KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT TOEGEPASTE WETENSCHAPPEN
DEPARTEMENT COMPUTERWETENSCHAPPEN
AFDELING NUMERIEKE ANALYSE EN
TOEGEPASTE WISKUNDE
Celestijnenlaan 200A – B-3001 Heverlee

ALGEBRAIC MULTIGRID FOR TWO-DIMENSIONAL TIME-HARMONIC MAGNETIC FIELD COMPUTATIONS

Promotoren:
Prof. Dr. ir. S. Vandewalle
Prof. Dr. -Ing. K. Hameyer

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de toegepaste wetenschappen

door

Domenico LAHAYE

December 2001

KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT TOEGEPASTE WETENSCHAPPEN
DEPARTEMENT COMPUTERWETENSCHAPPEN
AFDELING NUMERIEKE ANALYSE EN
TOEGEPASTE WISKUNDE
Celestijnenlaan 200A – B-3001 Heverlee

ALGEBRAIC MULTIGRID FOR TWO-DIMENSIONAL TIME-HARMONIC MAGNETIC FIELD COMPUTATIONS

Jury:
Prof. Dr. ir. E. Aernoudt, voorzitter
Prof. Dr. ir. S. Vandewalle, promotor
Prof. Dr. -Ing. K. Hameyer, promotor
Prof. Dr. P. De Groen, (Vrije Universiteit Brussel)
Prof. Dr. S. Poedts,
Prof. Dr. ir. D. Roose,
Dr. K. Stüben (Fraunhofer Institute, Germany)

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de toegepaste wetenschappen

door

Domenico LAHAYE

U.D.C. 681.3*G13

December 2001

D/2001/7515/37

ISBN 90-5682-326-4

Rev. 1

Algebraic Multigrid for Two-Dimensional Time-Harmonic Magnetic Field Computations

Domenico Lahaye
Departement Computerwetenschappen, K.U.Leuven
Celestijnenlaan 200A, B-3001 Heverlee, België

Abstract

The finite element simulation of electric energy transducers such as machines and transformers requires solving linear algebraic system with a large number of unknowns. In practical computations this solution process requires up to 90 percent of the total simulation time. This thesis contributes to the development of efficient iterative techniques for solving the finite element linear systems. The algorithms proposed were implemented in a simulation package in such a way to allow their use in the computation of industrial models.

The models considered in this thesis are built upon two-dimensional quasi-stationary approximations of the magnetic field equations. First stationary and time-harmonic magnetic field models are introduced. Afterwards models are treated in which the time-harmonic magnetic field is coupled with an external electrical circuit. The discrete variant of these models is solved using algebraic multigrid methods, possibly accelerated by an outer Krylov iteration. Algebraic multigrid (AMG) methods allow to obtain the mesh independent convergence characteristic for multigrid on models with a complicated geometry. This thesis is based on previously developed AMG codes. The class of problems for which these are applicable was extended.

The discretization of stationary problems without anti-periodic boundary conditions results in systems that belong to the class of problems for

which AMG was originally developed. Compared with the more conventional one-level methods that were used prior to the start of this thesis, the use of AMG results in a reduction of the required simulation time. The amount of this gain increases with the problem size. In problems with anti-periodic boundary conditions, the positive off-diagonal entries need to be taken correctly into account in the construction of the interpolation if the AMG code is to converge without Krylov acceleration.

The discretization of time-harmonic problems results in systems with symmetric, complex-valued coefficient matrices. The AMG algorithm is extended to these systems by basing the selection of the coarse grid points and the construction of the interpolation operator on the real part of the matrix. This extension is such that the Galerkin coarse grid discretization yields a matrix with similar structure and properties then its fine grid equivalent.

The discretization of field-circuit coupled problems results in two-by-two block structured systems. The first and second diagonal block represent the discretized field equations and the electrical circuit respectively. These two blocks are coupled by the magnetically induced current and voltages in the electrical conductors of the system. In the AMG algorithm this structure is exploited by basing the selection of the coarse grid points and the construction of the interpolation on the discretized field equations. The electrical circuit and the coupling terms are taken into account in the solve phase of the algorithm. For the implementation we developed an interface that allows to call AMG from within a software library for discretized differential equations. This library was in turn coupled with the finite element package considered in this thesis. The coupling between these three software components has proven to be efficient and robust in practical applications. In the computation of an induction machine for example the use of the generalization of AMG for field-circuit coupled problems results in an acceleration with a factor of 24 compared with previously implemented solvers.

