,k}\}$, $\{y'_{s,k}\}$
end function
```

matrix and fast convergence can be expected without preconditioning.

Algorithm 6.3 is a pseudo–code implementation of the fixed step size single shooting algorithm. Comparing Algorithm 6.3 and the fixed step size transient algorithm 6.1, it is plain that they have a very similar structure. The main difference occurs in the update of the initial condition $y_{s,0}$ using Newton's method.

Algorithm 6.4 is a pseudo–code implementation of the matrix-vector product procedure required for solving (6.15) using GMRES. This algorithm is equivalent to performing one period of transient analysis of a linear time–varying system starting with the initial condition $z_{s,0}$. The equivalent system is time–varying due to the dependence of the matrices $G\left(t_{i,k}, y_{i,k}\right)$.

6.1.3 Multiple Shooting Method

The first direct steady-state finite–element steady–state simulation algorithm applied to the magnetic diffusion equation capable of handling rotational motion and nonlinear materials in a general way was termed the the time–periodic finite–element method [52]. This strategy for solving boundary–value problems is more generally known as the direct multiple shooting method. In contrast to the single shooting method, the multiple shooting method searches

Algorithm 6.4 DIRK Single Shooting Jacobian Matrix–Vector Product

```
**function** SNMVP($z_{s,0}$, $\{y_{i,k}\}$, $\{t_k\}$)
    **for** $k = 1 : N_t$ **do**
        **for** $i = 1 : s$ **do**
            $v \leftarrow p_i z_{s,k-1}$
            **for** $j = 1 : (i-1)$ **do**
                $v \leftarrow v - z_{j,k}$
            **end for**
            $v \leftarrow h_k^{-1} C v - q_i G\left(t_k, y_{s,k-1}\right) z_{s,k-1}$
            $z_{i,k} \leftarrow \left[h_k^{-1} d_{i,i} C + G\left(t_{i,k}, y_{i,k}\right)\right]^{-1} v$
        **end for**
    **end for**
    **return** $z_{s,0} - z_{s,N_t}$
**end function**
```

simultaneously for the vectors $\bar{y}_k$ satisfying (5.48) at all times, in addition to the two–point boundary–value constraint $\bar{y}_0 = \bar{y}_{N_t}$. The resulting system of equations can be viewed as N_t coupled initial–value problems defined over the intervals $[t_{k-1}, t_k]$. The method becomes time–periodic due to the periodicity constraint.

The distinct character of the multiple shooting method can be made more clear by concatenating all of the $\bar{y}_k$ into a single vector $\bar{\bar{y}}$ where

$$\{\bar{\bar{y}}\}_k = \bar{y}_k. \tag{6.24}$$

Similar vectors $\bar{\bar{g}}$ and $\bar{\bar{f}}$ are constructed by concatenating the values of $\bar{g}_k$ and $\bar{f}_k$, respectively, where

$$\{\bar{\bar{g}}\}_k = \bar{g}_k + \bar{I}_q \bar{g}_{k-1}, \tag{6.25}$$

$$\{\bar{\bar{f}}\}_k = \bar{f}_k + \bar{I}_q \bar{f}_{k-1}. \tag{6.26}$$

Now (6.3) implicitly defines a nonlinear vector equation

$$\bar{\bar{C}}\bar{\bar{y}} + \bar{\bar{g}} = \bar{\bar{f}} \tag{6.27}$$

where the nonzero matrix entries of $\bar{\bar{C}}$ are given by

$$\begin{aligned}\{\bar{\bar{C}}\}_{k,k} &= \frac{1}{h_k}\bar{C}_d \\ \{\bar{\bar{C}}\}_{k,k-1} &= -\frac{1}{h_k}\bar{C}_p\end{aligned} \tag{6.28}$$

and the subscript $k-1$ is taken modulo N_t. In particular, this means

$$\{\bar{\bar{C}}\}_{1,N_t} = -\frac{1}{h_1}\bar{C}_p \tag{6.29}$$

due to the periodic boundary condition.

We solve (6.27) using Newton's method. This results in the linear equation

$$\left[\bar{\bar{C}} + \bar{\bar{G}}\right]\bar{\bar{\delta}} = \bar{\bar{r}} \tag{6.30}$$

where $\bar{\bar{G}}$ is a matrix resulting from the linearization of $\bar{\bar{g}}$ with nonzero entries

$$\begin{aligned}\{\bar{\bar{G}}\}_{k,k} &= \bar{G}_k, \\ \{\bar{\bar{G}}\}_{k,k-1} &= \bar{I}_q\bar{G}_{k-1}\end{aligned} \tag{6.31}$$

where $k-1$ is taken modulo N_t and the residual $\bar{\bar{r}}$ is given by

$$\bar{\bar{r}} = \bar{\bar{C}}\bar{\bar{y}} + \bar{\bar{g}} - \bar{\bar{f}}. \tag{6.32}$$

With a slight abuse of notation, we conflate the exact solution of (6.27) with an arbitrary Newton iterate. The correction $\bar{\bar{\delta}}$ is calculated by solving (6.30) and the solution is updated as

$$\bar{\bar{y}} \leftarrow \bar{\bar{y}} - \bar{\bar{\delta}}. \tag{6.33}$$

A fundamental difficulty of solving (6.30) is the large number of equations: The unknown vector $\bar{\bar{\delta}}$ is N_t times larger than the unknown vector for a single time. Most of the techniques investigated for solving this type linear equation revolve around relaxation schemes exploiting the block–structure of the matrix [53, 54, 55, 56, 57]. For example, if the upper–right hand block containing the matrix $\{\bar{\bar{C}}+\bar{\bar{G}}\}_{1,N_t}$ is set to zero, then the overall matrix has block–lower triangular form. An efficient relaxation method can be developed around this structure.

Alternatively, if it happened to be the case that all of the diagonal blocks were the same, $\bar{G}_1 = \bar{G}_2 = \cdots = \bar{G}_{N_t}$, then the matrix would be block–circulant. A useful property of circulant matrices is that they can be diagonalized by discrete Fourier transform matrix. Another way of saying this is that an M dimensional circulant matrix has eigenvectors defined by $\{v_k\}_m = e^{jk\frac{m-1}{M}}$. Block–circulant matrices have a similar property in a block–wise sense. If the time variation in the G matrices are due to nonlinear materials and not motion, the nearly–circulant property can be used effectively [58, 59]. Unfortunately for problems with motion, essential information about the motion tends to be discarded whenever a circulant approximation is made. This is because the average coupling between two variables, one in a fixed and another in a rotating reference frame, is zero.

Instead of a relaxation scheme, we will use GMRES to solve (6.30). Unlike the single shooting method, we will require a preconditioner in this case. Because we are primarily interested in problems containing rotational motion, we adopt the block lower–triangular Jacobian approximation as our preconditioner. In particular, the preconditioner $\bar{\bar{M}}$ has nonzero entries given by

$$\{\bar{\bar{M}}\}_{i,j} = \begin{cases} \{\bar{\bar{C}} + \bar{\bar{G}}\}_{i,j}, & i \leq j \\ 0, & j > i \end{cases} \tag{6.34}$$

Because the only entry in the blockwise upper–triangular part of the Jacobian occurs in the $(1, N_t)$ position, the preconditioner is obtained through a very minor structural modification.

Algorithm 6.5 contains a pseudo–code implementation of the fixed step size multiple shooting method. The function is simply an implementation of Newton's method applied to (6.27). The function in Algorithm 6.6 calculates the residual for the method. The linear equation on line 5 is solved using GMRES. The multiple shooting method avoids transient

Algorithm 6.5 Fixed Step Size Multiple Shooting Method

```
function MSFixed($\{y_{i,k}\}$, $\{t_k\}$, $\epsilon$, $N_{max}$)
    $\{r_{i,k}\}$, $\{y'_{i,k}\}$, $\varepsilon \leftarrow$MSResidual($\{y_{i,k}\}$, $\{t_k\}$)
    $N_{iter} \leftarrow 0$
    while $\varepsilon > \epsilon$ and $N_{iter} \leq N_{max}$ do
        $\bar{\bar{\delta}} \leftarrow \left[\bar{\bar{C}} + \bar{\bar{G}}\right]^{-1} \bar{\bar{r}}$        ▷ Solve using GMRES, MSMVP, MSPC
        $\bar{\bar{y}} \leftarrow \bar{\bar{y}} - \bar{\bar{\delta}}$
        $\{r_{i,k}\}$, $\{y'_{i,k}\}$, $\varepsilon \leftarrow$MSResidual($\{y_{i,k}\}$, $\{t_k\}$)
        $N_{iter} \leftarrow N_{iter} + 1$
    end while
    return $\{y_{i,k}\}$, $\{y'_{i,k}\}$
end function
```

analysis in calculating the residual so it is not possible to reuse any of the previously developed function in this implementation. Because transient analysis requires solving many linear equations for each time step, it is reasonable to expect that the residual calculation in Algorithm 6.6 is much faster than one period of transient analysis.

Algorithm 6.7 contains pseudo–code for calculating matrix-vector products with the matrix $\bar{\bar{C}} + \bar{\bar{G}}$ for an input vector $\bar{\bar{z}}$. Algorithm 6.8 contains pseudo–code for calculating application of the preconditioner $\bar{\bar{M}}$. It is worthwhile to compare Algorithm 6.8 to the matrix–vector product code in Algorithm 6.4 for the single shooting method to understand how they are similar in many respects. From a computational complexity standpoint, one application of the preconditioner in Algorithm 6.8 is about as expensive as one matrix–vector product in Algorithm 6.4. This can be understood by realizing that the main computational bottleneck in both functions occur when solving linear equations with the matrices $\frac{d_{i,i}}{h_k}C + G\left(t_{i,k}, y_{i,k}\right)$. Therefore, any differences in simulation time between the methods depend upon the total number of Newton iterations, cost of the residual calculation, and total number of GMRES iterations. The difference does not depend on the complexity of the individual matrix–vector products or preconditioner calculations when solving the linear equations for Newton's method.

Algorithm 6.6 DIRK Multiple Shooting Residual

```
function MSResidual($\{y_{i,k}\}$, $\{t_k\}$)
    $\alpha \leftarrow 0$
    $\beta \leftarrow 0$
    for $k = 1 : N_t$ do
        $h_k \leftarrow t_k - t_{k-1}$
        $y'_{s,k} \leftarrow -p_s y_{s,k-1}$                    ▷ $q_s = 0$ for L–Stable methods
        for $i = 1 : s$ do
            $y'_{s,k} \leftarrow y'_{s,k} + d_{s,i} y_{i,k}$
            $r_{i,k} \leftarrow g\left(t_{i,k}, y_{i,k}\right) - f\left(t_{i,k}\right) + q_i\left[g\left(t_{k-1}, y_{s,k-1}\right) - f\left(t_{k-1}\right)\right]$
            for $j = 1 : i$ do
                $r_{i,k} \leftarrow r_{i,k} + \frac{d_{i,j}}{h_k} C y_{j,k}$
            end for
            $\beta \leftarrow \beta + \|f\left(t_{i,k}\right)\|^2$
            $\alpha \leftarrow \alpha + \|r_{i,k}\|^2$
        end for
        $y'_{s,k} \leftarrow h_k^{-1} y'_{s,k}$
    end for
    for $k = 1 : N_t$ do
        $h_k \leftarrow t_k - t_{k-1}$
        for $i = 1 : (s-1)$ do
            $y'_{i,k} \leftarrow -p_i y_{s,k-1}$
            for $j = 1 : i$ do
                $y'_{i,k} \leftarrow y'_{i,k} + d_{i,j} y_{j,k}$
            end for
            $y'_{i,k} \leftarrow h_k^{-1} y'_{i,k} - q_i y'_{s,k-1}$
        end for
    end for
    $\varepsilon \leftarrow \alpha^{-1}\beta$
    $\varepsilon \leftarrow \varepsilon^{\frac{1}{2}}$
    return $\{r_{i,k}\}$, $\{y'_{i,k}\}$, $\varepsilon$
end function
```

Algorithm 6.7 DIRK Multiple Shooting Matrix–Vector Product

```
function MSMVP($\{z_{i,k}\}$, $\{y_{i,k}\}$, $\{t_k\}$)
    for $k = 1 : N_t$ do
        $h_k \leftarrow t_k - t_{k-1}$
        for $i = 1 : s$ do
            $r_{i,k} \leftarrow -p_i C z_{s,k-1}$
            for $j = 1 : i$ do
                $r_{i,k} \leftarrow r_{i,k} + d_{i,j} C z_{j,k}$
            end for
            $r_{i,k} \leftarrow h_k^{-1} r_{i,k} + q_i G\left(t_{k-1}, y_{s,k-1}\right) z_{s,k-1} + G\left(t_{i,k}, y_{i,k}\right) z_{i,k}$
        end for
    end for
    return $\{r_{i,k}\}$
end function
```

Algorithm 6.8 DIRK Multiple Shooting Preconditioner

```
function PC($\{z_{i,k}\}$, $\{y_{i,k}\}$, $\{t_k\}$)
    $k \leftarrow 1$
    $h_k \leftarrow t_k - t_{k-1}$
    for $i = 1 : s$ do
        $z_{i,k} \leftarrow \left[h_k^{-1} d_{i,i} C + G\left(t_{i,k}, y_{i,k}\right)\right]^{-1} z_{i,k}$
        for $j = (i+1) : s$ do
            $z_{j,k} \leftarrow z_{j,k} - h_k^{-1} d_{j,i} C z_{i,k}$
        end for
    end for

    for $k = 2 : N_t$ do
        $h_k \leftarrow t_k - t_{k-1}$
        for $i = 1 : s$ do
            $z_{i,k} \leftarrow z_{i,k} + h_k^{-1} p_i C z_{s,k-1} - q_i G\left(t_{k-1}, y_{s,k-1}\right) z_{s,k-1}$
        end for

        for $i = 1 : s$ do
            $z_{i,k} \leftarrow \left[h_k^{-1} d_{i,i} C + G\left(t_{i,k}, y_{i,k}\right)\right]^{-1} z_{i,k}$
            for $j = (i+1) : s$ do
                $z_{j,k} \leftarrow z_{j,k} - h_k^{-1} d_{j,i} C z_{i,k}$
            end for
        end for
    end for

    return $\{z_{i,k}\}$
end function
```

6.1.4 Adaptive Solution Refinement

A statement of the adaptive step size problem is similar to the fixed step size case except the set of time points $\{t_k\}$ is not specified ahead of time. For true initial–value problems, the standard method of adaptive time–marching is eminently reasonable. This is so because, presumably, one is interested in the solution at every time in the simulation interval. Requiring the solution to meet some local error estimate threshold at every time step is a sensible way to control the local accuracy over the entire interval. It is certainly possible to naïvely apply the classic adaptive time–marching algorithms to the steady–state simulation problem as stated in (6.1).

When the steady–state part of the solution is the main item of interest, the initial value problem is simply a means to an end. Requiring the solution to meet a uniform error estimate threshold over the entire interval incurs an unnecessary high computational cost when the transient part of the solution is non–negligible. This problem can be avoided by adopting a multigrid–like approach to determining the solution within a given tolerance. Such a strategy starts by solving the steady–state problem on a coarse discretization of the time axis. Local error estimates are calculated and the grid is refined based on these indicators. The solution from the coarse grid can be interpolated onto the refined grid to serve as a good initial condition and the procedure continues iteratively.

The standard analysis of multigrid algorithms operates under three assumptions: (1) the same number of "smoothing" iterations are performed on each grid; (2) the is solved exactly on the coarsest grid with a negligible amount of work compared to the finest grid; and (3) each refinement produces a grid with twice the number of variables. In this case, the total work is approximately equal to twice the work done on the finest grid [118]. In our context, a smoothing iteration corresponds to one period of transient analysis or one Newton iteration for the shooting methods. Generally, on all but the coarsest grid, the maximum number of smoothing iterations performed is small.[1] The next section introduces a step size selection

[1] This is usually in the context of linear multigrid methods, where relaxation schemes are used whose

algorithm based on the error indicators of Chapter 5. The subsequent section discusses its use in adaptively refining the solution in the manner previously described.

A Continuously Variable Step Size Selection Algorithm

Suppose after solving the steady–state problem on some fixed grid, we have some error indicator ϵ_k for the interval $[t_{k-1}, t_k]$. For a method of order p, we have seen from Chapter 5 that, with an appropriate choice of lower order solution, the quantity $(y_{s,k} - \hat{y}_{s,k}) \frac{(p-1)!}{h_k^{p-1}}$ is an $\mathcal{O}(h_k)$ approximation of the $(p-1)^{\text{th}}$ derivative of the solution over this interval. For small enough step sizes, the quantity $\epsilon_k h_k^{p-1}$ will be approximately constant independent of h_k. Therefore, if we wish to select an interval $[s_{l-1}, s_l]$ such that the local error is less than ε, we can estimate the required interval length as satisfying the inequality

$$s_l - s_{l-1} \leq \min_{k \in \mathcal{K}} \mathcal{H}_k\left(\varepsilon\right) \tag{6.35}$$

where

$$\mathcal{H}_k\left(\varepsilon\right) = h_k \left[\frac{\varepsilon}{\epsilon_k}\right]^{\frac{1}{p-1}} \tag{6.36}$$

and $\mathcal{K}$ is the set of all k satisfying $s_{l-1} \leq t_k$ and $t_{k-1} \leq s_l$. That is, $\mathcal{K}$ is the smallest set of integers such that $t_{k_{min}} \leq s_{l-1} < s_l \leq t_{k_{max}}$ where $k_{min} = \min \mathcal{K}$ and $k_{max} = \max \mathcal{K}$.

If we fix, for example, $s_0 = 0$, an entire set of new times $\{s_l\}$ can be chosen to satisfy (6.35) inductively. If we regard $\mathcal{H}\left(t_k\right) = \mathcal{H}_k$ as a piecewise constant function of time, then the local minima and maxima of the function govern the range of locally permissible step sizes. Figure 6.1 gives an example of one such function. The set of step sizes satisfying this upper bound was generated by Algorithm 6.9.

Algorithm 6.9 works by partitioning $\mathcal{H}\left(t\right)$ over intervals where it is either non–increasing or non–decreasing. On each of these intervals, we start from the local maximum and construct

convergence rate slows quite dramatically after just a few iterations. In fact, transient analysis has much the same problem. In the context of Newton's method to solve either of the shooting problems, having a good initial condition means that performing one or two iterations will produce a quite good approximation on that grid.

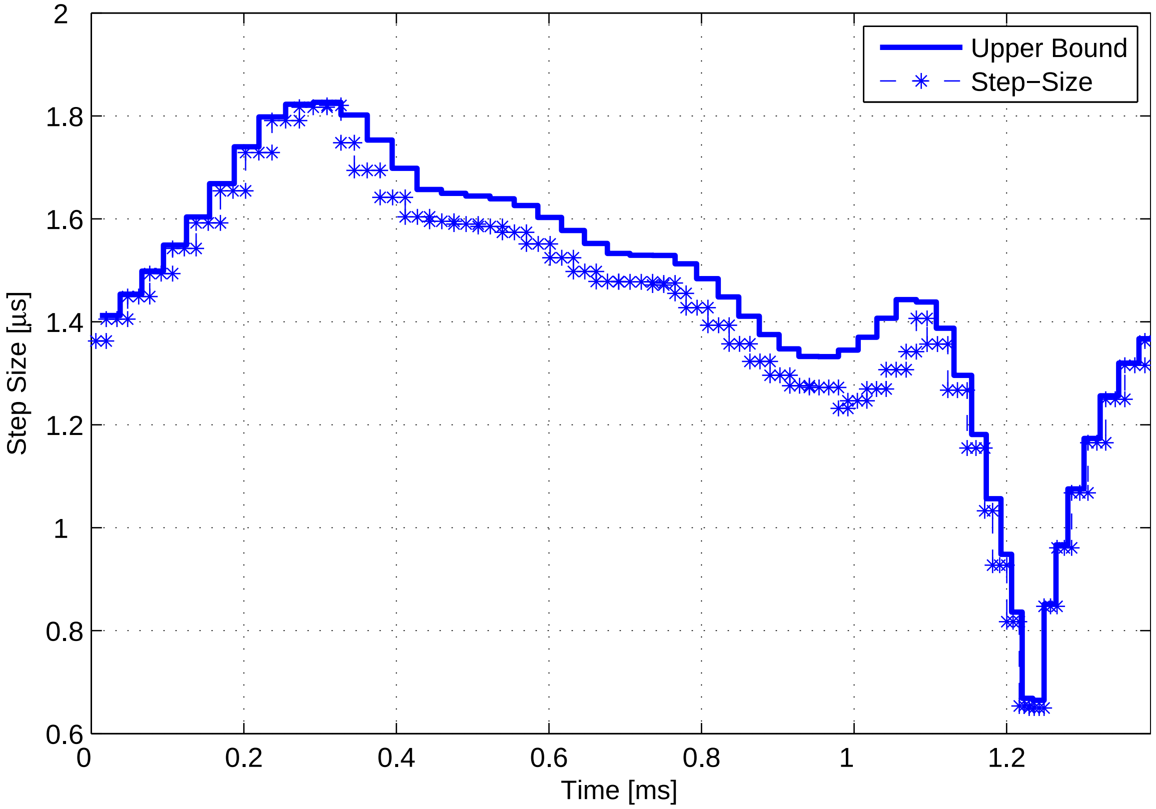

Figure 6.1: Upper bound on the step size determined by the error indicators and the set of step sizes generated by Algorithm 6.9.

a sequence of times moving toward the local minimum such that (6.35) is satisfied. Once the final new time generated in this manner falls outside the local partition, the sequence of times is scaled so that the smallest and largest times coincide with the endpoints of the partition.

A Generic Adaptive Framework

Implementing the standard approach of approximately doubling the number of grid points every iteration is simply a matter of reducing the tolerance ε used to calculate $\mathcal{H}_k$ during the grid refinement procedure by a factor of 2^{1-p}. The only remaining issue is how to pick the initial value of ε. In fact, the coarsest grid will be fixed in advanced and the solution on this grid will be determined to within a tolerance of ϵ_{exact}. From this solution, the initial tolerance ε_{init} that would have resulted in this set of time–steps can be estimated as the average of the error indicators:

$$\varepsilon_{init} = \frac{1}{N_t} \sum_{k=1}^{N_t} \epsilon_k. \tag{6.37}$$

In general, we may have in mind some tolerance ε_{max} that we would like to bound all of the error estimates on the finest grid. Assuming the total number of time steps doubles after each refinement, the total number of grids N_g can be estimated as satisfying

$$\varepsilon_{max} = \varepsilon_{init} 2^{(N_g-1)(1-p)}, \tag{6.38}$$

or

$$N_g = \left\lfloor \frac{\log \varepsilon_{max} - \log \varepsilon_{init}}{(1-p)\log 2} \right\rfloor. \tag{6.39}$$

Therefore, the tolerance for the first refinement should be set to $\varepsilon_{init} = \varepsilon_{max} 2^{(N_g-1)(p-1)}$. In reality, it is possible we will just fail to satisfy the tolerance threshold on the N_g^{th} grid, so we multiply ε_{init} by a safety factor of 0.5. The full adaptive procedure is represented in Algorithm 6.10. The function `FixedStepSizeAlgorithm` is a generic placeholder for which any of the previously developed time–domain steady–state algorithms can be substituted.

Algorithm 6.9 Time-Axis Grid Refinement