Algebraic Multigrid for Two-Dimensional Time-Harmonic Magnetic Field Computations

Domenico Lahaye
Departement Computerwetenschappen, K.U.Leuven
Celestijnenlaan 200A, B-3001 Heverlee, België

Samenvatting

De eindige-elementensimulatie van elektrische energieomzetters zoals machines en transformatoren vereist het oplossen van algebraïsche stelsels vergelijkingen met een groot aantal onbekenden. In praktische berekeningen neemt deze procedure tot 90 procent van de totale simulatietijd in beslag. Deze thesis levert bijdragen in de ontwikkeling van efficiënte iteratieve methoden voor de oplossing van de eindige-elementenstelsels. De voorgestelde algoritmen werden geïmplementeerd in een simulatiepakket teneinde ze te kunnen aanwenden in de berekening van industriële modellen.

De modellen beschouwd in deze thesis zijn gebouwd op basis van tweedimensionale quasi-stationaire benaderingen van de magnetische veldvergelijkingen. Eerst worden stationaire en tijdsharmonische magnetische veldmodellen geïntroduceerd. Daarna komen komen problemen aan bod waarin tijdsharmonische magnetische veldmodellen gekoppeld worden met een uitwendig elektrisch netwerk. De discrete variant van deze modellen wordt opgelost aan de hand van de algebraïsche multiroostermethoden (AMG), mogelijk versneld door een uitwendige Krylov-deelruimte-iteratie. AMG-methoden laten toe om de roosteronafhankelijke convergentie, karakteristiek voor multiroostermethoden, te bekomen in problemen met ingewikkelde geometrieën. Deze thesis steunt op de reeds eerder ontwikkelde AMG-codes. De klasse van problemen waarvoor deze codes toepasbaar zijn werd uitgebreid.

De discretisatie van stationaire problemen zonder anti-periodieke randvoorwaarden resulteert in stelsels die tot de klasse van problemen behoren waarvoor de AMG methode oorspronkelijk ontwikkeld werd. Vergeleken met de meer conventionele één-roostermethoden die vóór de aanvang van deze thesis werden gebruikt, leidt het gebruik van AMG tot een reductie van de vereiste simulatietijd. De grootte van de snelheidswinst neemt toe met de probleemgrootte. In problemen met anti-periodieke randvoorwaarden dienen de positieve niet-diagonaalelementen in de fijnroostermatrix correct in rekening te worden gebracht in de constructie van de interpolatie opdat het AMG-algoritme zonder Krylov-versnelling zou convergeren.

De discretisatie van tijdsharmonische problemen resulteert in stelsels met symmetrische matrices van complexe coëfficiënten. Het AMG-algoritme wordt naar deze stelsels uitgebreid door de selectie van de grofroosterpunten en de constructie van de interpolatie te baseren op het reëel van de matrix. Deze uitbreiding is danig dat de Galerkin-grof-roosterdiscretisatie een matrix geeft met dezelfde structuur en eigenschappen als zijn equivalent op het fijne rooster. Een numerieke validatie van dit algoritme toont een versnelling van de simulaties die toeneemt met de probleemgrootte net zoals bij stationaire problemen.

De discretisatie van tijdsharmonische veld-circuit gekoppelde problemen resulteert in twee-bij-twee blok-gestructureerde systemen. Het eerste en tweede diagonaalblok stellen respectievelijk de gediscretiseerde veldvergelijkingen en het elektrisch circuit voor. Deze twee blokken worden gekoppeld door de geïnduceerde stromen en spanningen in de elektrische geleiders van het model. In het AMG-algoritme wordt deze blokstructuur uitgebuit door de selectie van de grofroosterpunten en de constructie van de interpolatie te baseren op de gediscretiseerde veldvergelijkingen. Het elektrisch circuit en de koppelingstermen worden in rekening gebracht in de oplossingsfase van het algoritme. Voor de implementatie hebben we een interface ontwikkeld die toelaat om de AMG-codes aan te roepen vanuit een softwarebibliotheek voor het oplossen van gediscretiseerde differentiaalvergelijkingen. Deze bibliotheek werd op zijn beurt gekoppeld met het eindige-elementenpakket dat beschouwd wordt in deze thesis. De koppeling tussen deze drie softwarecomponenten heeft bewezen efficiënt en stabiel te zijn in praktische berekeningen. In de simulatie van een inductiemotor bijvoorbeeld levert de veralgemening van AMG voor veld-circuit gekoppelde problemen een versnelling op met een factor 24 in vergelijking met voorheen geïmplementeerde solvers.

Opgedragen aan Timothy en Thibault

Preface

After five years of research, I would like to thank several people who contributed to the shaping of this thesis.

I would like to express my gratitude to Prof. R. Belmans and Prof. K. Hameyer of the ELEN/ESAT group and to Prof. S. Vandewalle and Prof. D. Roose of the CW/SciComp group for assuring the funding and the scientific support that made this work possible. I would like to mention several colleagues with whom I had the pleasure to work with. In the ESAT/ELEN group, Ronny Mertens provided the first functionality in `Olympos` that made our collaboration possible, introduced me to the other members of the ESAT/ELEN group and realized the first coupling between `AMG1R5` and `Olympos`. In this way, he laid the foundations of much of the future work. Uwe Pahner was responsible for putting the newly integrated linear solvers in `Olympos` on fully fledged industrial test benches in less than twelve hours after the code was made available. The collaboration with Herbert De Gersem especially was one in which our combined effort exceeded by far what the two of us could have done individually. Much of this thesis is the result of numerous discussions with him. In the CW/SciComp group I quickly lost track of the number of questions regarding the theory and implementation of iterative solvers I asked Serge Goossens. He did never loose his enthusiasm in looking for the right answers. A lot of help on software related issues was provided by Geert Uytterhoeven and Jo Simoens. Other colleagues and former colleagues in both research groups contributed to the pleasant atmosphere to work in. Other members of the research communities I was envolved in contributed to this work through informal discussions during conferences or by e-mail.

I owe special thanks to Arnold Krechel, Dr. Klaus Stüben, Dr. Barry Smith and the `PETSc` team for providing valuable support in the software development of this work and for hosting me kindly at their instutitutes. I would like to thank my supervisors Prof. S. Vandewalle and Prof. K. Hameyer and other members of the jury for many remarks that allowed me to improve this text, and Prof. A. Aernoudt, Dr. Klaus Stüben, Prof. P. De Groen for accepting to be members of the jury.

This work finally would not have been possible without the loving care of my family and the encouragements of friends.

Domenico Lahaye
December 2001

Acknowledgement

The research in this thesis is financially supported by the Research Council of the Katholieke Universiteit Leuven (OT/94/16) and the Fund for Scientific Research Flanders (Fonds voor Wetenschappelijk Onderzoek - Vlaanderen), project 61.0427 and project G.0427.98.

Travel support by the Fund for Scientific Research Flanders is gratefully acknowledged.

Nederlandse samenvatting

De algebraïsche multiroostermethode voor tweedimensionale tijdsharmonische magnetische veldberekeningen

Inhoudsopgave

1 Inleiding

1.1 Context van het onderzoek

In het ontwerp van elektromagnetische toestellen wordt tegenwoordig het experimentele werk vaak aangevuld met computersimulaties. De doelstelling op lange termijn bij de ontwikkeling van de hiervoor vereiste software is het overbodig maken van het bouwen van dure prototypes.

Deze thesis levert verschillende bijdragen tot de verbetering van een pakket voor de numerieke simulatie van quasi-stationaire elektromagnetische energie-omzetters door het incorporeren van moderne technologie voor het oplossen van stelsels van lineaire vergelijkingen. Dit interdisciplinair onderzoek kadert in de vruchtbare samenwerking tussen twee vakgroepen van de Faculteit Toegepaste Wetenschappen van de K.U.Leuven. Deze groepen zijn de vakgroep Elektrische Energie van het Departement Elektrotechniek (ESAT/ELEN) enerzijds en de groep Technisch Wetenschappelijk Rekenen van het Departement Computerwetenschappen (CS/SciComp) anderzijds. De eerste groep heeft expertise in de studie van elektromagnetische toestellen zoals elektrische motoren en transformatoren [30, 77, 56, 55]. De tweede groep heeft expertise in het ontwerp, de analyse en de implementatie van algoritmen voor de snelle oplossing van gediscretiseerde partiële differentiaalvergelijkingen (PDEs) [60, 47, 71].