```
function RKRefine($\{\varepsilon_k\}$,$\{t_k\}$,$\varepsilon$)
    for $k = 1 : N_t$ do
        $\mathcal{H}_k \leftarrow \left[\frac{\varepsilon}{\varepsilon_k}\right]^{\frac{1}{p-1}} (t_k - t_{k-1})$
        $\mathcal{H}_{k+Nt} \leftarrow \mathcal{H}_k$
    end for
    $k_{max}$ ←FindMax($\{\mathcal{H}_k\}$,1)
    $k_{min}$ ←FindMin($\{\mathcal{H}_k\}$,$k_{max}$)
    $k_{max}$ ←FindMax($\{\mathcal{H}_k\}$,$k_{min}$)
    $\{s_l\}$ ←RefinePartition($k_{min}$, $k_{max}$, $\{\mathcal{H}_k\}$, $\{t_k\}$)
    while $\max_l s_l - \min_l s_l < T$ do
        if $k_{min} < k_{max}$ then
            $k_{min}$ ←FindMin($\{\mathcal{H}_k\}$,$k_{max}$)
        else
            $k_{max}$ ←FindMax($\{\mathcal{H}_k\}$,$k_{min}$)
        end if
        $\{s_l\} \leftarrow \{s_l\} \bigcup$RefinePartition($k_{min}$, $k_{max}$, $\{\mathcal{H}_k\}$, $\{t_k\}$)
    end while
    $\{s_l\}$ ←Modulo($\{s_l\}$, $T$)
    $\{s_l\}$ ←Sort($\{s_l\}$)
    return $\{s_l\}$
end function

function FindMin($\{\mathcal{H}_k\}$,$\{t_k\}$,$k$)
    while $\mathcal{H}_k \geq \mathcal{H}_{k+1}$ do
        $k \leftarrow k + 1$
    end while
    return $k$
end function

function FindMax($\{\mathcal{H}_k\}$,$\{t_k\}$,$k$)
    while $\mathcal{H}_k \leq \mathcal{H}_{k+1}$ do
        $k \leftarrow k + 1$
    end while
    return $k$
end function
```

```
function RefinePartition($k_{min}$, $k_{max}$, $\{\mathcal{H}_k\}$, $\{t_k\}$)
    $i \leftarrow 1$
    $s_i \leftarrow t_{k_{max}}$
    $k \leftarrow k_{max}$
    if $k_{max} < k_{min}$ then
        while $s_i < t_{k_{min}}$ do
            while $t_k - s_i < \mathcal{H}_k$ do
                $k \leftarrow k + 1$
            end while
            if $t_{k-1} - s_i > H_k$ then
                $k \leftarrow k - 1$
            end if
            $s_{i+1} \leftarrow s_i + \mathcal{H}_k$
            $i \leftarrow i + 1$
        end while
    else
        while $s_i > t_{k_{min}}$ do
            while $s_i - t_{k-1} < \mathcal{H}_k$ do
                $k \leftarrow k - 1$
            end while
            if $s_i - t_k > \mathcal{H}_k$ then
                $k \leftarrow k + 1$
            end if
            $s_{i+1} \leftarrow s_i - \mathcal{H}_k$
            $i \leftarrow i + 1$
        end while
    end if
    for $j = 1 : i$ do
        $s_j \leftarrow (s_j - t_{k_{max}}) \frac{t_{k_{max}} - t_{k_{min}}}{t_{k_{max}} - s_i} + t_{k_{max}}$
    end forreturn $\{s_i\}$
end function
```

Algorithm 6.10 Generic Adaptive Steady-State Algorithm

```
function SSAdaptive($\{y_{s,k}\}$, $\{t_k\}$, $\varepsilon_{max}$, $\epsilon_{exact}$, $\epsilon_{smooth}$, $N_{smooth}$)
    $\{y_{i,k}\}$, $\{y'_{i,k}\}$ ←FixedStepSizeAlgorithm($\{y_{i,k}\}$, $\{t_k\}$, $\epsilon_{exact}$, $\infty$)
    $\{\varepsilon_k\}$ ←RKError($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$)
    $\varepsilon \leftarrow N_t^{-1} \sum_{k=1}^{N_t} \varepsilon_k$
    $N_g \leftarrow \left\lfloor ([1-p]\log 2)^{-1} (\log \varepsilon_{max} - \log \varepsilon) \right\rfloor$
    $\varepsilon \leftarrow 0.5 \varepsilon_{max} 2^{N_g(p-1)}$
    while $\varepsilon_{max} > \max_k \varepsilon_k$ do
        $\varepsilon \leftarrow \varepsilon 2^{1-p}$
        $\{s_k\}$ ←RKRefine($\{\varepsilon_k\}$, $\{t_k\}$, $\varepsilon$)
        $\{y_{i,k}\}$ ←RKInterpolate($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$, $\{s_k\}$)
        $\{t_k\} \leftarrow \{s_k\}$
        $\{y_{i,k}\}$, $\{y'_{i,k}\}$ ←FixedStepSizeAlgorithm($\{y_{i,k}\}$, $\{t_k\}$, $\epsilon_{smooth}$, $N_{smooth}$)
        $\{\varepsilon_k\}$ ←RKError($\{y_{i,k}\}$, $\{y'_{i,k}\}$, $\{t_k\}$)
    end while
end function
```

6.2 Harmonic Balance

The harmonic balance method solves (6.1) by expanding the solution as a weighted sum of trigonometric basis functions. In this presentation, we will utilize complex–valued exponential functions with harmonic weights $\tilde{x}_k$ to write the approximate solution. Using $N_t = 2N_h$ basis functions, the approximate solution can be written as

$$x = \frac{1}{\sqrt{N_t}} \sum_{\substack{j=1 \\ j \neq N_h+1}}^{N_t} \tilde{x}_j e^{i\omega_j t} + \frac{1}{\sqrt{N_t}} \tilde{x}_{N_h+1} \cos\left(N_h \omega_e t\right) \tag{6.40}$$

with

$$\omega_j = \begin{cases} (j-1)\,\omega_e & : j < N_h + 1 \\ 0 & : j = N_h + 1 \\ (j - 1 - N_h)\,\omega_e & : j > N_h + 1 \end{cases} \tag{6.41}$$

where i is the imaginary unit and $\omega_e = \frac{2\pi}{T}$ is the fundamental electrical period of the solution. The approximation of the solution in terms of these basis functions leads directly to an approximation of the solution derivative:

$$x' = \frac{1}{\sqrt{N_t}} \sum_{\substack{j=1 \\ j \neq N_h+1}}^{N_t} i\omega_j \tilde{x}_j e^{i\omega_j t} - \frac{N_h \omega_e}{\sqrt{N_t}} \tilde{x}_{N_h+1} \sin\left(N_h \omega_e t\right) \tag{6.42}$$

The use of an even number of basis functions is motivated by the fact that, later on, we will present an adaptive harmonic balance algorithm by successively doubling the number of sample points. Having an even number of samples, it becomes necessary to deal with the so–called Nyquist element associated with the basis function $\cos\left(N_h \omega_e t\right)$. This is also behind the peculiar definition of ω_{N_h+1}, which is explained below.

If we choose a set of N_t evenly spaced sample times $t_k = (k-1)\,\Delta t$, $\Delta t = \frac{T}{N_t}$, then the harmonic coefficients $\tilde{x}_j$ can be put in one–to–one correspondence with samples of $x\left(t\right)$ in the time domain, denoted $\bar{x}_k$, where

$$\bar{x}_k = \frac{1}{\sqrt{N_t}} \sum_{\substack{i=1 \\ j \neq N_h+1}}^{N_t} \tilde{x}_j e^{i\omega_j t_k} + \frac{1}{\sqrt{N_t}} \tilde{x}_{N_h+1} \cos\left(N_h \omega_e t_k\right). \tag{6.43}$$

The derivative can be similarly sampled,

$$\bar{x}'_k = \frac{1}{\sqrt{N_t}} \sum_{j=1}^{N_t} i\omega_j \tilde{x}_j e^{i\omega_j t_k}. \tag{6.44}$$

The definition $\omega_{N_h+1} = 0$ stems from the fact that the derivative of the Nyquist basis function is zero at the sample points. This choice eliminates special cases that must be dealt with when writing algorithm loops.

The relationship between the harmonic coefficient and sample vectors can be written more concisely by concatenating the sets into large vectors denoted $\tilde{x}$ and $\bar{x}$ respectively, where

$$\{\bar{x}\}_k = \bar{x}_k, \tag{6.45}$$

$$\{\tilde{x}\}_j = \tilde{x}_j. \tag{6.46}$$

The vectors $\tilde{x}$ and $\bar{x}$ are related through a simple linear transformation,

$$\bar{x} = \tilde{\Gamma}^H \tilde{x}. \tag{6.47}$$

The matrix $\tilde{\Gamma}$ is a block–wise unitary discrete–Fourier transform matrix given by

$$\tilde{\Gamma} = \tilde{\gamma} \otimes I \tag{6.48}$$

where $\otimes$ is the Kronecker product operator, I is an identity matrix compatible with the size of x, and $\tilde{\gamma}$ is the scalar DFT matrix with entries

$$\{\tilde{\gamma}\}_{j,k} = \frac{1}{\sqrt{N_t}} e^{-i\omega_j t_k}. \tag{6.49}$$

Equation (6.47) is simply a restatement of (6.43). Similarly, the nonlinearity $g\left(x, t\right)$ and exogenous input $f\left(t\right)$ can be evaluated in the time domain using the sample times and sample solution values to produce sample vectors $\bar{g}_k$ and $\bar{f}_k$. These vectors are also concatenated into larger vectors $\bar{g}$ and $\bar{f}$ respectively:

$$\{\bar{g}\}_k = \bar{g}_k = g\left(\bar{x}_k, t_k\right), \tag{6.50}$$

$$\{\bar{f}\}_k = \bar{f}_k = f\left(t_k\right). \tag{6.51}$$

In turn, the sample vectors implicitly define harmonic coefficient vectors through the block–DFT:

$$\{\tilde{g}\}_j = \{\Gamma^H \bar{g}\}_j = \tilde{g}_j, \tag{6.52}$$

$$\{\tilde{f}\}_j = \{\Gamma^H \bar{f}\}_j = \tilde{f}_j. \tag{6.53}$$

Substituting these relationships into (4.72), we arrive at a set of N_t equations, one for each harmonic coefficient $\tilde{x}_j$,

$$i\omega_j C\tilde{x}_j + \tilde{g}_j = \tilde{f}_j, \tag{6.54}$$

which can be more conveniently written in concatenated form as

$$\tilde{C}\tilde{x} + \tilde{g} = \tilde{f}. \tag{6.55}$$

The matrix $\tilde{C}$ is a block–diagonal matrix with nonzero entries given by

$$\{\tilde{C}\}_{j,j} = i\omega_j C. \tag{6.56}$$

The block–DFT matrix $\tilde{\Gamma}$ can be used to transform (6.55) through change of variable $\tilde{x} = \tilde{\Gamma}\bar{x}$ and pre–multiplication by $\tilde{\Gamma}^H$, into the following equation:

$$\tilde{\Gamma}^H \tilde{C}\tilde{\Gamma}\bar{x} + \bar{g} = \bar{f}. \tag{6.57}$$

Clearly, the solutions of (6.55) and (6.57) are related by the transformation $\tilde{\Gamma}$. In this sense, they are equivalent statements of the harmonic balance problem. The difference is that (6.55) is presented in the frequency domain in terms of harmonic coefficients, while (6.57) is presented in the time domain in terms of solution samples.

Equation (6.57) shows that we have forced the harmonic expansion (6.40) of the solution to solve (4.72) exactly at the sample times. This was effected by putting the harmonic coefficients in one–to–one correspondence with the sample times. This form of the harmonic balance method is a collocation method. This is in contradistinction to a Galerkin method that would force the residual of the solution to be orthogonal to the subspace spanned by the trigonometric basis functions, but would not necessarily solve (4.72) at any of the sample times.

6.2.1 Linearization

Newton's method can be used to solve the harmonic balance equations. It is easiest to start with the time–domain statement of the problem to derive Newton's method and subsequently transform the equations into the frequency–domain. Proceeding along these lines, we employ a slight abuse of notation and conflate $\bar{x}$ with the true solution of (6.57) and an arbitrary Newton iterate. Evaluating (6.57) at the approximate solution yields a residual $\bar{r}$:

$$\bar{r} = \tilde{\Gamma}^H \tilde{C} \tilde{\Gamma} \bar{x} + \bar{g}\left(\bar{x}\right) - \bar{f}. \tag{6.58}$$

The Jacobian for the problem can be thought of as a block matrix with entries defined by

$$\{\bar{J}\}_{j,k} = \frac{\partial \bar{r}_j}{\partial \bar{x}_k}. \tag{6.59}$$

One part of the Jacobian is defined by the matrix $\Gamma^H \tilde{C} \Gamma^H$. The other part is due to the Jacobian of the nonlinear function $\bar{g}$. Since the nonlinearity is assumed to be independent for each time, its contribution to $\bar{J}$ is nonzero only on the diagonal blocks. We define the block–diagonal matrix $\bar{G}$ with nonzero entries given by

$$\{\bar{G}\}_{k,k} = \bar{G}_{k,k} = G\left(\bar{x}_k, t_k\right). \tag{6.60}$$

Now the Jacobian can be simply written as

$$\bar{J} = \tilde{\Gamma}^H \tilde{C} \tilde{\Gamma} + \bar{G}. \tag{6.61}$$

Using this notation, Newton's method can be expressed in the time domain as solving the equation

$$\bar{J}\bar{\delta} = \bar{r}, \tag{6.62}$$

and updating the approximate solution as

$$\bar{x} \leftarrow \bar{x} - \bar{\delta}. \tag{6.63}$$

An equivalent statement of Newton's method in the frequency domain can be obtained using a similarity transformation, this time with the change of variable $\bar{r} = \tilde{\Gamma}^H \tilde{r}$ and pre–multiplication by $\tilde{\Gamma}$. The Jacobian in the frequency domain can be written as

$$\tilde{J} = \tilde{C} + \tilde{\Gamma}\bar{G}\tilde{\Gamma}^H. \tag{6.64}$$

The linear equation to be solved is

$$\tilde{J}\tilde{\delta} = \tilde{r}, \tag{6.65}$$

and the approximate solution is updated as

$$\tilde{x} \leftarrow \tilde{x} - \tilde{\delta}. \tag{6.66}$$

Algorithm 6.11 Harmonic Balance Method

```
function HBFixed($\{\bar{x}_k\}$, $\{t_k\}$, $\epsilon$, $N_{max}$)          ▷ Solve (6.57)
    $\{\bar{r}_k\}$, $\varepsilon$ ←HBRes($\{\bar{x}_k\}$)
    $N_{iter} \leftarrow 0$
    while $\varepsilon > \epsilon$ and $N_{iter} \leq N_{max}$ do          ▷ Solve (6.62)
        $\bar{\delta} \leftarrow \bar{J}^{-1}\bar{r}$          ▷ Use GMRES, HBMVP, HBPC
        $\bar{x} \leftarrow \bar{x} - \bar{\delta}$
        $\{\bar{r}_k\}$, $\{\bar{x}'_k\}$, $\varepsilon$ ←HBRes($\{\bar{x}_k\}$)
        $N_{iter} \leftarrow N_{iter} + 1$
    end while
    return $\{\bar{x}_k\}$
end function
```

Algorithm 6.12 Harmonic Balance Residual

```
function HBRes($\{x_j\}$)                                   ▷ Evaluate (6.58)
    $\alpha \leftarrow 0$
    $\beta \leftarrow 0$
    $\{\tilde{x}_j\} \leftarrow$FFT($\{x_j\}$)
    for $j = 1 : N_t$ do
        $\tilde{r}_j \leftarrow i\omega_j C\tilde{x}'_j$
    end for

    $\{\bar{r}_j\} \leftarrow$IFFT($\{\tilde{r}_j\}$)
    for $k = 1 : N_t$ do
        $\bar{r}_k \leftarrow \bar{r}_k + g\left(\bar{x}_k, t_k\right) - f\left(t_k\right)$
        $\beta \leftarrow \beta + \|f\left(t_k\right)\|^2$
        $\alpha \leftarrow \alpha + \|\bar{r}_k\|^2$
    end for
    $\varepsilon \leftarrow \alpha^{-1}\beta$
    $\varepsilon \leftarrow \varepsilon^{\frac{1}{2}}$
    return $\{\bar{x}_k\}$, $\varepsilon$
end function
```

Algorithm 6.13 Harmonic Balance Matrix-Vector Product

```
function HBMVP($\{\bar{z}_k\}$)                              ▷ MVP used with GMRES
    $\{\tilde{z}_j\} \leftarrow$FFT($\{\bar{z}_k\}$)                          ▷ $\bar{\lambda} = \bar{J}\bar{z}$
    for $j = 1 : N_t$ do
        $\tilde{\lambda}_j \leftarrow i\omega_j C\tilde{z}_j$
    end for
    $\{\bar{\lambda}_k\} \leftarrow$IFFT($\{\tilde{\lambda}_j\}$)
    for $k = 1 : N_t$ do
        $\bar{\lambda}_k \leftarrow \bar{\lambda}_k + \bar{G}_{k,k}\bar{z}_k$
    end for
    return $\{\lambda_k\}$
end function
```

Algorithm 6.14 Harmonic Balance Preconditioner

```
function HBPC($\{\bar{z}_k\}$)                    ▷ Preconditioner used with GMRES
    for $k = 1 : N_t$ do                              ▷ $\bar{\lambda} = \tilde{\bar{M}}^{-1}\bar{z}$
        $\bar{\lambda}_k \leftarrow \left[N_t T^{-1} C + \bar{G}_{k,k}\right]^{-1} \bar{z}_k$
        $\bar{z}_k \leftarrow \bar{z}_k - \bar{G}_{k,k}\bar{\lambda}_k$
    end for
    $\{\tilde{\lambda}_j\} \leftarrow$FFT($\{\bar{\lambda}_k\}$)
    $\{\tilde{z}_j\} \leftarrow$FFT($\{\bar{z}_k\}$)
    for $j = 1 : N_t$ do
        $\tilde{z}_j \leftarrow \tilde{z}_j - i\omega_j C \tilde{\lambda}_j$
        $\tilde{\lambda}_j \leftarrow \tilde{\lambda}_j + \left[i\omega_j C + \tilde{G}_0\right]^{-1} \tilde{z}_j$
    end for
    $\{\bar{\lambda}_k\} \leftarrow$IFFT($\{\tilde{\lambda}_j\}$)
    return $\{\bar{\lambda}_k\}$
end function
```

6.2.2 Preconditioners

The main difficulty in the harmonic balance method is solving the large linear system of equations resulting from the application of Newton's method. For all but the smallest number of harmonics, applying a factorization algorithm to $\bar{J}$ or $\tilde{J}$ is intractable. Therefore, iterative methods are typically employed to solve (6.65). Since the Jacobian is nonsymmetric, a nonsymmetric Krylov subspace method GMRES can be used for this purpose.

Frequency-Domain Block-Diagonal Preconditioner

A robust preconditioning strategy is necessary to avoid potentially slow convergence of GMRES. A popular type of preconditioner can be constructed by extracting the block diagonals of $\tilde{J}$ from the frequency domain representation of Newton's method. We will denote this preconditioner by $\tilde{M}_f$. The matrix $\tilde{C}$ is already a block–diagonal matrix, so it is completely included in $\tilde{M}_f$. The matrix $\tilde{\Gamma}\bar{G}\tilde{\Gamma}^H$ has a constant block–diagonal $\tilde{G}_0$, with values equation to the average of the block–diagonal entries of $\bar{G}$:

$$\tilde{G}_0 = \frac{1}{N_t}\sum_{k=1}^{N_t} \bar{G}_{k,k}. \tag{6.67}$$

Combining these two expressions, the nonzero entries of $\tilde{M}_f$ are given by

$$\{\tilde{M}_f\}_{j,j} = \tilde{C}_{j,j} + \tilde{G}_0. \tag{6.68}$$

The structure of this preconditioner is useful because problems of the form $\tilde{M}_f\tilde{\delta} = \tilde{r}$ can be solved for $\tilde{\delta}$ by solving N_t smaller independent problems, each of which are the same size as the underlying DAE. For problems without motion where the magnetic fields are not driven too far into saturation, the actual time variation of the $\bar{G}_{k,k}$ will be small. In this case, the average $\tilde{G}_0$ does a good job representing the fundamental behavior of the linearization. As the device is driven further into saturation, the time–variation of the $\bar{G}_{k,k}$ becomes stronger as the incremental reluctivity begins to vary from its low–field value toward that of free–space in the strong field–limit. In this case, $\tilde{G}_0$ fails to capture the harmonic content generated by the saturation nonlinearity.

For problems with motion, the problem is even more extreme. Each $\bar{G}_{k,k}$ can be permuted into the following form,

$$G_{k,k} = \begin{bmatrix} G_{k,k}^{r,r} & G_{k,k}^{r,s} \\ G_{k,k}^{s,r} & G_{k,k}^{s,s} \end{bmatrix}. \tag{6.69}$$

where $G_{k,k}^{r,r}$ and $G_{k,k}^{s,s}$ represent matrices associated with the rotor and stator, respectively, and $G_{k,k}^{r,s}$ and $G_{k,k}^{s,r}$ represent matrices coupling the rotor and stator through the airgap. Because the rotor is assumed to be rotating at a constant velocity, these matrices are time–varying with a mean–value of zero. In other words, $\tilde{G}_0$, the average of the block-diagonals of $\bar{G}$, can be written as

$$\tilde{G}_0 = \begin{bmatrix} \tilde{G}_0^{r,r} & 0 \\ 0 & \tilde{G}_0^{s,s} \end{bmatrix}. \tag{6.70}$$

This means that, in employing the frequency domain block–diagonal preconditioner, the airgap between the stator and rotor is implicitly modeled as a perfect magnetic insulator.

Time-Domain Block-Diagonal Preconditioner

A similar approach to constructing a block–diagonal preconditioner can be taken starting from the time–domain representation of Newton's method by extracting the block-diagonals of $\bar{J}$. Since $\bar{G}$ is block–diagonal, it would be included entirely within this matrix. The block–diagonals of $\tilde{\Gamma}^H \tilde{C} \tilde{\Gamma}$ are equal to the average of the block-diagonals of $\tilde{C}$. In fact, these blocks are zero because

$$\sum_{j=1}^{N_t} \omega_j = 0. \tag{6.71}$$

The result is that extracting the block-diagonals of $\bar{J}$ leaves the matrix $\bar{G}$. Therefore, a preconditioner constructed this way exactly represents the part of the Jacobian due to the magnetic nonlinearity and motion. However, it fails to capture any aspects of the time-derivatives in the problem because all of that information is contained in the off–diagonal blocks of $\tilde{\Gamma}^H \tilde{C} \tilde{\Gamma}$. We could expect this preconditioner to work well at low frequencies. This is a moot point since it would not be necessary to model the dynamics of such a low frequency problem in the first place.

It would be beneficial to include some information about the conductivity in a time–domain preconditioner. This can be accomplished by adding a positive scalar multiple of C to each block on the diagonal of $\bar{G}$. This approach can be motivated by considering what a block–diagonal preconditioner would look like if (4.72) was discretized with the implicit–Euler method instead of the harmonic expansion of (6.40).

The matrix $\tilde{\Gamma}^H \tilde{C} \tilde{\Gamma}$ is the result of calculating the time–derivative in the frequency–domain; it resembles what would occur if the time–derivatives were approximated using a central finite–difference scheme. Central difference approximations necessarily result in zeros on the diagonal of the derivative matrix. If we replace the derivative approximation in the Jacobian $\bar{J}$ with one based on the implicit–Euler method, (6.62) can be approximated as

$$\hat{\bar{J}} \hat{\bar{\delta}} = \bar{r} \tag{6.72}$$

where the approximate Jacobian $\hat{\bar{J}}$ is

$$\hat{\bar{J}} = \bar{C} + \bar{G}, \tag{6.73}$$

and the nonzero entries of $\bar{C}$ are given by

$$\begin{aligned} \bar{C}_{i,i} &= \frac{N_t}{T} C \\ \bar{C}_{i,i-1} &= -\frac{N_t}{T} C. \end{aligned} \tag{6.74}$$

The matrix $\hat{\bar{J}}$ has a similar structure to the multiple shooting method Jacobian using equally spaced sample times and the implicit–Euler method as the numerical integration scheme.

Even though they have different structures, $\hat{\bar{J}}$ approximates $\bar{J}$ in the sense that the solutions to (6.62) and (6.72) should be similar, i.e. $\hat{\bar{\delta}} \approx \bar{\delta}$, for an $\bar{r}$ that is fixed between problems. A time domain block–diagonal preconditioner $\bar{M}_t$ for (6.62) can be constructed by extracting the block–diagonals of $\hat{\bar{J}}$, which are given by

$$\{\bar{M}_t\}_{i,i} = \frac{N_t}{T} C + \bar{G}_{i,i}. \tag{6.75}$$

The key difference between $\bar{M}_t$ and a preconditioner constructed by simply extracting the block-diagonals of $\bar{J}$ is that the frequency–domain derivative calculation is replaced with the backward–difference approximation in order to collect information about the conductivity on the block–diagonal.

An Overlapping Time/Frequency Domain Preconditioner

We have seen in the previous section that the block–diagonals of the Jacobian capture different aspects of the problem depending on whether it is stated in the time domain or the frequency domain. In the frequency domain, the block–diagonals of the Jacobian capture the effects of magnetic diffusion and average effect of the nonlinearity, but discards all the information about rotational motion. In the time domain, the block–diagonals of the approximated Jacobian capture the effects of rotational motion and time variation due to the nonlinearity, but does not contain any information about how the fields are coupled in time through the magnetic diffusion process. Clearly it would be useful to combine these two ideas. In this section, we present a method for doing so based on a two step relaxation procedure.

First let us examine the preconditioned Richardson iteration for solving an equation of the form $My = b$ with preconditioner $\hat{M}$ [119]. Starting from an approximation $\hat{y}$ of y, the preconditioned Richardson iteration updates $\hat{y}$ using the following formula:

$$\hat{y} \leftarrow \hat{y} - \hat{M}^{-1}\left[M\hat{y} - b\right] = \hat{M}^{-1}b + \left[I - \hat{M}^{-1}M\right]\hat{y}. \tag{6.76}$$

This iterative method can be generalized to a sequence of m preconditioners $\hat{M}_k$ that update $\hat{y}$ in a series of steps:

$$\begin{aligned} \hat{y} \leftarrow \hat{y} - &\hat{M}_1^{-1}\left[M\hat{y} - b\right] \\ &\vdots \\ \hat{y} \leftarrow \hat{y} - &\hat{M}_m^{-1}\left[M\hat{y} - b\right]. \end{aligned} \tag{6.77}$$

If we specialize this method for $m = 2$, it is possible to combine these two steps into a single equation,

$$\hat{y} \leftarrow \hat{M}_{1,2}^{-1}b + \left[I - \hat{M}_{1,2}^{-1}M\right]\hat{y}, \tag{6.78}$$

where

$$\hat{M}_{1,2}^{-1} = \hat{M}_1^{-1} + \hat{M}_2^{-1} - \hat{M}_2^{-1}M\hat{M}_1^{-1}. \tag{6.79}$$

Comparing (6.76) and (6.78) reveals that the sequence of two preconditioners implicitly defines a single preconditioner $\hat{M}_{1,2}$ whose inverse is given by (6.79).

A similar approach can be used to derive a mixed time/frequency domain preconditioner using the previously described block–diagonal Jacobian approximations. Starting with the time domain representation (6.62), apply the iteration with preconditioner M_t:

$$\bar{\delta}_1 = \bar{\delta} - \bar{M}_t^{-1}\left[\bar{J}\bar{\delta} - \bar{r}\right]. \tag{6.80}$$

The approximation $\bar{\delta}_1$ is then transformed into the frequency domain,

$$\tilde{\delta}_1 = \Gamma\bar{\delta}_1. \tag{6.81}$$

Next, the relaxation iteration can be applied using the frequency domain representation (6.65) and preconditioner $\tilde{M}_f$:

$$\tilde{\delta}_2 = \tilde{\delta}_1 - \tilde{M}_f^{-1}\left[\tilde{J}\tilde{\delta}_1 - \tilde{r}\right] \tag{6.82}$$

Finally, $\bar{\delta}$ is calculated by transforming $\tilde{\delta}_2$ back into the time domain,

$$\bar{\delta} \leftarrow \tilde{\Gamma}^H\tilde{\delta}_2. \tag{6.83}$$

Combining the preceding steps reveals the following expression for the overlapping time/frequency domain preconditioner $\tilde{\bar{M}}$:

$$\tilde{\bar{M}}^{-1} = \bar{M}_t^{-1} + \tilde{\Gamma}^H\tilde{M}_f^{-1}\tilde{\Gamma} - \tilde{\Gamma}^H\tilde{M}_f^{-1}\tilde{\Gamma}\bar{J}\bar{M}_t^{-1} \tag{6.84}$$

This derivation of $\tilde{\bar{M}}$ is based on the combination of two block–diagonal preconditioners through a preconditioned Richardson iteration. However, the iterative method, even with the preconditioner, may fail to converge quickly. In fact, it could diverge if the iteration matrix $I - \tilde{\bar{M}}^{-1}\bar{J}$ has eigenvalues with modulus greater than one. The preconditioner can be used with GMRES to guarantee convergence regardless of this fact.

6.2.3 Error Estimation and Refinement

The harmonic balance method is more constrained in the method of adaptive refinement than time–domain methods based on Runge–Kutta numerical integration. The DFT matrix $\tilde{\Gamma}$, as we have defined it, is orthonormal due to the equally spaced discretization of both the time and frequency axes. In fact, it is possible to define other "discrete Fourier transformations" using arbitrary sample times and frequencies. The problem with doing so is that the corresponding transformation matrix drifts further from orthonormality as the sampling becomes increasingly nonuniform. The deviation from orthonormality severely degrades the quality of the solution approximation and it becomes necessary to reinterpret the basis functions in this context [120]. Therefore, when performing adaptive solution refinement within the context of the HB method, we are limited to either increasing the number of sample times and number of harmonics by an integer multiple, or accepting the current solution as our final approximation.

The simplest method for estimating the error in the harmonic balance solution is to create an error signal by setting to zero the lower half of the solution bandwidth. In the time–domain, this results in the error vectors $\bar{e}_k$, where

$$\bar{e}_k = \frac{1}{\sqrt{N_t}} \sum_{j \in \mathcal{J}} \tilde{x}_j e^{i\omega_j t_k} + \frac{1}{\sqrt{N_t}} \tilde{x}_{N_h+1} \cos\left(N_h \omega_e t_k\right), \tag{6.85}$$

and $\mathcal{J} = \{j : |\omega_j| > \frac{N_h}{2}\omega_e\}$. The error indicators for the solution are taken as the seminorm of the error vectors, normalized to the actual solution:

$$\varepsilon_k = \frac{\|\bar{e}_k\|_W}{\max\limits_k \|x_k\|_W}. \tag{6.86}$$

This quantity is actually an estimation of the error in a solution approximation with about half the bandwidth. A similar issue is encountered when estimating the local error in the time–domain algorithms using a lower order quadrature weight vector.

Algorithm 6.15 Adaptive Harmonic Balance Method

```
function HBAdaptive($\{\bar{x}_k\}$, $\{t_k\}$, $\varepsilon_{max}$, $\epsilon_{exact}$, $\epsilon_{smooth}$, $N_{smooth}$)    ▷ Solve (6.57)
    $\{\bar{x}_k\}$ ←HBFixed($\{\bar{x}_k\}$, $\{t_k\}$, $\epsilon_{exact}$, $\infty$)
    $\{\varepsilon_k\}$ ←HBError($\{\bar{x}_k\}$, $\{t_k\}$)
    while $\varepsilon_{max} > \max_k \varepsilon_k$ do
        $\{\bar{x}_k\}$, $\{t_k\}$ ←HBRefine($\{\bar{x}_k\}$, $\{t_k\}$)
        $\{\bar{x}_k\}$ ←HBFixed($\{\bar{x}_k\}$, $\{t_k\}$, $\epsilon_{smooth}$, $N_{smooth}$)
        $\{\varepsilon_k\}$ ←HBError($\{\bar{x}_k\}$, $\{t_k\}$)
    end while
    return $\{\bar{x}_k\}$, $\{t_k\}$
end function
```

Algorithm 6.15 is a pseudo–code implementation of the adaptive harmonic balance method proposed here. The structure of the code is largely the same as the structure for the adaptive time–domain framework in Algorithm 6.10. Algorithm 6.16 implements the previously discussed error estimation scheme. Algorithm 6.17 implements a simple equidistant grid division scheme and interpolates the previous solution onto the next grid.

Algorithm 6.16 Adaptive Harmonic Error Estimate

```
function HBError($\{\bar{x}_k\}$)
    $\{\tilde{E}_j\} \leftarrow$FFT($\{\bar{x}_k\}$)
    $\tilde{E}_1 \leftarrow 0$
    for $j = 2 : \lceil \frac{N_h}{2} \rceil$ do
        $\tilde{E}_j \leftarrow 0$
        $\tilde{E}_{N_t-j+1} \leftarrow 0$
    end for
    $\{\bar{e}_k\} \leftarrow$IFFT($\{\tilde{E}_j\}$)
    $\epsilon \leftarrow 0$
    for $k = 1 : N_t$ do
        $\varepsilon_k \leftarrow \|\bar{e}_k\|_W$
        $\epsilon \leftarrow \max(\epsilon, \|\bar{x}_k\|_W)$
    end for
    for $k = 1 : N_t$ do
        $\varepsilon_k \leftarrow \epsilon^{-1}\varepsilon_k$
    end for
    return $\{\varepsilon_k\}$
end function
```

Algorithm 6.17 Harmonic Balance Grid Refinement

```
function HBRefine($\{\bar{x}_k\}$, $\{t_k\}$)
    $\{\tilde{x}_k\} \leftarrow$FFT($\{\bar{x}_k\}$)
    $l \leftarrow 1$
    for $k = 1 : N_h$ do
        $\tilde{y}_k \leftarrow \tilde{x}_l$
        $l \leftarrow l + 1$
    end for

    $\tilde{y}_{N_h+1} \leftarrow \frac{1}{2}\tilde{x}_l$
    for $k = (N_h + 2) : (2N_t - N_h + 1)$ do
        $\tilde{y}_k \leftarrow 0$
    end for
    $\tilde{y}_{2N_t - N_h} \leftarrow \frac{1}{2}\tilde{x}_{N_h+1}$

    for $k = (2N_t - N_h + 2) : (2N_t)$ do
        $\tilde{y}_k \leftarrow \tilde{x}_l$
        $l \leftarrow l + 1$
    end for
    $\{\bar{x}_k\} \leftarrow$IFFT($\{\tilde{y}_k\}$)

    $l \leftarrow 1$
    for $k = 1 : (N_t - 1)$ do
        $s_l \leftarrow t_k$
        $l \leftarrow l + 1$
        $s_l \leftarrow \frac{1}{2}(t_k + t_{k+1})$
        $l \leftarrow l + 1$
    end for
    $s_{2N_t - 1} \leftarrow t_{N_t}$
    $s_{2N_t} \leftarrow t_{N_t} + \frac{T}{2N_t}$
    $\{t_k\} \leftarrow \{s_k\}$