1.2 Het `Olympos`-pakket

Het onderzoek in computermodellen voor elektromagnetische energie-omzetters in de ESAT/ELEN-onderzoeksgroep is sinds 1995 geconcentreerd rond de ontwikkeling van het `Olympos`-pakket [86, 76, 35, 27]. Vanwege het belang van dit pakket in deze thesis, zullen we een korte beschrijving ervan geven.

In `Olympos` wordt een beperkt aantal van de Maxwell-vergelijkingen opgelost voor het magnetisch veld. Voor het type van toepassingen die van belang zijn voor de ESAT/ELEN-groep, is het voldoende om quasi-stationaire benaderingen te beschouwen. De verplaatsingsstroom kan met andere woorden verwaarloosd worden. Het pakket laat toe om zowel stationaire, tijdsharmonische als transitorische magnetische velden te simuleren. Uit het berekend magnetisch veld kunnen praktisch relevante grootheden zoals krachten en koppels afgeleid worden.

De complexe geometrie van praktische toestellen wordt in eerste stap benaderd door het modelleren van tweedimensionale dwarsdoorsneden of van axi-symmetrische configuraties. De magnetische vectorpotentiaal wordt ingevoerd als onbekende. In *stationaire* formuleringen bestaat het continue model uit een diffusievergelijking voor de component van de vectorpotentiaal loodrecht op het beschouwde domein. In *tijdsharmonische* formuleringen wordt de variatie van de amplitude van de vectorpotentiaal in de ruimte beschreven door de Helmholtz-vergelijking met complexe verschuiving. In praktische toepassingen is de berekening van het magnetisch veld onmogelijk zonder de eigenschappen en de interconnectie van de elektrisch geleidende gebieden in het rekendomein in rekening te brengen. In deze gevallen dient de differentiaalvergelijking voor het magnetisch veld aangevuld te worden met bijkomende vergelijkingen die het elektrisch circuit modelleren. Dit resulteert in zogenoemde *veld-circuit* gekoppelde formuleringen.

In `Olympos` wordt het continue model gediscretiseerd met lineaire eindige elementen. Praktische ervaring met het pakket heeft uitgewezen dat het oplossen van de eindige-elementenstelsels de flessenhals vormt in berekeningen. Deze flessenhals is danig dat simulaties waarin stelsels van hoge-resolutiediscretisaties herhaaldelijk dienen opgelost te worden bijzonder tijdrovend zijn. Het doel van deze thesis bestaat erin deze flessenhals weg te werken door gebruik te maken van numerieke algoritmen die snel zijn in CPU-tijd en die beperkte geheugenvereisten hebben. Het algoritme centraal in dit werk is de *algebraïsche multiroosterm*ethode. In wat volgt geven we een korte inleiding op deze methode met als doel de nodige terminologie te introduceren.

1.3 De algebraïsche multiroostermethode

De eindige-elementendiscretisatie van PDEs resulteert in lineaire stelsels met ijle matrices van coëfficiënten. Dit motiveert de keuze voor iteratieve oplossingstechnieken. De klassieke iteratieve methoden gebaseerd op een splitsing van de matrix zijn conceptueel eenvoudig, maar convergeren onaanvaardbaar traag voor problemen met een groot aantal onbekenden. In multiroosterverbeteringen van deze methoden wordt de convergentie op een gegeven fijn rooster versneld door berekeningen op een hiërarchie van grovere roosters.

De klassieke multiroostermethoden [111] vormen snelle iteratieve solvers voor elliptische modelproblemen en werden succesvol toegepast in verschillende onderzoeksdisciplines. Hun implementatie is echter lastig voor problemen met ingewikkelde geometrieën. Als een alternatief voor de klassieke multiroostermethoden werden daarom algebraïsche multiroostermethoden (AMG) ontwikkeld.