    return $\{\bar{x}_k\}$, $\{t_k\}$
end function
```

Chapter 7

Simulations

This Chapter presents a comparative study of the previously developed algorithms. Section 7.1 describes the surface–mount permanent magnet machine model that will be the focus of the simulations. Sections 7.2 presents a simulation of this machine when the windings are excited by a sinusoidal voltage waveforms. The efficiency of the algorithms applied to this situation is examined in Section 7.3. In particular, we look at the simulation time as a function of the discretization error threshold. Finally, we fix the discretization threshold and examine several factors that may affect the simulation times in Section 7.4. All of the simulations in this chapter are performed using a MATLAB toolbox developed by the author [121].

7.1 Surface–Mount Permanent Magnet Machine Model

A single pole of the machine model that will be used in the subsequent simulations is depicted in Fig. 7.1. The length of the machine is assumed to be 95.26mm in the z–direction. The blue regions are nonlinear ferromagnetic materials. The nonlinear B–H and corresponding

M–B curves for this material are shown in Figs. 7.2 and 7.3, respectively. The M–B curve is modeled using the function

$$M\left(B\right) = C_0 + \frac{\chi_0}{2}\left[B - \left(B - B_s\right)\mathrm{erf}\left(a\left[B - B_s\right]\right) - \frac{1}{a\sqrt{\pi}}e^{-a^2\left(B-B_s\right)^2}\right], \tag{7.1}$$

where

$$C_0 = \frac{\chi_0}{2}\left[B_s\mathrm{erf}\left(aB_s\right) + \frac{1}{a\sqrt{\pi}}e^{-a^2B_s^2}\right] \tag{7.2}$$

and the parameters B_s, χ_0, and a are given in Table 7.1. The corresponding B–H curve is given by

$$H\left(B\right) = \frac{1}{\mu_0}B - M\left(B\right). \tag{7.3}$$

Roughly speaking, B_s represents the saturation flux–density, χ_0 represents the initial slope of the M–B curve, and a controls the smoothness of the transition between the "linear" region and "saturation" region. The value of χ_0 in Table 7.1 corresponds to a linear relative permeability of approximately 434. It should be noted that, while the value of B_s does correspond well to the saturation region of the M–B curve, the slope of the B–H curve flattens out at much lower flux densities. The relationship between B_s and the region of diminishing returns in the B–H curve is a direct function of the transition parameter a.

The core loss density in the ferromagnetic material is estimated using the generalized Steinmetz equation

$$p_{core} = \sum_k \alpha\left[kf_e\right]^\beta\left[B_{k,x}^2 + B_{k,y}^2\right]^\gamma, \tag{7.4}$$

with the parameters α, β, and γ given in Table 7.2. The quantities $B_{k,x}$ and $B_{k,y}$ represent the k^{th} harmonics of the x– and y–components of the magnetic flux–density, respectively. In the discrete problem, the harmonic decomposition is performed in each finite–element and the total core losses are obtained by summing over all the elements.

Returning to Fig. 7.1, the magenta regions are permanent magnets modeled with a remnant flux–density of 1.23T. The yellow regions represent copper stator windings. In addition

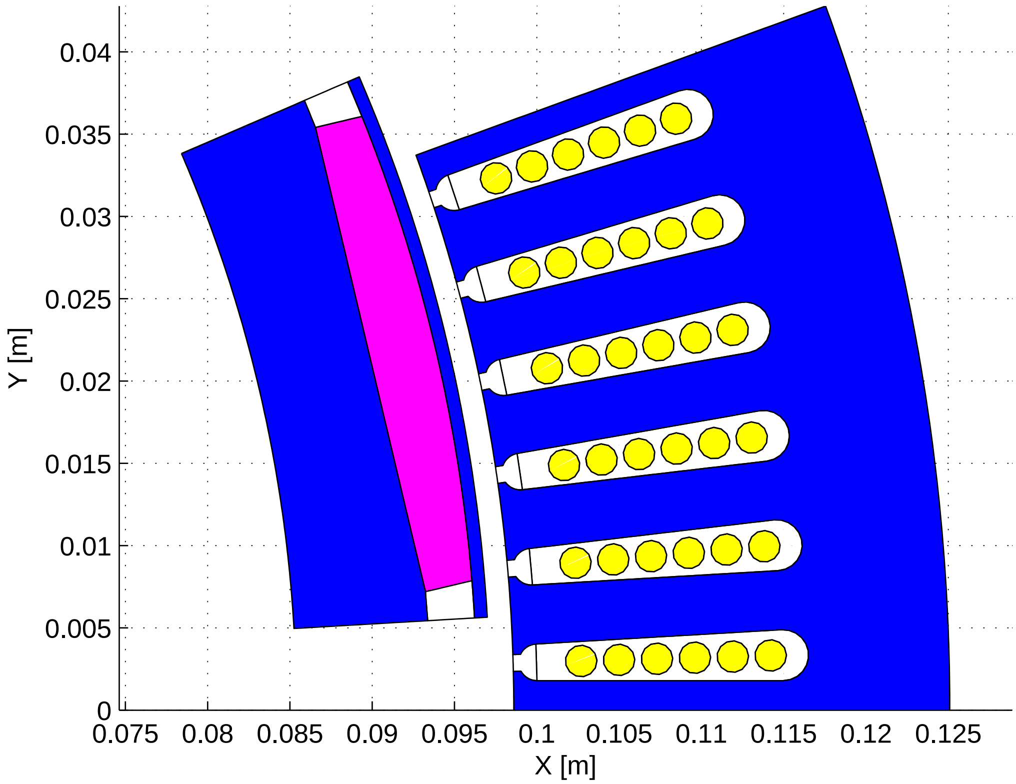

Figure 7.1: Single pole model of an 18–pole surface mount permanent magnet machine.

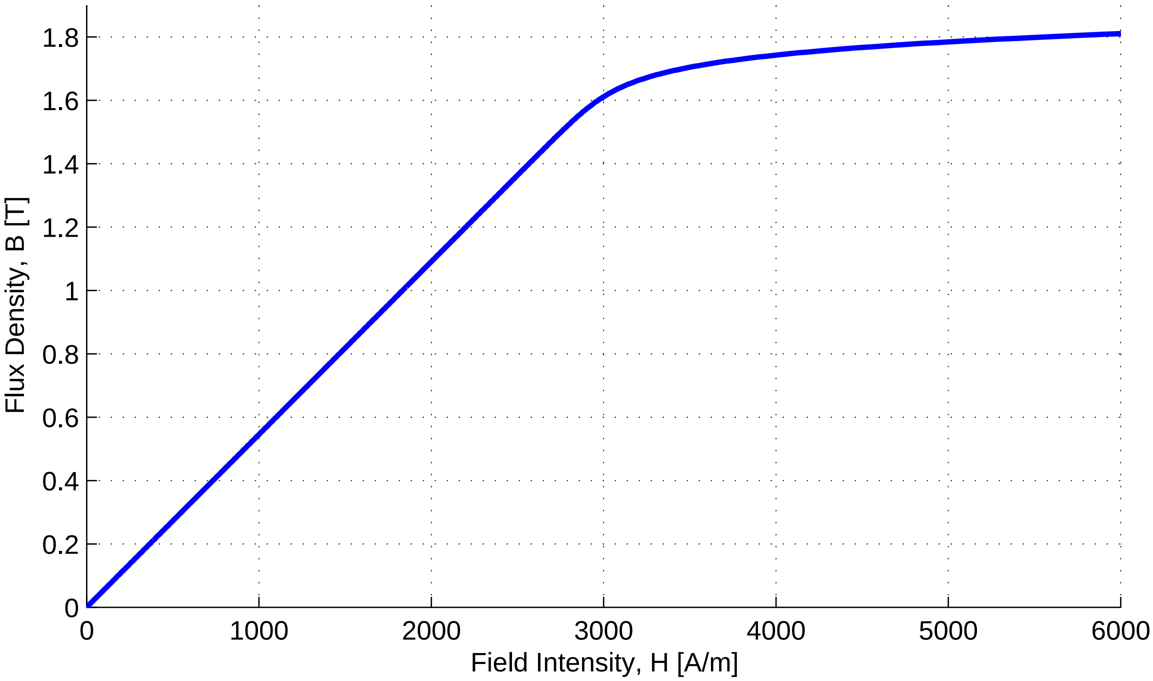

Figure 7.2: B–H curve of a nonlinear ferromagnetic material.

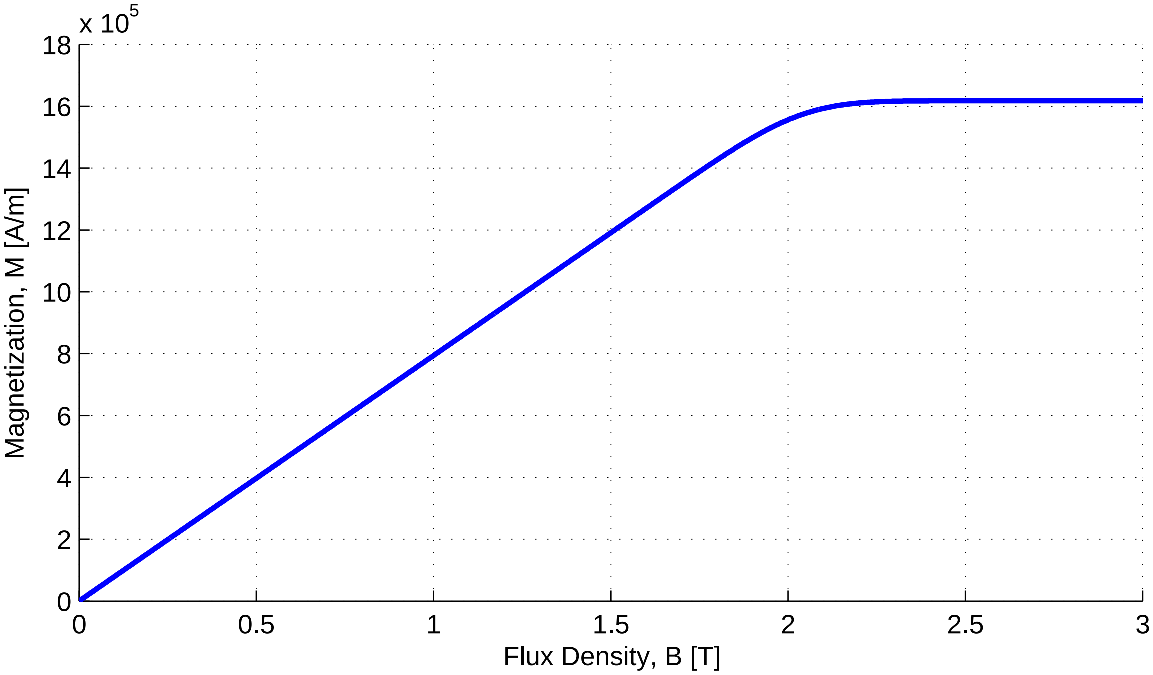

Figure 7.3: M–B curve of a nonlinear ferromagnetic material.

Table 7.1: M–B Curve Parameters

B_s	χ_0	a
$2.04\,[\mathrm{T}]$	$7.939 \times 10^5 \left[\frac{\mathrm{A}}{\mathrm{m{-}T}}\right]$	$4.95 \left[\frac{1}{T}\right]$

Table 7.2: Generalized Steinmetz Equation Parameters

α	β	γ
40.0	1.30	0.90

Table 7.3: Nonzero Conductivities

Copper	Permanent Magnet	Unlaminated (Solid) Steel
$5.24 \times 10^7\,[S]$	$2.12 \times 10^6\,[S]$	$6.67 \times 10^5\,[S]$

to the windings, the permanent magnets are also modeled with a nonzero conductivity. In the base case simulation, the ferromagnetic regions are assumed to represent steel laminations and will be modeled with zero conductivity. In one variation, we will assume that the rotor back–iron (the innermost blue region) will be modeled as a solid piece of ferromagnetic steel with nonzero conductivity. The relevant nonzero conductivities are given in Table 7.3. Finally, the white/transparent regions represent air, modeled using vacuum properties.

The three–phase stator windings are connected in an ungrounded wye–configuration. A standard three–phase winding layout is assumed with 6 turns per slot and 2 slots per phase. For each phase, 3 coils connect 6 different poles in series, and the coils themselves are connected in parallel. This results in 46 variables related to the stator winding in a voltage driven problem. Of these, 36 correspond to E_ϕ appearing in the field–current constraints, 3 correspond to the strand currents, 3 correspond to the phase–voltages, 3 correspond to the bundle currents,[1] and 1 corresponds to the common–mode voltage. The rotor has one external circuit variable, E_ϕ, corresponding to the voltage drop across the permanent magnet.

The triangular mesh used to discretized the continuous problem is displayed in Fig 7.4. First–order elements are used during the finite–element procedures. Table 7.4 gives the number of elements and nodes in the mesh. After the application of the anti–periodic boundary condition and the inclusion of the circuit variables, there are a total of 2459 variables in the spatially discretized problem.

[1] Because all of the strands are connected in series, in this, the strand currents equal the bundle currents. The modeling software, however, is setup to handle more general situations than this. As a result, redundant equations appear in degenerate cases.

Table 7.4: Mesh Size

	Elements	Nodes
Rotor	478	288
Stator	4164	2159
Total	4642	2447

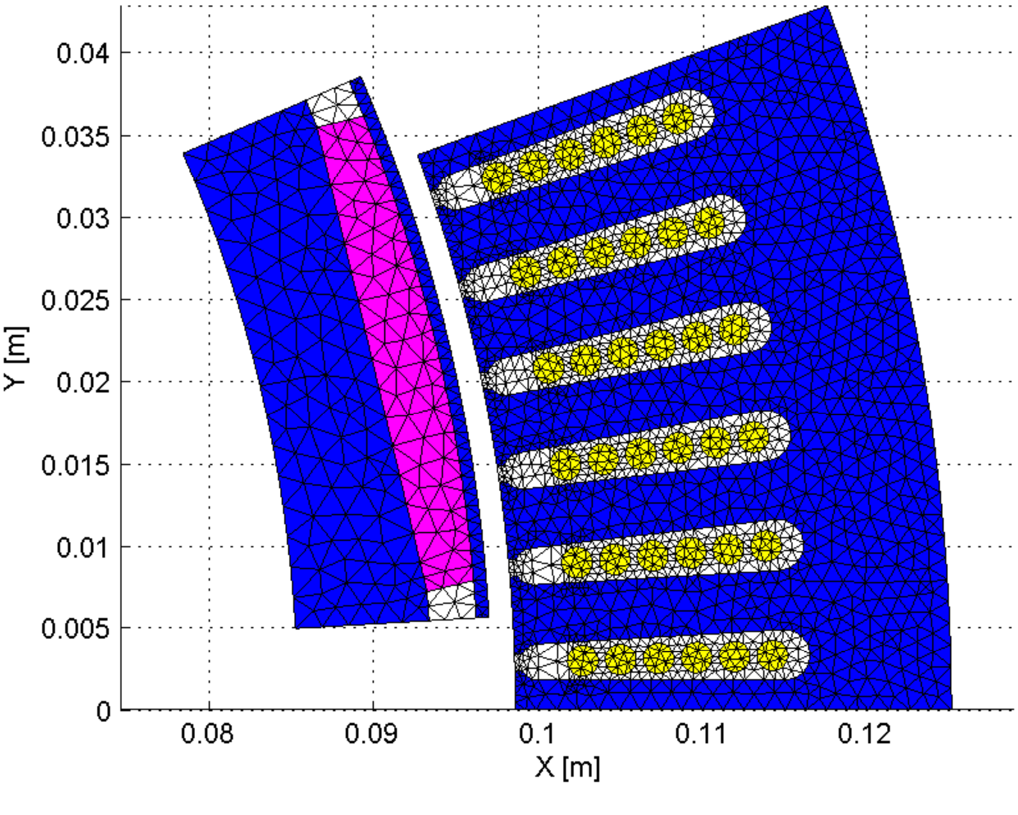

Figure 7.4: Mesh of the single pole model electric machine model in Fig. 7.1.

7.2 Sinusoidal Voltage–Driven Problem

As our base case simulation, the machine is excited by a purely sinusoidal three–phase voltage waveform. We assume the machine is operating at a mechanical angular velocity of 8000RPM, corresponding to an electrical frequency of 1200Hz. The amplitude of the applied line–to–line voltages is 340V. The phase of the waveforms are chosen so that they are 90 degrees out of phase with the voltage induced by the permanent magnets during open circuit operation. This results in a current waveform producing very close to the maximum torque obtainable with this voltage constraint.

7.2.1 Generating the Voltage Waveform

The open circuit flux linkage can be calculated by performing magnetostatic finite–element simulations using N_t evenly spaced time points. The flux linkage waveform is plotted in Fig. 7.5 using $N_t = 18$. The amplitudes of the harmonic content of this waveform are displayed in Fig. 7.6. In addition to the fundamental component, there is a significant 3^{rd} harmonic.

The open circuit phase voltages can be determined by differentiating the flux linkage waveform in the frequency domain. The resulting waveform is plotted in Fig. 7.7. The waveform displays a distinct trapezoidal shape. The harmonic content of this waveform is displayed in Fig. 7.8. The third harmonic of this waveform has an amplitude of $85.5V$.

Based on the phase voltages, the fundamental of the open circuit line–to–line voltages have angles of $\phi_{ab} = -120°$, $\phi_{bc} = 0°$, $\phi_{ca} = 120°$. This result was effected deliberately by choosing the initial mechanical angle of the rotor as $\phi_{rotor} = \frac{10}{3}^{\circ}$. The applied line–to–line voltages have their phases shifted 90° relative to the open circuit voltages.

7.2.2 Dynamic Simulation

We have calculated an assumed "exact" solution of the dynamic voltage driven problem using the adaptive harmonic balance method with a discretization error tolerance of 10^{-9}.

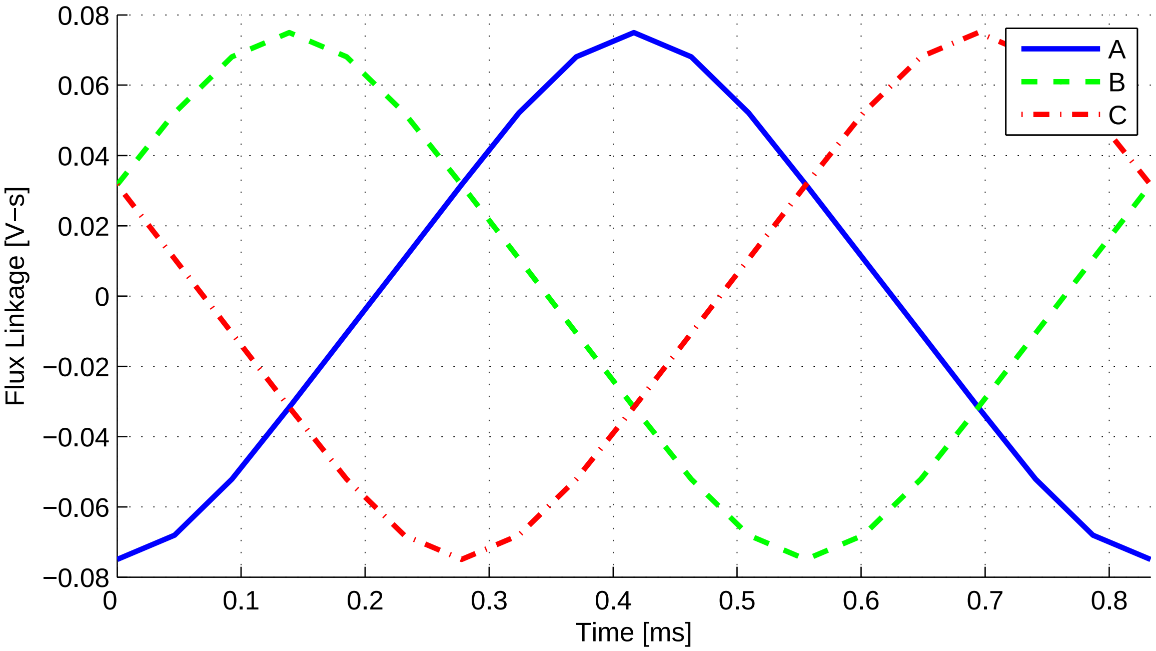

Figure 7.5: Open circuit flux linkage waveform calculated by performing magnetostatic simulations.

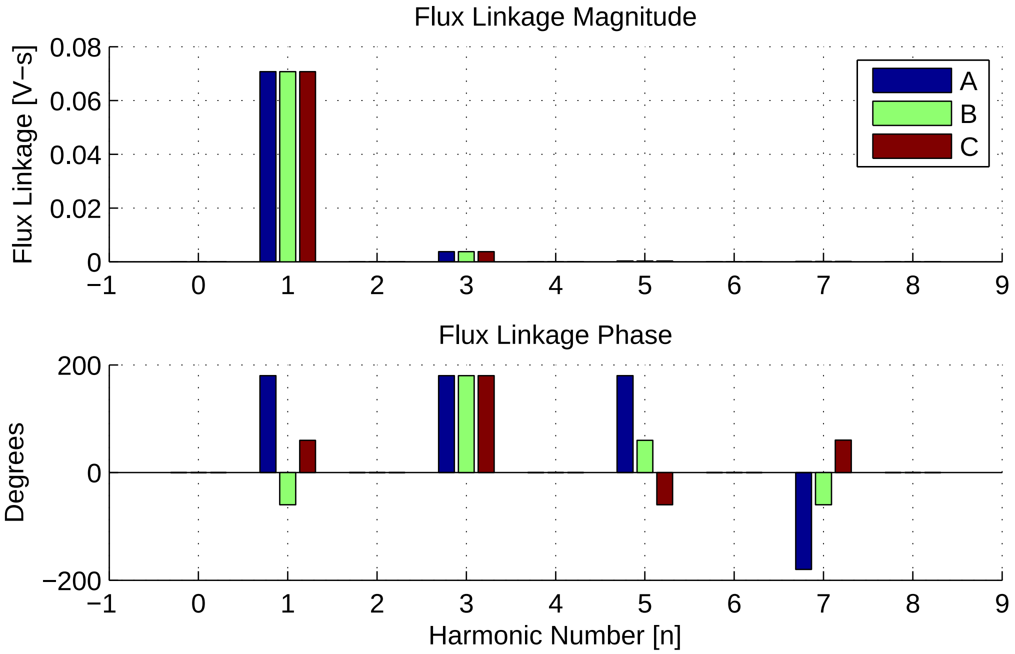

Figure 7.6: Open circuit flux linkage harmonics calculated by performing magnetostatic simulations.

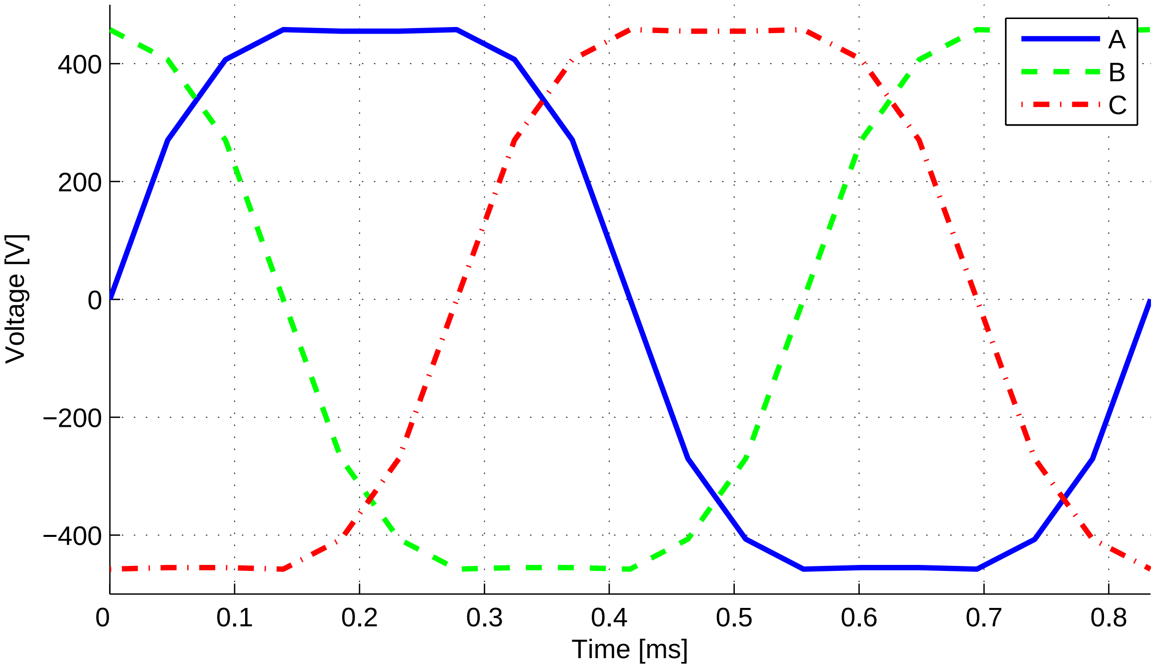

Figure 7.7: Open circuit flux linkage waveform calculated by performing magnetostatic simulations.

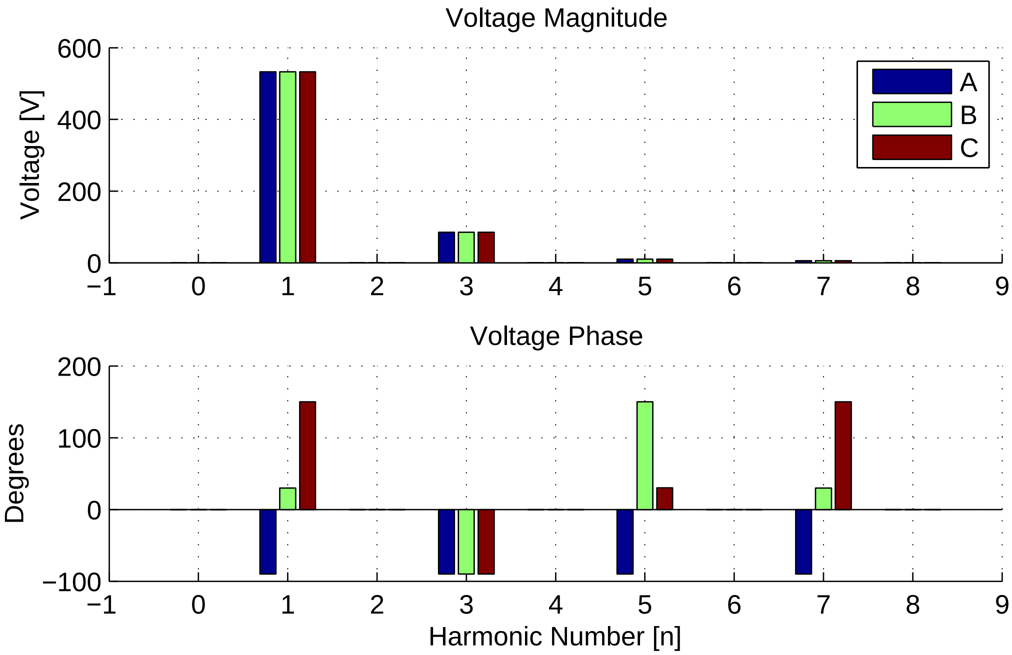

Figure 7.8: Open circuit flux linkage harmonics calculated by performing magnetostatic simulations.

Table 7.5: Stator and rotor losses for the sinusoidal voltage driven problem.

	Rotor	Stator
Conduction	213W	12.8kW
Core	15.7W	581W

The resulting line–to–neutral voltages and line currents from this simulation are shown in Figs. 7.9 and 7.10. A significant 3rd harmonic exists in the phase–voltage waveforms due to the permanent magnets as predicted in the previous section. The line currents are almost perfectly sinusoidal; the 1^{st} and 5^{th} harmonics have amplitudes of 554.3A and 3.1A, respectively.

The torque waveform determined from this simulation is shown in Fig. 7.11. The average torque over one period is 170N–m, giving an average output power of 142kW. The ripple torque has a peak–to–peak amplitude of about 34.4N–m. This is mostly due to the 6th and 12th harmonics of amplitudes 5.5N–m and 13.4N-m, respectively.

The loss density for this problem, averaged over one period, is shown in Fig. 7.12. The total losses calculated from this simulation were 13.6kW, giving an efficiency of 91.3%.[2] Table 7.5 separates the stator and rotor losses, and the conduction and core losses. The majority of the losses occur due to conduction in the stator. By way of comparison, a simple calculation of the i^2R losses based on the DC winding resistance would predict stator conduction losses of 6.94kW, 46% less than the 12.8kW calculated through the steady–state finite–element simulation. A close–up of the stator winding loss–density is shown in Fig. 7.13 demonstrating how the losses increase due to current redistribution in the windings along the length of the slot.

7.2.3 Estimating the Phase Voltage Waveform

With a fixed line–to–line voltage waveform, magnetostatic simulations can also be used to estimate the resulting line–current and phase voltage waveforms in the dynamic simulation

[2] Excluding end–turn effects

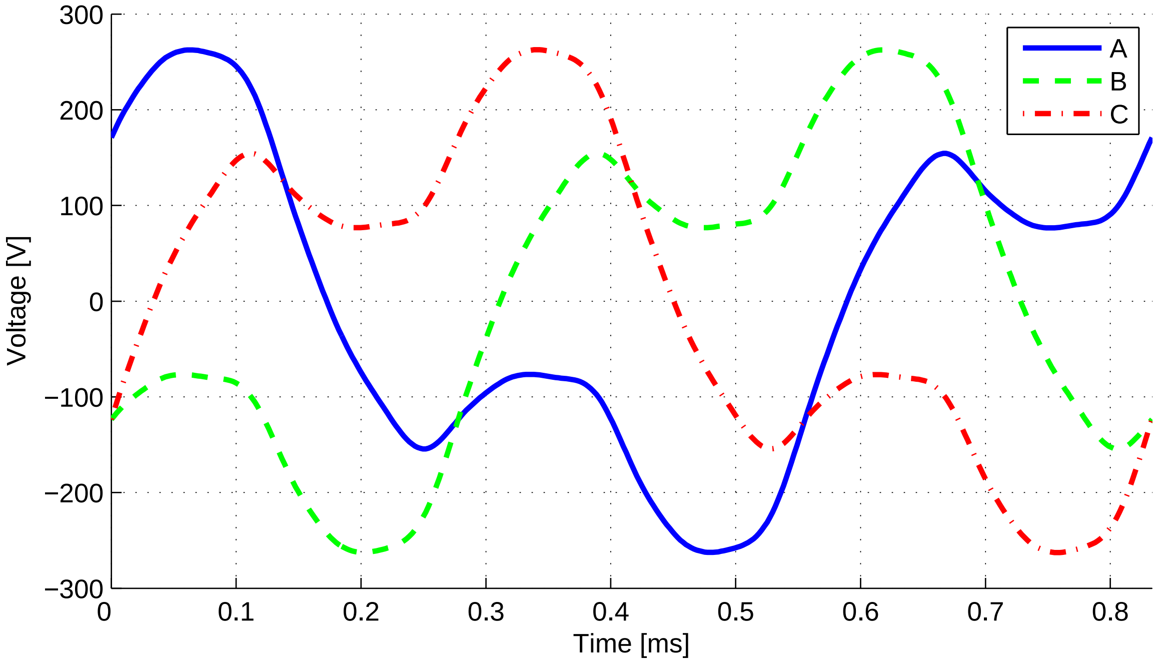

Figure 7.9: Line–to–neutral voltages from the sinusoidal voltage driven problem.

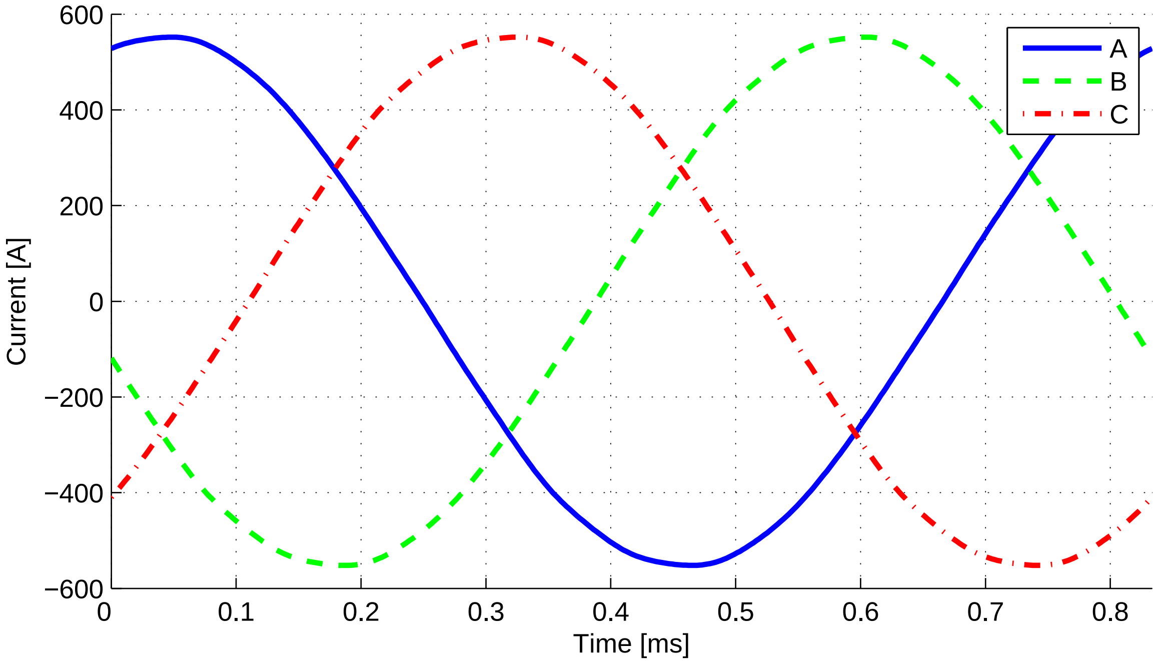

Figure 7.10: Line currents from the sinusoidal voltage driven problem.

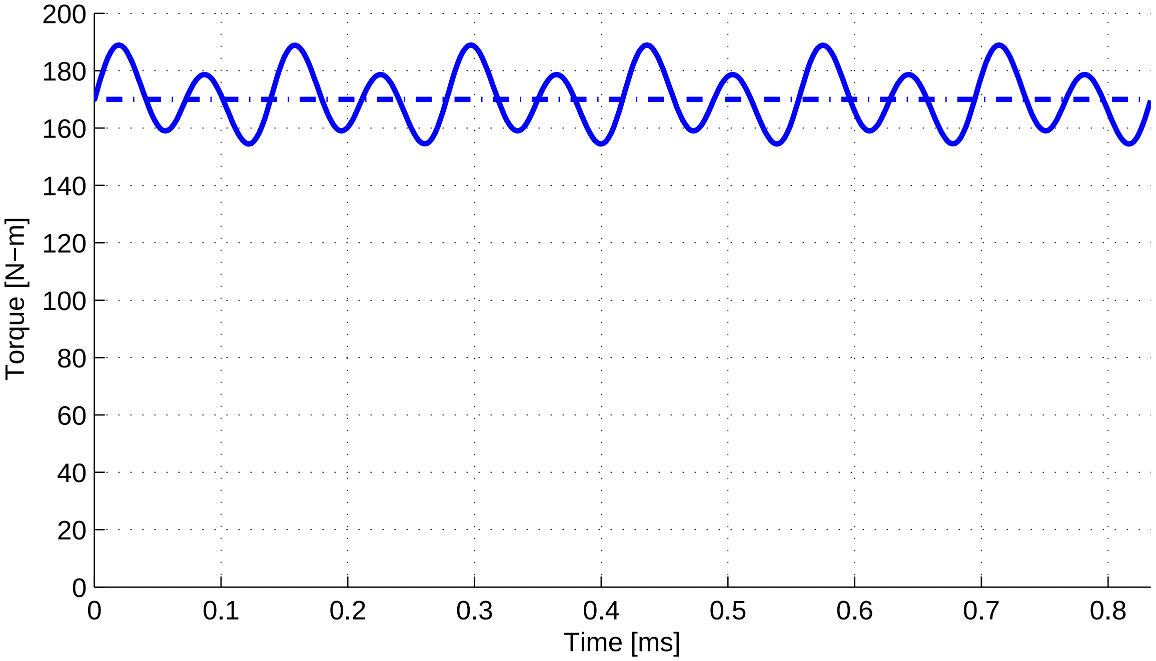

Figure 7.11: Torque waveform from the sinusoidal voltage driven problem.

if there is an equivalent external circuit model of the machine available. If we focus on calculating the fundamental harmonic of the current waveform, we can model the relationship between the voltage and current as

$$v = Ri + \frac{\partial \lambda\left(i\right)}{\partial t} \tag{7.5}$$

where v is a vector of applied line–to–neutral voltages, R is a diagonal matrix of resistances, and λ is a vector of nonlinear flux linkages. We assume that v and i can be written as

$$v = \tilde{v}_{-1} e^{-j\omega_e t} + \tilde{v}_1 e^{j\omega_e t}, \tag{7.6}$$

and

$$i = \tilde{i}_{-1} e^{-j\omega_e t} + \tilde{i}_1 e^{j\omega_e t}, \tag{7.7}$$

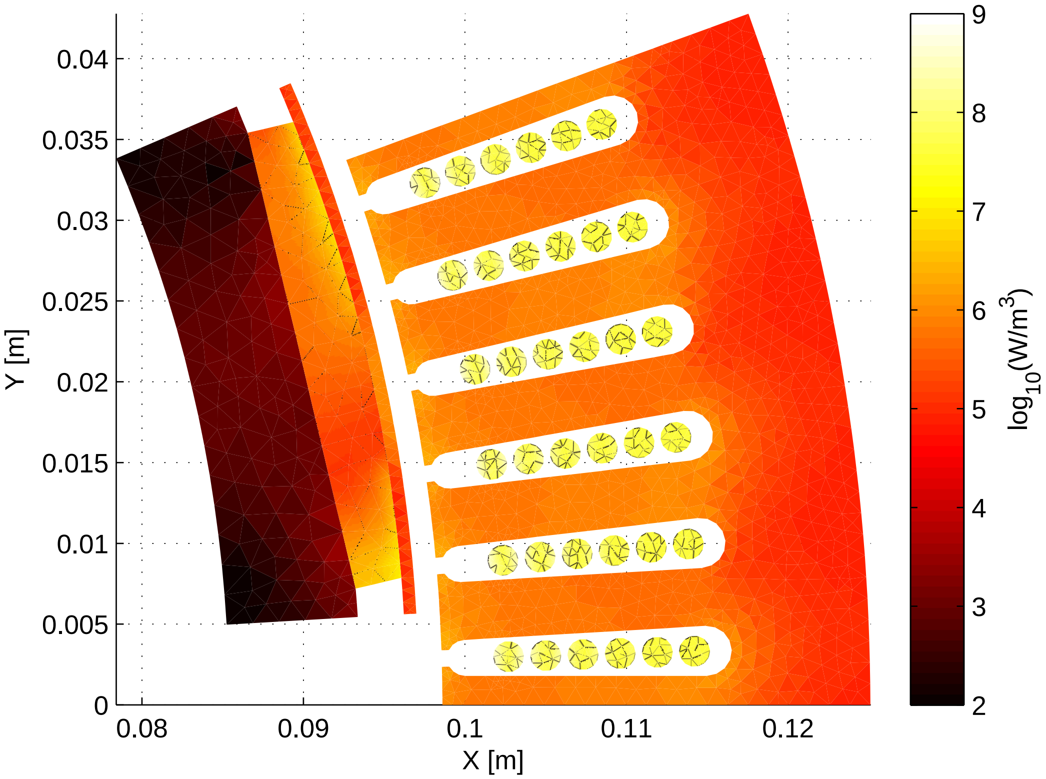

Figure 7.12: Average loss density $\mathbf{E} \cdot \mathbf{J}$ over one period from the sinusoidal voltage driven problem.

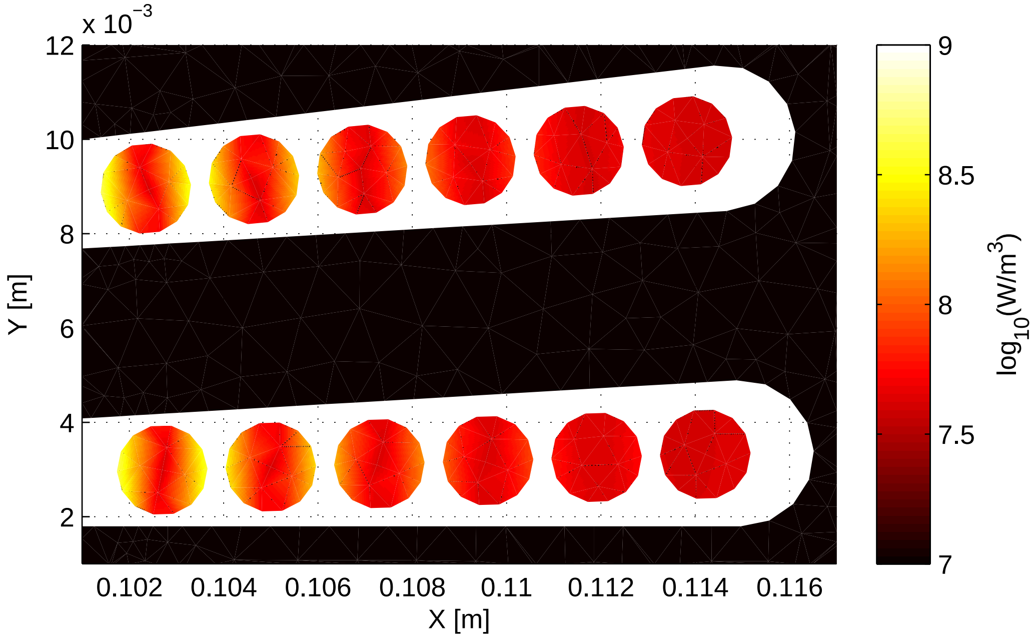

Figure 7.13: Close up of the average stator winding loss density from the sinusoidal voltage driven problem..

respectively. We search for i_n, $n = \pm 1$ that solves (7.5) in a weak sense[3] by requiring the residuals r_m,

$$\begin{aligned} r_m &= \frac{1}{T}\int_0^T e^{-jm\omega_e t}\left(Ri + \frac{\partial \lambda\left(i\right)}{\partial t} - V\right)dt, \\ &= Ri_m + v_{\lambda,m} - v_m \end{aligned} \tag{7.8}$$

vanish, where $m = \pm 1$ and

$$v_{\lambda,m} = \frac{1}{T}\int_0^T e^{-jm\omega_e t}\frac{\partial \lambda\left(i\right)}{\partial t}dt. \tag{7.9}$$

Newton's method can be applied to solve this equation with Jacobian given by

$$\frac{\partial r_m}{\partial i_n} = R + \frac{1}{T}\int_0^T e^{j(n-m)\omega_e t}\left[\frac{\partial L\left(i\right)}{\partial t} + jn\omega_e L\left(i\right)\right]dt \tag{7.10}$$

[3] This is essentially a single frequency Galerkin type harmonic balance problem.

Table 7.6: Incremental self and mutual inductances extracted from the magnetostatic simulation.

	Incremental Self Inductance	Incremental Mutual Inductance
0^{th} harmonic	117μH	21.4μH
2^{nd} harmonic	12.1μH	1.17μH

where

$$L\left(i\right) = \frac{\partial \lambda\left(i\right)}{\partial i} \tag{7.11}$$

is the nonlinear incremental inductance matrix. These inductances can also be extracted from the magnetostatic simulations. Since we have restricted $m, n = \pm 1$, it is sufficient to assume that L takes the form

$$L\left(i\right) = L_0 + L_{-2}e^{-j2\omega_e t} + L_2 e^{j2\omega_e t}, \tag{7.12}$$

in which case the Jacobian can be written as