In AMG-methoden wordt de hiërarchie van grof-roosterdiscretisaties automatisch geconstrueerd. Deze methoden opereren enkel op de fijn-roostermatrix. Geen enkele geometrische informatie over het gediscretiseerde probleem is vereist. Dit maakt AMG bijzonder aantrekkelijk in gebruik. In deze thesis maken we gebruik van twee AMG-codes. De eerste werd ontwikkeld door Ruge en Stüben [95] en zullen we aanduiden met `AMG1R5`. De tweede werd later ontwikkeld door Stüben [109] en zullen we aanduiden met `SAMG`.

1.4 Bijdragen van de thesis

Dit werk heeft bijgedragen tot algoritmische ontwikkelingen voor het efficiënt oplossen van het stationaire, het tijdsharmonische alsook het tijdsharmonisch veld-circuit gekoppelde probleem. De voorgestelde algoritmen

werden geïncorporeerd in `Olympos` om ze op die manier te kunnen aanwenden in het oplossen van industrieel relevante problemen.

De discretisatie van *stationaire* problemen leidt tot symmetrisch positief definiete M-matrices. Het is voor deze klasse van matrices dat `AMG1R5` en `SAMG` oorspronkelijk ontwikkeld werden. Aan de hand van numerieke voorbeelden tonen we dat het gebruik van AMG als solver leidt tot een substantiële versnelling van simulaties vergeleken met één-rooster gepreconditioneerde Krylov-deelruimtemethoden die voorheen in `Olympos` werden gebruikt. De grootte van de snelheidswinst neemt toe met de probleemgrootte. We tonen ook dat het gebruik van AMG als preconditioner verder de snelheid en de robuustheid van de code verhoogt.

De discretisatie van *anti-periodieke* randvoorwaarden introduceert grote positieve niet-diagonaalelementen in de matrix. We hebben ontdekt dat `AMG1R5` deze positieve elementen niet correct behandelt, en dat dit de convergentie van de code verhindert als deze gebruikt wordt zonder Krylov-deelruimteversnelling. Dit probleem werd opgelost in `SAMG` door de positieve niet-diagonaalelementen correct in rekening te brengen in de constructie van de interpolatie.

De discretisatie van *tijdsharmonische* problemen leidt tot complex symmetrische stelsels. Een gepaste Krylov-deelruimtemethode voor deze stelsels is de Symmetrische Quasi-Minimum-Residu-methode [39]. In problemen *zonder circuitrelaties* wordt een algebraïsche multiroostermethode geconstrueerd door de selectie van de grof-roosterpunten en de constructie van de interpolatie te baseren op het reëel deel van de coëfficiëntenmatrix.

In problemen waarin *circuitrelaties* aanwezig zijn, leidt de discretisatie tot 2×2 blok-gestructureerde matrices. We ontwikkelden een algoritme dat deze blokstructuur uitbuit en dat toelaat AMG te hergebruiken voor het oplossen van veld-circuit gekoppelde problemen.

Voor de implementatie van het veralgemeend AMG-algoritme voor het gekoppelde probleem hebben we een interface op punt gesteld die toelaat de AMG-codes op te roepen vanuit de Portable, Extensible Toolkit for Scientific Computations (`PETSc`) [5]. Dit pakket is een algemene softwarebibliotheek voor het oplossen van gediscretiseerde PDEs. De interface tussen AMG en `PETSc` werd geïncorporeerd in `Olympos`. De resulterende solver is danig dat `Olympos` de discretisatie en assemblage van het lineaire stelsel voor zijn rekening neemt. AMG construeert de hiërarchie van grof-roosterdiscretisaties en `PETSc` is verantwoordelijk voor de multiroosteriteratie versneld door Krylov-deelruimtesolvers. Bij het berekenen van modellen met praktische relevantie is gebleken dat de koppeling tussen deze drie pakketten efficiënt en robuust is. Vergeleken met één-rooster gepreconditioneerde Krylov-deelruimtesolvers levert het nieuwe algoritme in de berekening van bijvoorbeeld een inductiemotor een versnelling met een factor tussen 5 en 24 afhankelijk van de probleemgrootte.