$$\left\{\frac{\partial r_m}{\partial i_n}\right\} = \begin{bmatrix} R - j\omega_e L_0 & -j2\omega_e L_{-2} \\ j2\omega_e L_2 & R + j\omega_e L_0 \end{bmatrix}. \tag{7.13}$$

We have run this iterative procedure four times, once under open circuit operation conditions and twice more to correct the initial current predication. The line currents are estimated to have amplitudes of $553.6A$. This is very close to fundamental current harmonic amplitude calculated from the dynamic simulation of the previous section. The amplitudes of the self and mutual inductances calculated from the last magnetostatic simulation are given in Table 7.6. The phase voltage waveforms estimated from the magnetostatic simulations are shown in Fig. 7.14. All of these results can be combined to produce a good initial condition for dynamic simulations.

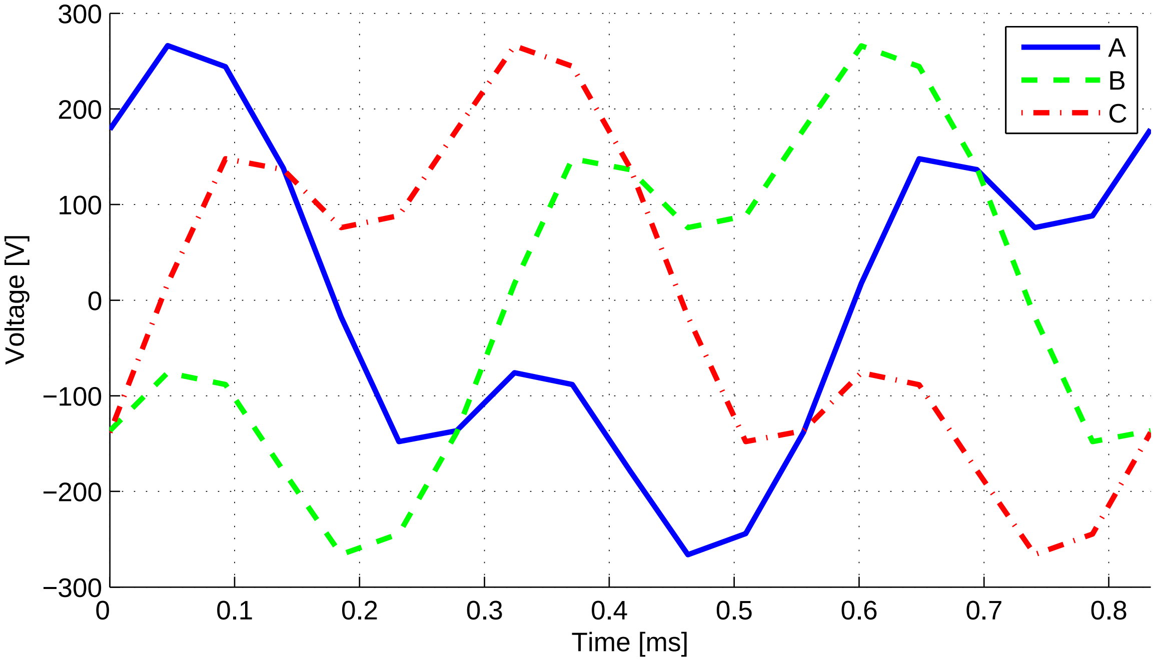

Figure 7.14: Line–to–neutral voltages predicted from the magnetostatic simulations.

7.3 Algorithm Efficiency

We now attempt to answer the question “which algorithm is fastest?” In order to do this, we have run both the dynamic voltage driven problem with varying accuracy thresholds and recorded the resulting simulation times. First, we ran the adaptive algorithms until a desired threshold for the estimated discretization error was achieved. The number of time steps/sample times from each adaptive simulation was used to choose a corresponding step size for the non–adaptive algorithms. In the following, the harmonic balance method is denoted HBA and HBF for the adaptive and fixed (non–adaptive) algorithms, respectively. The adaptive and fixed step size transient, single shooting, and multiple shooting methods are similarly denoted TRAx, TRFx, SSAx, SSFx, MSAx, and MSFx, respectively. The numeral x indicates the numerical integration technique used, with $x = 1$ for the implicit–Euler method and $x = 2, 3$ for the ESDIRK2 and ESDIRK3 methods of Chapter 5, respectively.

The tolerances and parameters used for the simulations are given in Table 7.7. The non–adaptive algorithms are solved using ϵ_{exact} for the Newton tolerance. In the following

Table 7.7: Simulation Parameters

ϵ_{exact}	ϵ_{smooth}	N_{smooth}	ϵ_{gmres}
10^{-6}	10^{-2}	10	10^{-3}

simulations, whenever a matrix appears in any of the algorithms of Chapter 6, it has been constructed and stored. If a matrix inverse appears in any of the algorithms, the corresponding matrix has been factored and stored in advanced for use within the GMRES iteration. The matrix construction and factorization time is included in the overall simulation time. The case where the matrices are not stored, i.e. constructed and factored on demand, is addressed in the sequel. In all cases, the zero–vector is used as an initial condition. The effect of using a more accurate initial condition as described in Section 7.2.3 is also addressed in the sequel.

7.3.1 Adaptive Versus Fixed

The resulting data for the sine wave voltage simulation are plotted in Fig. 7.15, separated by discretization method. In general, we see the adaptive algorithms outperform the fixed step size algorithms by between a factor of 2× to 3×. The speedup achievable moving from a fixed to an adaptive algorithm is limited by the number of "outer" iterations. This corresponds to the number of Newton iteration for the steady–state methods and number of simulated periods for transient analysis.

It appears that the adaptive transient algorithms achieve much greater speedup that the other methods. This is because: (1) transient analysis takes many more outer iterations to solve the equations to the same accuracy on a fixed grid of time points; and (2) we have limited the number of smoothing iterations. A small number of smoothing iterations is sufficient for the algorithms using Newton's method because of the fast convergence afforded by the root finding method. The larger smoothing tolerance typically limits the total number of iterations to 2 or less. The convergence of transient analysis is quite slow by comparison, and the same number of smoothing iterations does not reduce the residual of the discrete problem by nearly the same amount after the grid is refined. Even with this optimistic estimation of the transient analysis simulation times, the single and multiple shooting algorithms still outperform at practical accuracy levels of interest.

7.3.2 Global Comparison

The results of the adaptive and fixed step size algorithms have been collected in Fig. 7.16 to facilitate a visual comparison. The adaptive harmonic balance algorithm reports smaller simulation times over all error levels. Among the time domain methods, transient analysis is generally the slowest while the shooting methods are fairly competitive with each other. The adaptive multiple shooting methods outperform the adaptive single shooting methods. The fixed step size single shooting methods tend to be better than the fixed step size multiple shooting algorithms, although the difference is small. Asymptotically, the methods using ESDIRK3 are best as expected, but the results are much closer for intermediate accuracy levels.

In particular, the results are quite striking if one focuses on the comparing the "typical" method of performing steady–state analysis, transient analysis using the implicit–Euler method (TRA1 and TRF1), with the other algorithms. At modest accuracy levels, we generally see at least an order of magnitude improvement in simulation time. The adaptive harmonic balance method HBA1 shows more than two orders of magnitude improvement over the fixed step size transient analysis algorithm TRF1.

7.3.3 Practical Accuracy Considerations

We are not necessarily interested in the accuracy of the solution in and of itself, but rather a few quantities derived from the solution such as torque and losses. Since the losses depend on the derivative of the solution and the torque does not, it is generally safe to assume that the accuracy of the torque calculation is higher than the accuracy of the loss calculation. The simulation time as a function of the error in the stator and rotor conduction losses calculations is plotted in Fig. 7.17. The convergence demonstrated in these plots are considerably more noisy than plots of the discretization error because the losses are a nonlinear function of both the solution and its derivative, but some general trends can be observed.

Since the bulk of the losses are due to conduction in the stator windings, the accuracy of this number is most important for efficiency calculations. The algorithms using the implicit–Euler method are noticeably less accurate than the harmonic balance methods and the time domain algorithms using the higher order Runge–Kutta methods. The rotor losses are important when the effects of rotor heating are being examined. The harmonic balance method shows a clear advantage for this calculation. Compared to implicit–Euler based transient analysis, the more advanced steady–state analysis algorithms show a significant improvement in terms of both speed and accuracy for both calculations.

There is some difference observed in the loss errors between the adaptive and fixed step size algorithms. This is mostly due to differences in time step selection. In some cases, the adaptive algorithms are not being solved to the same accuracy as the fixed step size algorithms on the finest grid due to the limited number of smoothing iterations. Practically speaking, these differences are not very important with some notable exception in the rotor loss calculations: The implicit–Euler and fixed step size transient analysis methods demonstrate very poor accuracy on this score.

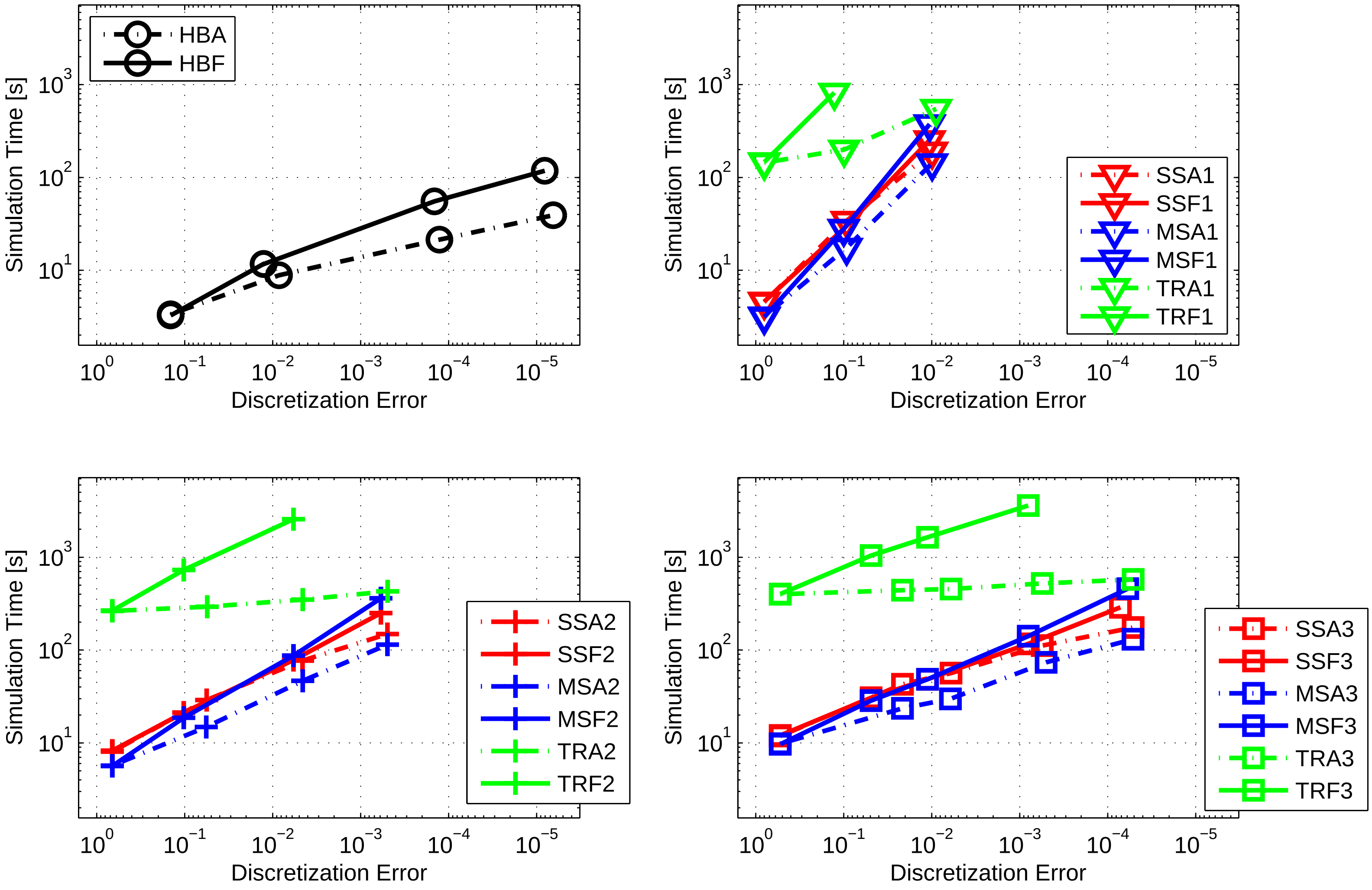

Figure 7.15: Simulation time as a function of discretization error for the voltage driven problem.

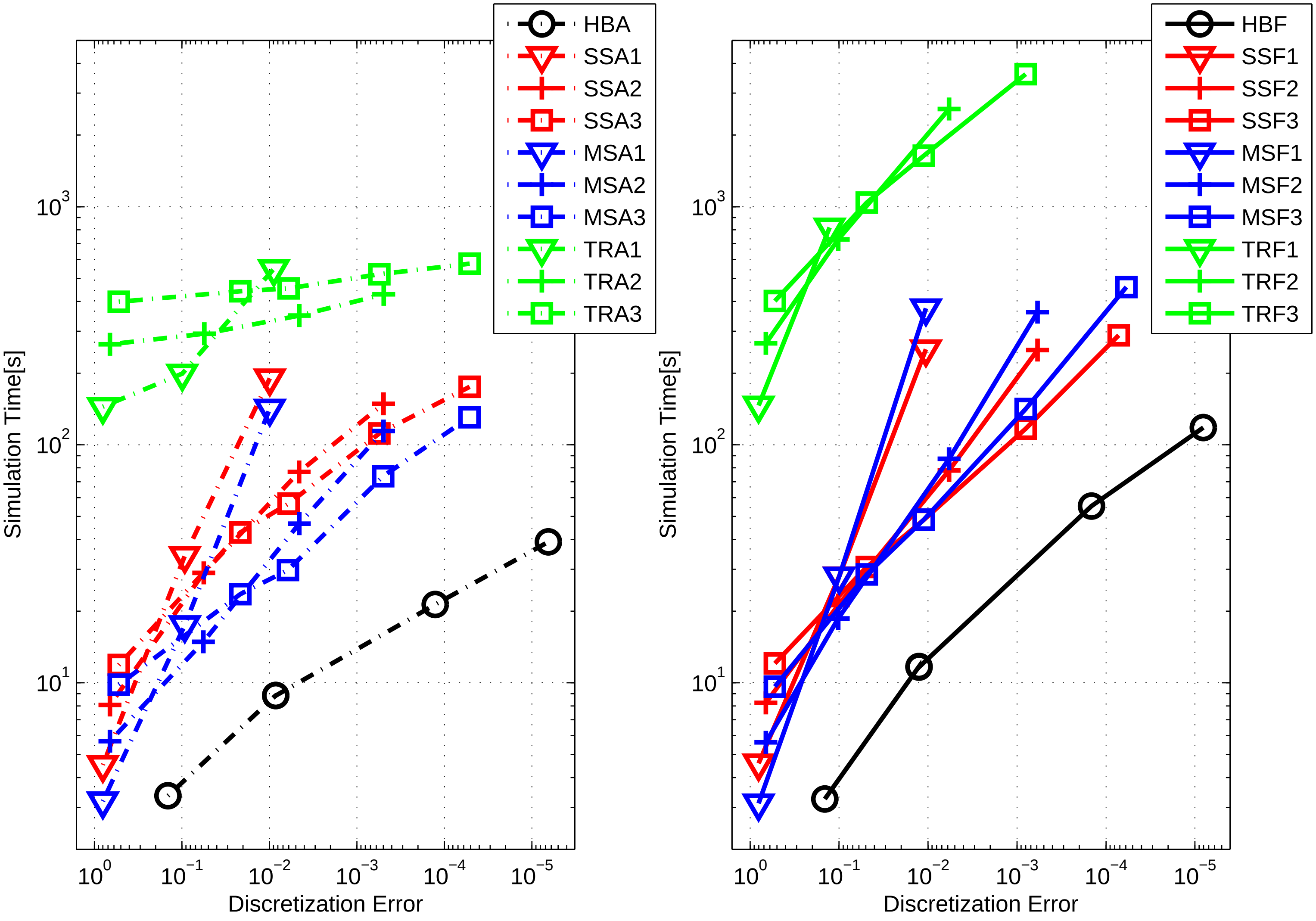

Figure 7.16: Comparison of the simulation time for the adaptive (left) and non–adaptive (right) algorithms as a function of discretization error for the voltage driven problem.

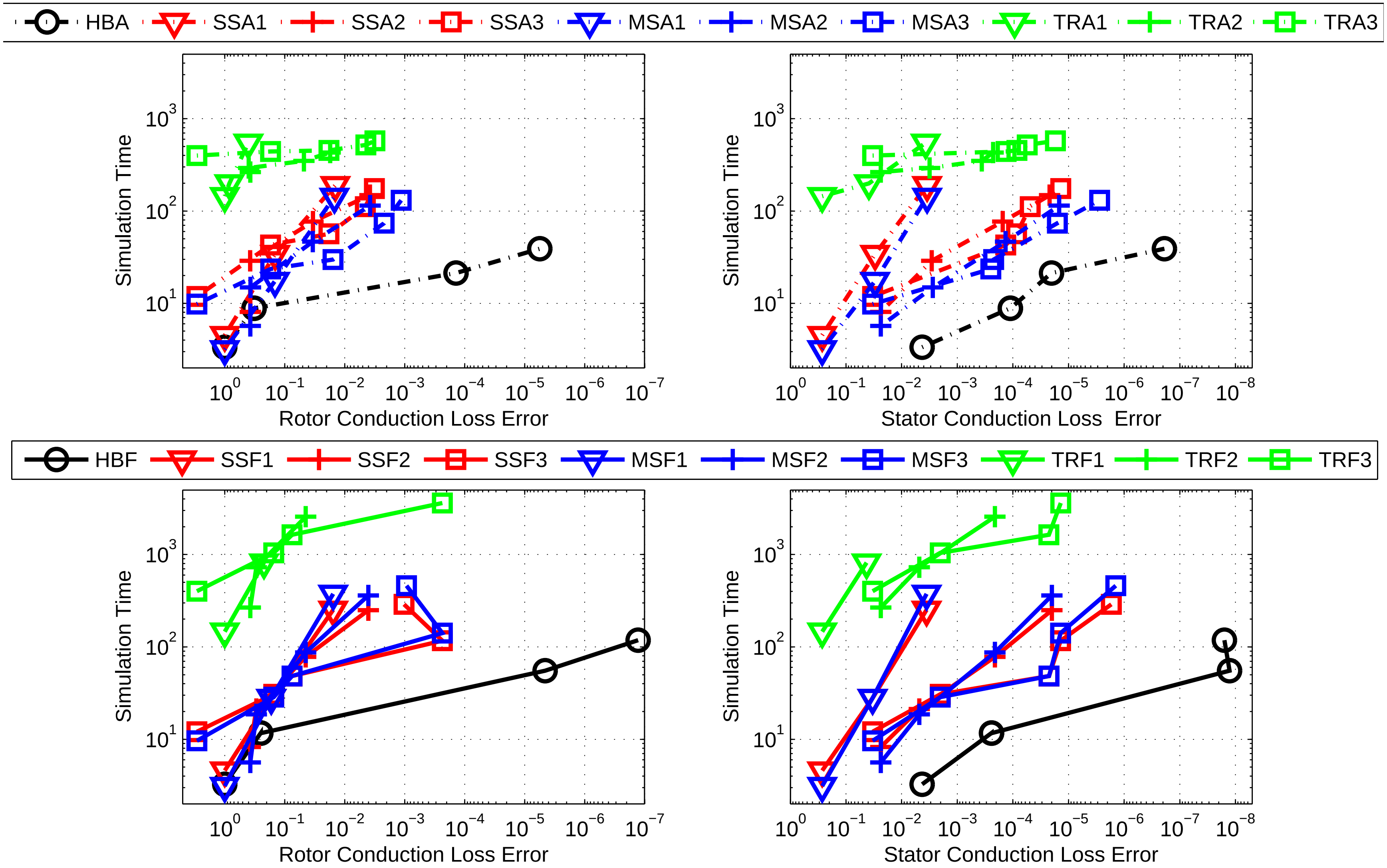

Figure 7.17: Simulation time as a function of the error in the calculated stator and rotor conduction losses.

Table 7.8: Stator and rotor losses for the sinusoidal voltage driven problem with solid rotor backiron.

	Rotor	Stator
Conduction	192W	12.8kW
Core	14.7W	580W

7.4 Factors Affecting Simulation Time

We now examine several different factors influencing the simulation times of the algorithms. The first is the effect of storing all of the necessary matrices and factorizations for use in calculating the residuals, matrix–vector products, and preconditioners within the GMRES process. Storing these matrices reduces computation time but drastically increases memory requirements. For large problems, especially in three dimensions, this may be impractical. Therefore, examining algorithm performance when all matrices are constructed on demand will give a good indication of what can be expected for three dimensional problems.

A second factor influencing simulation time are the time constants of the machine under consideration. Larger time constants generally correspond to longer simulation times, but the effect is not uniform across the algorithms. Electric machines with solid rotor backiron will exhibit larger time constants due to magnetic diffusion occurring across the conductive ferromagnetic region. This is a situation of practical interest because solid rotors are simpler to manufacture but may suffer from increased losses and heating. A plot of the average loss density calculated for this situation using the sinusoidal voltage configuration is shown in Fig. 7.18. Compared to Fig. 7.12, Fig. 7.18 demonstrates increased losses in the rotor backiron near the permanent magnets due to eddy current conduction. The eddy current conduction decreases the amplitudes of the oscillating fields away from the permanent magnets and so reduces the core losses in these areas.

Third, the ability to generate an accurate initial condition can reduce the number of Newton iterations and overall simulation time if it can be done cheaply. For many practical

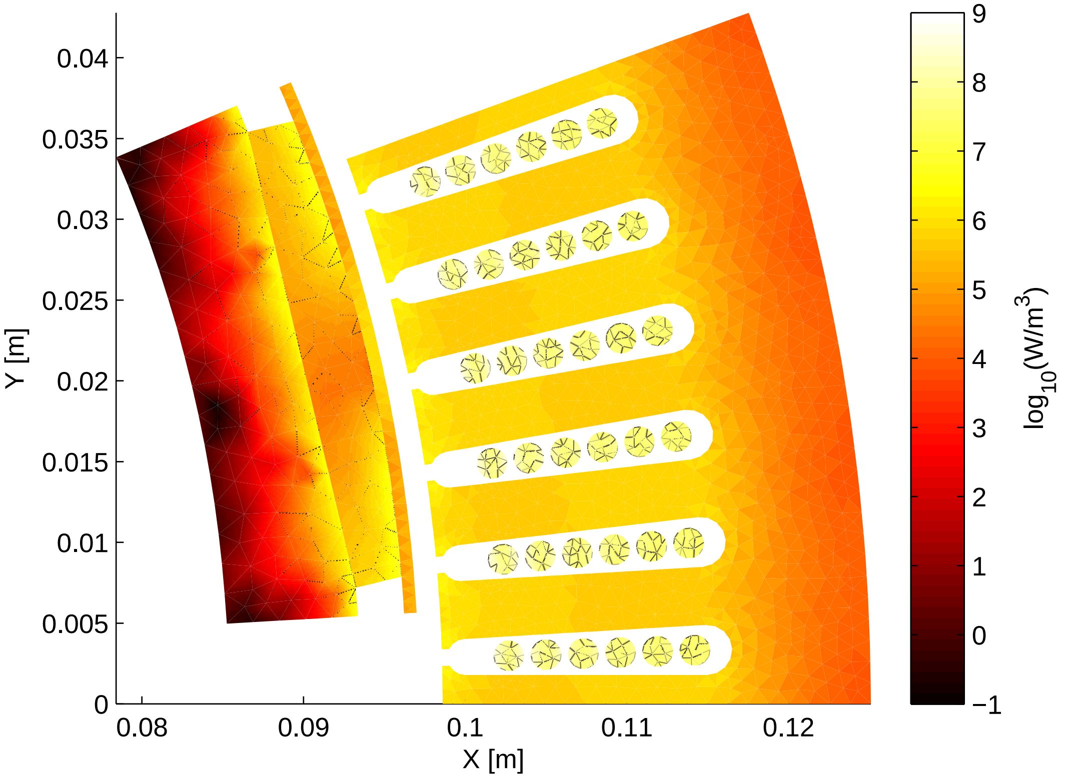

Figure 7.18: Average loss density $\mathbf{E} \cdot \mathbf{J}$ over one period from the sinusoidal voltage driven problem with solid rotor backiron.

Table 7.9: Simulation time in minutes for various algorithmic configurations of the sinusoidal voltage driven problem. The discretization error threshold was chosen as $\varepsilon = 10^{-3}$.

Rotor Backiron	Laminated				Solid			
Initial Condition	No		Yes		No		Yes	
Store Decompositions	Yes	No	Yes	No	Yes	No	Yes	No
HBA	**0.35**	**2.06**	**0.42**	**2.05**	**0.35**	**2.02**	**0.42**	**1.98**
HBF	0.93	6.71	0.59	4.30	0.94	7.03	0.57	3.82
SSA3	1.87	2.77	1.94	2.76	1.81	3.38	1.70	2.52
SSF3	2.00	4.83	2.03	3.58	3.97	15.18	2.09	7.34
MSA3	1.28	4.23	1.38	4.61	1.44	8.72	1.49	7.67
MSF3	2.43	11.77	1.86	9.18	4.71	56.38	2.55	21.98
TRA3	8.71		6.50		12.16		6.06	
TRF3	60.80		41.09		95.15		36.34	

problems, there is an intuitive way to generate an initial condition using bulk parameter circuit models to estimate the winding currents as described in Section 7.2.3. From a software implementation standpoint, this can be cumbersome to automate as it requires the ability to translate different quantities of the solution between three different models. Therefore, it is useful to investigate whether or not there is any benefit to be gained by developing such a framework.

7.4.1 Results

In the following simulations, a discretization error tolerance of $\varepsilon = 10^{-3}$ was used for the adaptive algorithms. As in the last section, the number of time steps/samples from the adaptive algorithms are used to choose a number of equally spaced times in the non–adaptive algorithms. For the time domain algorithms, ESDIRK3 is used as the numerical integration method.

Table 7.9 reports the simulation time in minutes for the various algorithmic configurations for the sinusoidal voltage driven problem. The adaptive harmonic balance method tends to

Table 7.10: Reported discretization error for various algorithmic configurations of the sinusoidal voltage driven problem. The discretization error threshold was chosen as $\varepsilon = 10^{-3}$.

Rotor Backiron	Laminated	Solid
HBA	0.13×10^{-3}	0.12×10^{-3}
HBF	0.15×10^{-3}	0.15×10^{-3}
SSA3	0.55×10^{-3}	0.53×10^{-3}
SSF3	0.80×10^{-3}	0.70×10^{-3}
MSA3	0.50×10^{-3}	0.39×10^{-3}
MSF3	0.80×10^{-3}	0.46×10^{-3}
TRA3	0.55×10^{-3}	0.53×10^{-3}
TRF3	0.80×10^{-3}	0.70×10^{-3}

be the fastest algorithm, although for each configuration there is typically another algorithm that is within roughly a factor of 4 and sometimes much closer. The single shooting method becomes more competitive with the harmonic balance method when when no matrices are stored. In each situation, the fastest steady–state method is faster than all of the transient analysis algorithms, with the fixed step size algorithms being consistently worst.

Some care must be taken when comparing the simulation time numbers since they reflect simulation results with only approximately equal accuracy. The discretization errors reported at the end of each simulation are given in Table 7.10. The harmonic balance methods actually report slightly better discretization errors than the time–domain methods.

7.4.2 Matrix Storage

The trends in these results can be understood by considering the number of Newton iterations, GMRES iterations, and the cost of each under the various configurations. Storing the matrix decompositions makes the GMRES iterations cheap and doing so tends to improve the methods with expensive matrix–vector products and preconditioners the most. This favors the algorithm with the cheapest Newton iteration in terms of cost of calculating the residual, number of iterations, and number of factorization performed to prepare the matrix–

Table 7.11: Number of observed Newton iterations N_{newton} for the non–adaptive steady–state algorithms and number of simulated periods for transient analysis.

Rotor Backiron	Laminated		Solid	
Initial Condition	No	Yes	No	Yes
HBF	8	4	8	4
SSF3	4	4	8	4
MSF3	10	7	10	7
TRF3	111	74	173	65

vector product and preconditioner subroutines. The cost of evaluating the single shooting residual is one period of transient analysis, and is much more expensive than evaluating the residual for any of the harmonic balance or multiple shooting algorithms. The latter two algorithms are much closer to parity on the other factors. The number of observed Newton iterations N_{Newton} for the non–adaptive algorithms applied to the sinusoidal voltage problem are reported in Table 7.11. The simulation time for the adaptive algorithms depend on the number of Newton iterations performed on the initial grid, as well as the number of smoothing iterations performed on each refined grid. Much of the improvement of HBA over MSA3 stems from the accuracy of the frequency domain interpolation onto the refined grid, reducing the number of smoothing iterations performed.

The number of Newton iterations necessary for the non–adaptive algorithms is also essential to understanding the theoretical maximum speedup that can be obtained when moving to an adaptive algorithm. For very small discretization error tolerances the number of smoothing iterations on each grid reduces to 1 as the interpolation error decreases as at least $\mathcal{O}(h)$, and the total simulation time is approximately twice the simulation time on the finest grid. Therefore, the maximum speedup obtainable by an adaptive algorithm is approximately $\frac{N_{Newton}}{2}$. This means the maximum speedup for the single shooting method is about 2×, while it is closer to 4× for the harmonic balance method and 5× for the multiple shooting method when the rotor is laminated. The chosen tolerance of 10^{-3} is hardly small enough to

Table 7.12: Maximum number of GMRES iterations observed over all Newton iterations.

Rotor Backiron	Laminated		Solid	
Initial Condition	No	Yes	No	Yes
HB	5	6	5	5
SS	3	3	8	11
MS	4	4	27	13

come close to these asymptotic estimates, however, and we generally observe speedup of less than the maximum. Similar arguments show that there is a limit to the speedup obtainable through the use of a non–zero initial condition.

The circumstances change somewhat when the matrix decompositions are not stored. In this case, the total number of GMRES iterations is a much more important contributor to the overall simulation time. The maximum number of GMRES iterations observed over all Newton iterations for the laminated and solid rotor backiron simulations are presented in Table 7.12. When the rotor backiron is laminated, all of the methods require a very small number of GMRES iterations. The number of iterations increase for all methods when the rotor backiron is solid. However, the time domain methods show much more sensitivity to the increased time constants produced by the solid rotor backiron than the harmonic balance methods. By way of comparison, the number of GMRES iterations for the harmonic balance method increases from approximately 5 to more than 50 when only the standard block–diagonal frequency domain preconditioner is used. That is greater than a ten–fold decrease in the number of GMRES iterations achieved by employing the novel preconditioning technique developed in Section 6.2.

7.4.3 Initial Condition

Starting from the fixed step size algorithms with a zero initial condition, moving to an adaptive version of the algorithm with a zero initial condition or generating an accurate initial condition for the non–adaptive version both tend to improve simulation time. The move from the fixed to the adaptive algorithm tends to be more beneficial than the move from zero initial condition to accurate initial condition. Asymptotically, an accurate initial condition does not improve the performance of the adaptive algorithms. If an adaptive algorithm is already quite fast, the time taken to generate the initial condition can outweigh any benefit.

The non–asymptotic results of Table 7.9 demonstrate that the slower adaptive methods do realize some benefit from the initial condition, while the fastest methods generally do not. One interesting result is the maximum number of GMRES iterations decreases significantly for the multiple shooting method in the solid rotor case with an initial condition. This indicates that the initial condition accelerates the simulation of the magnetic diffusion process in such a way that the Newton correction terms are better approximated in small dimensional Krylov subspaces generated by the residual and preconditioned Jacobian.

7.5 Conclusion

From the previous section, we can draw a few conclusions. Generally speaking, the adaptive methods are superior to the non–adaptive methods. The steady–state analysis algorithms using Newton's method are vastly superior to running transient analysis until convergence. The implicit–Euler method demonstrates deficiencies when considering the accuracy of loss calculations. The higher order Runge–Kutta methods ESDIRK2 and ESDIRK3 produce much better results.

Due to the sensitivity of the time domain methods to large time constants, the harmonic balance method is expected to perform best in these situations. When the time constants are small, usually we can find a time domain method coming within a factor of 4 of the harmonic balance method in terms of simulation time. In particular, when problem size and memory constraints dictate that all the necessary matrices and their factorizations cannot be stored, the single shooting method might be considered as it typically gives the fastest convergence in terms of number of number of Newton iterations and total number of GMRES iterations.

7.6 Future Work

Throughout this thesis, we have used a few common simplifying assumptions that have allowed us to better focus on a few particular issues. Several issues remain unexplored. Relaxing these assumptions and resolving the remaining issues is necessary to realize a fully capable steady–state analysis tool.

The main topic of this thesis is broad enough in scope that many issues relevant to the design and simulation of electric machines have been set aside through some standard simplifying assumptions in order to focus on the steady–state simulation issue. The first simplification occurs by assuming two–dimensional electric machine models. The model equations we use to develop the algorithms in this thesis are valid for two and three dimensional models. However, in non–dynamic finite–element analysis of electric machines, the methods used for solving two dimensional problems are different than those used to solve three dimensional problems owing to the difference in size and basis functions used for the discretization process. Because of the strategies employed in the algorithms developed in this thesis, a similar change must be made when moving from two– to three–dimensional steady–state analysis. Ultimately, the full development of three dimensional steady–state algorithms would result in an additional inner iterative loop with its own associated tolerances and parameters to tune.

To a first approximation, the inner–most loop in a three dimensional steady–state simulation can be regarded as orthogonal to the outer loops common to both two– and three–dimensional problems. In this respect, this thesis can be regarded as a large initial step towards a complete theory of steady–state analysis in both two and three dimensions. On the other hand, the richer structure of the matrices in the three–dimensional case may provide additional opportunities to improve these algorithms beyond the simple translation of the two dimensional algorithms suggested here.

The other major assumption in this thesis is that the effects of magnetic hysteresis are ignored. This is a difficult problem in itself and is a major source of modeling error in electric machine simulations, especially when trying to calculate losses. From experimental results, we know that the hysteresis loop of periodically excited ferromagnetic samples will converge to a multi–valued hysteresis curve. Specialized algorithms may be able to assist in calculating steady–state electric machine behavior using hysteretic magnetization models. The main difficulty lies in choosing a hysteresis model that is amenable to a reformulation in terms of a periodic solution. Even though this issue is not dealt with here directly, the somewhat non–standard treatment of non–hysteretic nonlinear magnetic materials in this dissertation may be considered an attempt to make plain where the necessary modification to the steady–state theory must be made to incorporate hysteresis.

We have also ignored the effects of invert switching harmonics in the voltage driven problem. This is perhaps one of the more interest applications of steady–state analysis. For the time–domain algorithms, it becomes necessary to deal with discontinuities in the solution, but there are techniques available to solve this problem. For the harmonic balance method, the analysis of switching harmonics requires the inclusion of a large number of basis functions and degrades the speed of this strategy. It may be possible to use the time–mapping technique [120] to improve the efficiency of the harmonic balance method for this type of problem.

Our main focus has been on the simulation of synchronous machines. Asynchronous (induction) machines are a natural candidate for simulation using steady–state methods since the rotor currents are necessarily determined dynamically. The main issue is dealing with the incommensurate slip and excitation frequencies. For the time–domain methods, the true period of the solution is equal to the least common multiple of the slip period and excitation period and can lead to unnecessarily long simulation times unless proper care is taken. This has been dealt with in other situations by using a poly–phase periodic boundary condition in the time domain [61]. For the harmonic balance method, it may be possible to deal with this issue by approaching it using either a two dimensional Fourier transform or the previously mentioned time–mapping technique.

The increasing reliance of chip manufactures on increased core count to improve processor capabilities means that algorithms must be readily parallelized if they are to benefit from future advances in processor technology. Time stepping is an inherently serial process. The typical method of parallelizing such algorithms is through a spatial partitioning. Higher degrees of parallelization come at the expense of reduced convergence rates, so one generally observes less than the optimal linear speedup (even ignoring communication overhead). The situation is much different for steady–state analysis. High performance serial algorithms resemble spatial problems where the time axis appears as an extra dimension. It turns out most of these algorithms are parallelizable by a natural partition of the time axis and can be expected to demonstrate optimal speedup, less communication overhead. The "matrix of matrices" view produces partitions where each is as large as the entire spatial part of the domain, and are therefore relatively large when compared to the spatial partition approach offered by time marching.

Appendix A

Generalized Minimum Residual Method

The generalized minimum residual method (GMRES) is an iterative algorithm which approximates the solution to the equation $Ax = b$ by minimizing the norm of the residual $r = A\hat{x} - b$ for $\hat{x}$ in the m–dimensional Krylov subspace [122, 123],

$$\mathcal{K}^m\left(b\right) = \texttt{span}\{b, Ab, A^2b, \ldots, A^{m-1}b\}. \tag{A.1}$$

Ideally, an acceptable approximation to the true solution can be found when m is much smaller than the size of A. Given $z_m = A^{m-1}b$, a key step in the algorithm involves computing $z_{m+1} = Az_m$. If A is available explicitly, z_{m+1} can be computed in the usual way. Often, however, A is only available through a function which evaluates matrix-vector products, e.g. $z_{m+1} = \texttt{A}\left(z_m\right)$. This gives rise to the term "matrix-free" in conjunction with iterative methods which only require A to be available through the computation of matrix-vector products. This fact is used extensively in the development of the steady–state algorithms in Chapter 6. Similar statements hold for solving the preconditioned system $AM^{-1}y = b$.

Bibliography

[1] M. Capek, *The Philosophical Impact of Contemporary Physics.* Literary Licensing, LLC, 2011.

[2] M. Chari and P. Silvester, “Finite-element analysis of magnetically saturated d-c machines,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-90, no. 5, pp. 2362–2372, Sept 1971.

[3] P. Silvester and M. Chari, “Finite element solution of saturable magnetic field problems,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-89, no. 7, pp. 1642–1651, Sept 1970.

[4] M. Chari and P. Silvester, “Analysis of turboalternator magnetic fields by finite elements,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-90, no. 2, pp. 454–464, March 1971.

[5] A. Wexler, “Finite-element field analysis of an inhomogeneous, anisotropic, reluctance machine rotor,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-92, no. 1, pp. 145–149, Jan 1973.

[6] P. Silvester, H. Cabayan, and B. Browne, “Efficient techniques for finite element analysis of electric machines,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-92, no. 4, pp. 1274–1281, July 1973.

[7] P. Unterweger, “Computation of magnetic fields in electrical apparatus,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-93, no. 3, pp. 991–1002, May 1974.

[8] M. Chari, “Nonlinear finite element solution of electrical machines under no-load and full-load conditions,” *Magnetics, IEEE Transactions on*, vol. 10, no. 3, pp. 686–689, Sep 1974.

[9] K. Binns, M. Jabbar, and W. Barnard, “A rapid method of computation of the magnetic field of permanent magnets,” *Magnetics, IEEE Transactions on*, vol. 11, no. 5, pp. 1538–1540, Sep 1975.

[10] P. Silvester and P. Rafinejad, “Curvilinear finite elements for two-dimensional saturable magnetic fields,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-93, no. 6, pp. 1861–1870, Nov 1974.

[11] P. Rafinejad and J.-C. Sabonnadiere, “Finite element computer programs in design of electromagnetic devices,” *Magnetics, IEEE Transactions on*, vol. 12, no. 5, pp. 575–578, Sep 1976.

[12] N. Demerdash and T. Nehl, “Flexibility and economics of implementation of the finite element and difference techniques in nonlinear magnetic fields of power devices,” *Magnetics, IEEE Transactions on*, vol. 12, no. 6, pp. 1036–1038, Nov 1976.

[13] P. SILVESTER and C. R. S. HASLAM, “Magnetotelluric modelling by the finite element method*,” *Geophysical Prospecting*, vol. 20, no. 4, pp. 872–891, 1972. [Online]. Available: http://dx.doi.org/10.1111/j.1365-2478.1972.tb00672.x

[14] M. Chari, “Finite-element solution of the eddy-current problem in magnetic structures,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-93, no. 1, pp. 62–72, Jan 1974.

[15] M. Chari and Z. Csendes, “Finite element analysis of the skin effect in current carrying conductors,” *Magnetics, IEEE Transactions on*, vol. 13, no. 5, pp. 1125–1127, Sep 1977.

[16] N. Demerdash and T. Nehl, “An evaluation of the methods of finite elements and finite differences in the solution of nonlinear electromagnetic fields in electrical machines,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-98, no. 1, pp. 74–87, Jan 1979.

[17] M. Chari, “Finite element analysis of electrical machinery and devices,” *Magnetics, IEEE Transactions on*, vol. 16, no. 5, pp. 1014–1019, Sep 1980.

[18] S. H. Minnich, “Incremental permeabilities for transient analysis of large turbine generators by the finite-element method,” *Journal of Applied Physics*, vol. 52, no. 3, 1981.

[19] N. Demerdash, T. Nehl, O. Mohammed, R. Miller, and F. Fouad, “Solution of eddy current problems using three dimensional finite element complex magnetic vector potential,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-101, no. 11, pp. 4222–4229, Nov 1982.

[20] J. Hwang and W. Lord, “Finite element analysis of the magnetic field distribution inside a rotating ferromagnetic bar,” *Magnetics, IEEE Transactions on*, vol. 10, no. 4, pp. 1113–1118, Dec 1974.

[21] N. Burais, A. Foggia, A. Nicolas, J. Pascal, and J.-C. Sabonnadiere, “Numerical solution of eddy currents problems including moving conducting parts,” *Magnetics, IEEE Transactions on*, vol. 20, no. 5, pp. 1995–1997, Sep 1984.

[22] J. Bigeon, J.-C. Sabonnadiere, and J. L. Coulomb, “Finite element analysis of an electromagnetic brake,” *Magnetics, IEEE Transactions on*, vol. 19, no. 6, pp. 2632–2634, Nov 1983.

[23] F. Fouad, T. Nehl, and N. Demerdash, “Magnetic field modeling of permanent magnet type electronically operated synchronous machines using finite elements,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-100, no. 9, pp. 4125–4135, Sept 1981.

[24] S. Ratnajeevan and H. Hoole, “Rotor motion in the dynamic finite element analysis of rotating electrical machinery,” *Magnetics, IEEE Transactions on*, vol. 21, no. 6, pp. 2292–2295, Nov 1985.

[25] A. Abdel-Razek, J. L. Coulomb, M. Feliachi, and J.-C. Sabonnadiere, “Conception of an air-gap element for the dynamic analysis of the electromagnetic field in electric machines,” *Magnetics, IEEE Transactions on*, vol. 18, no. 2, pp. 655–659, Mar 1982.

[26] S. Cristina and A. Di Napoli, “Combination of finite and boundary elements for magnetic field analysis,” *Magnetics, IEEE Transactions on*, vol. 19, no. 6, pp. 2337–2339, Nov 1983.

[27] M. Feliachi, J. L. Coulomb, and H. Mansir, “Second order air-gap element for the dynamic finite-element analysis of the electromagnetic field in electric machines,” *Magnetics, IEEE Transactions on*, vol. 19, no. 6, pp. 2300–2303, Nov 1983.

[28] P. Silvester and M.-S. Hsieh, “Finite-element solution of 2-dimensional exterior-field problems,” *Electrical Engineers, Proceedings of the Institution of*, vol. 118, no. 12, pp. 1743–1747, December 1971.

[29] B. McDonald and A. Wexler, “Finite-element solution of unbounded field problems,” *Microwave Theory and Techniques, IEEE Transactions on*, vol. 20, no. 12, pp. 841–847, Dec 1972.

[30] P. Silvester, D. Lowther, C. Carpenter, and E. Wyatt, “Exterior finite elements for 2-dimensional field problems with open boundaries,” *Electrical Engineers, Proceedings of the Institution of*, vol. 124, no. 12, pp. 1267–1270, December 1977.

[31] D. Lowther, C. Rajanathan, and P. Silvester, “A finite element technique for solving 2-d open boundary problems,” *Magnetics, IEEE Transactions on*, vol. 14, no. 5, pp. 467–469, Sep 1978.

[32] A. Foggia, J. Sabonnadiere, and P. Silvester, “Finite element solution of saturated travelling magnetic field problems,” *Power Apparatus and Systems, IEEE Transactions on*, vol. 94, no. 3, pp. 866–871, May 1975.

[33] A. Hannalla and D. MacDonald, “Numerical analysis of transient field problems in electrical machines,” *Electrical Engineers, Proceedings of the Institution of*, vol. 123, no. 9, pp. 893–898, September 1976.

[34] J. MacBain, “A numerical analysis of time-dependent two-dimensional magnetic fields,” *Magnetics, IEEE Transactions on*, vol. 17, no. 6, pp. 3259–3261, Nov 1981.

[35] N. Burais and G. Grellet, “Numerical modelling of iron losses in ferromagnetic steel plate,” *Magnetics, IEEE Transactions on*, vol. 18, no. 2, pp. 558–562, Mar 1982.

[36] U. Jeske, “Eddy current calculation in 3d using the finite element method,” *Magnetics, IEEE Transactions on*, vol. 18, no. 2, pp. 426–430, Mar 1982.

[37] S. Tandon, A. Armor, and M. Chari, “Nonlinear transient finite element field computation for electrical machines and devices,” *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-102, no. 5, pp. 1089–1096, May 1983.

[38] A. Kamar, “Solution of nonlinear eddy current problems using residual finite element method for space and time discretization,” *Magnetics, IEEE Transactions on*, vol. 19, no. 5, pp. 2204–2206, Sep 1983.

[39] F. Bouillault and A. Razek, “Dynamic model for eddy current calculation in saturated electric machines,” *Magnetics, IEEE Transactions on*, vol. 19, no. 6, pp. 2639–2642, Nov 1983.

[40] ——, “Eddy currents due to stator teeth in synchronous machine rotors,” *Magnetics, IEEE Transactions on*, vol. 20, no. 5, pp. 1939–1941, Sep 1984.

[41] A. Hannalla, “Analysis of transient field problems in electrical machines allowing for end leakage and external reactances,” *Magnetics, IEEE Transactions on*, vol. 17, no. 2, pp. 1240–1243, Mar 1981.

[42] A. Konrad, “The numerical solution of steady-state skin effect problems–an integrodifferential approach,” *Magnetics, IEEE Transactions on*, vol. 17, no. 1, pp. 1148–1152, Jan 1981.

[43] T. Nakata and N. Takahashi, “Direct finite element analysis of flux and current distributions under specified conditions,” *Magnetics, IEEE Transactions on*, vol. 18, no. 2, pp. 325–330, Mar 1982.

[44] J. Weiss and Z. Cendes, “Efficient finite element solution of multipath eddy current problems,” *Magnetics, IEEE Transactions on*, vol. 18, no. 6, pp. 1710–1712, Nov 1982.

[45] A. Konrad, “Integrodifferential finite element formulation of two-dimensional steady-state skin effect problems,” *Magnetics, IEEE Transactions on*, vol. 18, no. 1, pp. 284–292, Jan 1982.

[46] D. Shen, G. Meunier, J. L. Coulomb, and J.-C. Sabonnadiere, “Solution of magnetic fields and electrical circuits combined problems,” *Magnetics, IEEE Transactions on*, vol. 21, no. 6, pp. 2288–2291, Nov 1985.

[47] S. Zhi-ming, X. De-xin, and H. Cheng-qian, “The finite element solution of transient axisymmetrical nonlinear eddy-current field problems,” *Magnetics, IEEE Transactions on*, vol. 21, no. 6, pp. 2303–2306, Nov 1985.

[48] E. Strangas, "Coupling the circuit equations to the non-linear time dependent field solution in inverter driven induction motors," *Magnetics, IEEE Transactions on*, vol. 21, no. 6, pp. 2408–2411, Nov 1985.

[49] S. Minnich, S. Tandon, and D. Atkinson, "Comparison of two methods for modeling large-signal alternating magnetic fields using finite-elements," *Power Apparatus and Systems, IEEE Transactions on*, vol. PAS-103, no. 10, pp. 2952–2960, Oct 1984.

[50] B. Luetke-Daldrup, "Comparison of exact and approximate finite-element solution of the two-dimensional nonlinear eddy-current problem with measurements," *Magnetics, IEEE Transactions on*, vol. 20, no. 5, pp. 1936–1938, Sep 1984.

[51] A. Konrad, "Eddy currents and modelling," *Magnetics, IEEE Transactions on*, vol. 21, no. 5, pp. 1805–1810, Sep 1985.

[52] T. Hara, T. Naito, and J. Umoto, "Time-periodic finite element method for nonlinear diffusion equations," *Magnetics, IEEE Transactions on*, vol. 21, no. 6, pp. 2261–2264, Nov 1985.

[53] T. Nakata, N. Takahashi, K. Fujiwara, and A. Ahagon, "3-d non-linear eddy current analysis using the time-periodic finite element method," *Magnetics, IEEE Transactions on*, vol. 25, no. 5, pp. 4150–4152, Sep 1989.

[54] T. Nakata, N. Takahashi, K. Fujiwara, K. Muramatsu, H. Ohashi, and H. L. Zhu, "Practical analysis of 3-d dynamic nonlinear magnetic field using time-periodic finite element method," *Magnetics, IEEE Transactions on*, vol. 31, no. 3, pp. 1416–1419, May 1995.

[55] K. Muramatsu, N. Takahashi, T. Nakata, M. Nakano, Y. Ejiri, and J. Takehara, "3-d time-periodic finite element analysis of magnetic field in non-oriented materials taking into account hysteresis characteristics," *Magnetics, IEEE Transactions on*, vol. 33, no. 2, pp. 1584–1587, Mar 1997.

[56] T. Matsuo and M. Shimasaki, "Time-periodic finite-element method for hysteretic eddy-current analysis," *Magnetics, IEEE Transactions on*, vol. 38, no. 2, pp. 549–552, Mar 2002.

[57] X. Wang and D. Xie, "Analysis of induction motor using field-circuit coupled time-periodic finite element method taking account of hysteresis," *Magnetics, IEEE Transactions on*, vol. 45, no. 3, pp. 1740–1743, March 2009.

[58] O. Biro and K. Preis, "An efficient time domain method for nonlinear periodic eddy current problems," *Magnetics, IEEE Transactions on*, vol. 42, no. 4, pp. 695–698, April 2006.

[59] O. Biro, G. Koczka, and K. Preis, "Fast time-domain finite element analysis of 3-d nonlinear time-periodic eddy current problems with T, $\phi - \phi$ formulation," *Magnetics, IEEE Transactions on*, vol. 47, no. 5, pp. 1170–1173, May 2011.

[60] Y. Takahashi, T. Iwashita, H. Nakashima, T. Tokumasu, M. Fujita, S. Wakao, K. Fujiwara, and Y. Ishihara, "Parallel time-periodic finite-element method for steady-state analysis of rotating machines," *Magnetics, IEEE Transactions on*, vol. 48, no. 2, pp. 1019–1022, Feb 2012.

[61] Y. Takahashi, T. Tokumasu, M. Fujita, T. Iwashita, H. Nakashima, S. Wakao, and K. Fujiwara, "Time-domain parallel finite-element method for fast magnetic field analysis of induction motors," *Magnetics, IEEE Transactions on*, vol. 49, no. 5, pp. 2413–2416, May 2013.

[62] S. Yamada and K. Bessho, "Harmonic field calculation by the combination of finite element analysis and harmonic balance method," *Magnetics, IEEE Transactions on*, vol. 24, no. 6, pp. 2588–2590, Nov 1988.

[63] S. Yamada, K. Bessho, and J. Lu, "Harmonic balance finite element method applied to nonlinear ac magnetic analysis," *Magnetics, IEEE Transactions on*, vol. 25, no. 4, pp. 2971–2973, Jul 1989.

[64] S. Yamada, P. Biringer, K. Hirano, and K. Bessho, "Finite element analysis of nonlinear dynamic magnetic field with dc component in the harmonic domain," *Magnetics, IEEE Transactions on*, vol. 26, no. 5, pp. 2199–2201, Sep 1990.

[65] J. Lu, S. Yamada, and K. Bessho, "Time-periodic magnetic field analysis with saturation and hysteresis characteristics by harmonic balance finite element method," *Magnetics, IEEE Transactions on*, vol. 26, no. 2, pp. 995–998, Mar 1990.

[66] S. Yamada, P. Biringer, and K. Bessho, "Calculation of nonlinear eddy-current problems by the harmonic balance finite element method," *Magnetics, IEEE Transactions on*, vol. 27, no. 5, pp. 4122–4125, Sep 1991.

[67] S. Yamada, K. Bessho, and M. Kitagawa, "Finite element analysis of nonlinear magnetic devices combined with circuit equations by tableau approach," *Magnetics, IEEE Transactions on*, vol. 28, no. 5, pp. 2256–2258, Sep 1992.

[68] J. Lu, S. Yamada, and K. Bessho, "Harmonic balance finite element method taking account of external circuits and motion," *Magnetics, IEEE Transactions on*, vol. 27, no. 5, pp. 4024–4027, Sep 1991.

[69] R. Albanese, E. Coccorese, R. Martone, G. Miano, and G. Rubinacci, "Periodic solutions of nonlinear eddy current problems in three-dimensional geometries," *Magnetics, IEEE Transactions on*, vol. 28, no. 2, pp. 1118–1121, Mar 1992.

[70] H. Hedia, J. Remacle, P. Dular, A. Nicolet, A. Genon, and W. Legros, "A sinusoidal magnetic field computation in nonlinear materials," *Magnetics, IEEE Transactions on*, vol. 31, no. 6, pp. 3527–3529, Nov 1995.

[71] J. W. Lu, S. Yamada, and H. Harrison, “Application of harmonic balance-finite element method (hbfem) in the design of switching power supplies,” *Power Electronics, IEEE Transactions on*, vol. 11, no. 2, pp. 347–355, Mar 1996.

[72] G. Paoli, O. Biro, and G. Buchgraber, “Complex representation in nonlinear time harmonic eddy current problems,” *Magnetics, IEEE Transactions on*, vol. 34, no. 5, pp. 2625–2628, Sep 1998.

[73] J. Gyselinck, P. Dular, C. Geuzaine, and W. Legros, “Harmonic-balance finite-element modeling of electromagnetic devices: a novel approach,” *Magnetics, IEEE Transactions on*, vol. 38, no. 2, pp. 521–524, Mar 2002.

[74] J. Gyselinck, L. Vandevelde, P. Dular, C. Geuzaine, and W. Legros, “A general method for the frequency domain fe modeling of rotating electromagnetic devices,” *Magnetics, IEEE Transactions on*, vol. 39, no. 3, pp. 1147–1150, May 2003.

[75] R. Pascal, P. Conraux, and J.-M. Bergheau, “Coupling between finite elements and boundary elements for the numerical simulation of induction heating processes using a harmonic balance method,” *Magnetics, IEEE Transactions on*, vol. 39, no. 3, pp. 1535–1538, May 2003.

[76] S. Ausserhofer, O. Biro, and K. Preis, “An efficient harmonic balance method for nonlinear eddy-current problems,” *Magnetics, IEEE Transactions on*, vol. 43, no. 4, pp. 1229–1232, April 2007.

[77] G. Koczka, S. Auberhofer, O. Biro, and K. Preis, “Optimal convergence of the fixed-point method for nonlinear eddy current problems,” *Magnetics, IEEE Transactions on*, vol. 45, no. 3, pp. 948–951, March 2009.

[78] X. Zhao, J. Lu, L. Li, H. Li, Z. Cheng, and T. Lu, “Fixed-point harmonic-balanced method for dc-biasing hysteresis analysis using the neural network and consuming function,” *Magnetics, IEEE Transactions on*, vol. 48, no. 11, pp. 3356–3359, Nov 2012.

[79] X. Zhao, L. Li, J. Lu, Z. Cheng, and T. Lu, “Characteristics analysis of the square laminated core under dc-biased magnetization by the fixed-point harmonic-balanced fem,” *Magnetics, IEEE Transactions on*, vol. 48, no. 2, pp. 747–750, Feb 2012.

[80] O. Biro, G. Koczka, G. Leber, K. Preis, and B. Wagner, “Finite element analysis of three-phase three-limb power transformers under dc bias,” *Magnetics, IEEE Transactions on*, vol. 50, no. 2, pp. 565–568, Feb 2014.

[81] H. Sande, F. Henrotte, H. De Gersem, and K. Hameyer, “An effective reluctivity model for nonlinear and anisotropic materials in time-harmonic finite element computations,” *Magnetics, IEEE Transactions on*, vol. 41, no. 5, pp. 1508–1511, May 2005.

[82] O. Deblecker and J. Lobry, “A new efficient technique for harmonic-balance finite-element analysis of saturated electromagnetic devices,” *Magnetics, IEEE Transactions on*, vol. 42, no. 4, pp. 535–538, April 2006.

[83] I. Ciric and F. Hantila, “An efficient harmonic method for solving nonlinear time-periodic eddy-current problems,” *Magnetics, IEEE Transactions on*, vol. 43, no. 4, pp. 1185–1188, April 2007.

[84] H. Igarashi and K. Watanabe, “Complex adjoint variable method for finite-element analysis of eddy current problems,” *Magnetics, IEEE Transactions on*, vol. 46, no. 8, pp. 2739–2742, Aug 2010.

[85] X. Zhao, J. Lu, L. Li, Z. Cheng, and T. Lu, “Analysis of the dc bias phenomenon by the harmonic balance finite-element method,” *Power Delivery, IEEE Transactions on*, vol. 26, no. 1, pp. 475–485, Jan 2011.

[86] S. Li and H. Hofmann, “Numerically efficient steady-state finite-element analysis of magnetically saturated electromechanical devices,” *Magnetics, IEEE Transactions on*, vol. 39, no. 6, pp. 3481–3485, Nov 2003.

[87] D. Zhong and H. Hofmann, “Steady-state finite-element solver for rotor eddy currents in permanent-magnet machines using a shooting-newton/gmres approach,” *Magnetics, IEEE Transactions on*, vol. 40, no. 5, pp. 3249–3253, Sept 2004.

[88] Y. Takahashi, T. Tokumasu, A. Kameari, H. Kaimori, M. Fujita, T. Iwashita, and S. Wakao, “Convergence acceleration of time-periodic electromagnetic field analysis by the singularity decomposition-explicit error correction method,” *Magnetics, IEEE Transactions on*, vol. 46, no. 8, pp. 2947–2950, Aug 2010.

[89] H. Katagiri, Y. Kawase, T. Yamaguchi, T. Tsuji, and Y. Shibayama, “Improvement of convergence characteristics for steady-state analysis of motors with simplified singularity decomposition-explicit error correction method,” *Magnetics, IEEE Transactions on*, vol. 47, no. 5, pp. 1458–1461, May 2011.

[90] Y. Takahashi, H. Kaimori, A. Kameari, T. Tokumasu, M. Fujita, S. Wakao, T. Iwashita, K. Fujiwara, and Y. Ishihara, “Convergence acceleration in steady state analysis of synchronous machines using time-periodic explicit error correction method,” *Magnetics, IEEE Transactions on*, vol. 47, no. 5, pp. 1422–1425, May 2011.

[91] H. Katagiri, Y. Kawase, T. Yamaguchi, T. Tsuji, and Y. Shibayama, “Improvement of convergence characteristics for steady-state analysis of motors with simplified singularity decomposition-explicit error correction method*,” *Magnetics, IEEE Transactions on*, vol. 47, no. 6, pp. 1786–1789, June 2011.

[92] H. Igarashi, Y. Watanabe, and Y. Ito, “Why error correction methods realize fast computations,” *Magnetics, IEEE Transactions on*, vol. 48, no. 2, pp. 415–418, Feb 2012.

[93] A. Nicolet and F. Delince, “Implicit runge-kutta methods for transient magnetic field computation,” *Magnetics, IEEE Transactions on*, vol. 32, no. 3, pp. 1405–1408, May 1996.

[94] T. Noda, K. Takenaka, and T. Inoue, "Numerical integration by the 2-stage diagonally implicit runge-kutta method for electromagnetic transient simulations," *Power Delivery, IEEE Transactions on*, vol. 24, no. 1, pp. 390–399, Jan 2009.

[95] H. Li, S. Ho, and W. Fu, "Application of multi-stage diagonally-implicit runge-kutta algorithm to transient magnetic field computation using finite element method," *Magnetics, IEEE Transactions on*, vol. 48, no. 2, pp. 279–282, Feb 2012.

[96] F. Cameron, R. Piche, and K. Forsman, "Variable step size time integration methods for transient eddy current problems," *Magnetics, IEEE Transactions on*, vol. 34, no. 5, pp. 3319–3322, Sep 1998.

[97] H. Wang, S. Taylor, J. Simkin, C. Biddlecombe, and B. Trowbridge, "An adaptive-step time integration method applied to transient magnetic field problems," *Magnetics, IEEE Transactions on*, vol. 37, no. 5, pp. 3478–3481, Sep 2001.

[98] G. Benderskaya, W. Ackermann, H. De Gersem, and T. Weiland, "Adaptive time stepping for electromagnetic models with sinusoidal dynamics," *Magnetics, IEEE Transactions on*, vol. 44, no. 6, pp. 1262–1265, June 2008.

[99] M. Clemens, M. Wilke, and T. Weiland, "Extrapolation strategies in numerical schemes for transient magnetic field simulations," *Magnetics, IEEE Transactions on*, vol. 39, no. 3, pp. 1171–1174, May 2003.

[100] G. Benderskaya, M. Clemens, H. De Gersem, and T. Weiland, "Embedded runge-kutta methods for field-circuit coupled problems with switching elements," *Magnetics, IEEE Transactions on*, vol. 41, no. 5, pp. 1612–1615, May 2005.

[101] G. Benderskaya, H. De Gersem, and T. Weiland, "Integration over discontinuities in field-circuit coupled simulations with switching elements," *Magnetics, IEEE Transactions on*, vol. 42, no. 4, pp. 1031–1034, April 2006.

[102] K. Geldhof, T. Vyncke, F. De Belie, L. Vandevelde, J. Melkebeek, and R. Boel, "Embedded runge-kutta methods for the integration of a current control loop in an srm dynamic finite element model," *Science, Measurement Technology, IET*, vol. 1, no. 1, pp. 17–20, January 2007.

[103] W. Fu, S. Ho, and P. Zhou, "Reduction of computing time for steady-state solutions of magnetic field and circuit coupled problems using time-domain finite-element method," *Magnetics, IEEE Transactions on*, vol. 48, no. 11, pp. 3363–3366, Nov 2012.

[104] G. Bertotti, *Hysteresis in Magnetism: For Physicists, Materials Scientists, and Engineers*, ser. Electromagnetism. Academic Press.

[105] A. Hubert and R. Schafer, *Magnetic Domains: The Analysis of Magnetic Microstructures*. Springer-Verlag, 2008.

[106] J. R. Melcher, *Continuum Electromechanics*. Cambridge, MA: MIT Press, 1981. [Online]. Available: http://ocw.mit.edu/ans7870/resources/melcher/originals/cem_100.pdf

[107] J. Oden and J. Reddy, *An Introduction to the Mathematical Theory of Finite Elements*, ser. Dover Books on Engineering. Dover Publications, Mar. 2011.

[108] J. Mizia, K. Adamiak, A. Eastham, and G. Dawson, "Finite element force calculation: comparison of methods for electric machines," *Magnetics, IEEE Transactions on*, vol. 24, no. 1, pp. 447–450, Jan 1988.

[109] Z. Ren, "Comparison of different force calculation methods in 3d finite element modelling," *Magnetics, IEEE Transactions on*, vol. 30, no. 5, pp. 3471–3474, Sep 1994.

[110] S. Salon, S. Bhatia, and D. Burow, "Some aspects of torque calculations in electrical machines," *Magnetics, IEEE Transactions on*, vol. 33, no. 2, pp. 2018–2021, Mar 1997.

[111] L. A. Piegel, *The NURBS Book*, ser. Monographs in Visual Communication. Springer-Verlag.

[112] S. J. Salon, *Finite Element Analysis of Electrical Machines.* Norwell, MA: Kluwer Academic Publishers, 1995.

[113] N. Bianchi, *Electrical Machine Analysis Using Finite Elements.* Boca Raton, FL: CRC Press, 2005.

[114] E. Hairer and W. Gerhard, *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, ser. Springer Series in Computational Mathematics. Springer-Verlag.

[115] F. Cameron, "A class of low order dirk methods for a class of daes," *Applied Numerical Mathematics*, vol. 31, no. 1, pp. 1 – 16, 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0168927498001238

[116] R. Williams, K. Burrage, I. Cameron, and M. Kerr, "A four-stage index 2 diagonally implicit runge–kutta method," *Applied Numerical Mathematics*, vol. 40, no. 3, pp. 415 – 432, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0168927401000903

[117] J. Pries and H. Hofmann, "Steady-state algorithms for nonlinear time-periodic magnetic diffusion problems using diagonally-implicit runge-kutta methods," *Magnetics, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.

[118] U. Trottenberg, C. Oosterlee, and A. Schüller, *Multigrd.* Academic Press, 2001.

[119] J. P. Boyd, *Chebyshev and Fourier Spectral Methods.* Dover Publications, 2001.

[120] O. Nastov and J. White, "Time-mapped harmonic balance," in *Design Automation Conference, 1999. Proceedings. 36th*, 1999, pp. 641–646.

[121] "Motorproto: Electric machine simulation toolbox in matlab," https://github.com/MPEL/MotorProto.

[122] Y. Saad and M. Schultz, "Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 3, pp. 856–869, 1986.

[123] Y. Saad, *Iterative Methods for Sparse Linear Systems.* Society for Industrial and Applied Mathematics, 2003.

[124] L. Skvortsov, "Diagonally implicit rungekutta methods for differential algebraic equations of indices two and three," *Computational Mathematics and Mathematical Physics*, vol. 50, no. 6, pp. 993–1005, 2010. [Online]. Available: http://dx.doi.org/10.1134/S0965542510060072

[125] D. Dyck and P. Weicker, "Periodic steady-state solution of voltage-driven magnetic devices," *Magnetics, IEEE Transactions on*, vol. 43, no. 4, pp. 1533–1536, April 2007.

[126] A. Kvaern, "Runge-kutta methods applied to fully implicit differential-algebraic equations of index 1," *Mathematics of Computation*, vol. 54, no. 190, pp. pp. 583–625, 1990.