1.5 Overzicht van de thesis

Deze thesis bestaat uit 9 hoofdstukken. In Hoofdstuk 2 worden continue modellen voor magnetische veldberekingen in elektromagnetische toestellen afgeleid, vertrekkende van de Maxwell-vergelijkingen. In Hoofdstuk 3 worden deze modellen gediscretiseerd, gebruik makend van de eindige-elementenmethode, en de eigenschappen van de resulterende stelsels beschreven. In Hoofdstuk 4 wordt een overzicht van Krylov-deelruimtemethoden gegeven, en de convergentie-eigenschappen van deze methoden worden besproken. In Hoofdstuk 5 worden de fundamentele concepten van multiroostermethoden in het algemeen en geometrische multiroostermethoden in het bijzonder geïntrodu-ceerd. In een tijdsharmonisch modelprobleem wordt de performantie van een geometrische multiroosteralgoritme geanalyseerd. In Hoofdstuk 6 worden algebraïsche multiroostermethoden in detail besproken. De performantie van de originele Ruge-Stüben-`AMG1R5` en zijn opvolger `SAMG` worden gevalideerd aan de hand van een verzameling van testproblemen. Zowel stationaire als tijdsharmonische problemen worden beschouwd. In hoofdstuk 7 worden verschillende algoritmen voor veld-circuit gekoppelde problemen voorgesteld en met elkaar vergeleken. Numerieke resultaten tonen dat het veralgemeend AMG-algoritme snel is zowel in aantal iteraties en als in CPU-tijd. Hoofdstuk 8 is gewijd aan de beschrijving van de interface tussen de AMG-codes en `PETSc`. In Hoofdstuk 9 tenslotte wordt deze thesis afgesloten met enkele besluiten.

2 Magnetische veld modellen

2.1 Quasi-stationaire magnetische velden

In deze thesis worden *quasi-stationaire* benaderingen voor het magnetisch veld beschouwd. In deze benaderingen gaat men ervan uit dat de verplaatsingsstroom verwaarloosbaar klein is ten opzichte van de geleidingsstroom. Gegeven deze veronderstelling, kan men de wet van Ampère in functie van de magnetische vectorpotentiaal $\mathbf{A}$ schrijven als

$$\nabla \times (\nu \nabla \times \mathbf{A}) = \mathbf{J}\,, \tag{1}$$

met ν en $\mathbf{J}$ respectievelijk de magnetische reluctiviteit en de stroomdichtheidsvector. Door integratie van de wet van Faraday, bekomt men de volgende uitdrukking voor het elektrisch veld $\mathbf{E}$

$$\mathbf{E} = -\nabla\phi - \frac{\partial \mathbf{A}}{\partial t}\,, \tag{2}$$

met ϕ de elektrische potentiaal. Met behulp van deze uitdrukking en van de wet van Ohm

$$\mathbf{J} = \sigma\,\mathbf{E}\,, \tag{3}$$

met σ de elektrische geleidbaarheid, kan de vergelijking (1) geschreven worden als

$$\nabla \times (\nu\,\nabla \times \mathbf{A}) + \sigma\,\frac{\partial \mathbf{A}}{\partial t} = -\sigma\,\nabla\phi\,. \tag{4}$$

Voor *tweedimensionale* geometrieën kiest met het cartesisch assenstelsel dan nig dat het rekendomein in het xy-vlak ligt. De vectorpotentiaal heeft slechts één component loodrecht op het rekendomein, en deze component is enkel van x en y afhankelijk

$$\mathbf{A} = (0, 0, A_z(x, y, t))\ . \tag{5}$$

De vergelijking (4) herleidt zich voor 2D-geometrieën dan tot

$$\mathcal{L}(A_z) + \sigma\frac{\partial \mathbf{A}_z}{\partial t} = -\sigma\frac{\partial \phi}{\partial z}\,, \tag{6}$$

waarbij we de volgende notatie hebben ingevoerd

$$\mathcal{L}(A_z) = -\frac{\partial}{\partial x}\left(\nu\,\frac{\partial A_z}{\partial x}\right) - \frac{\partial}{\partial y}\left(\nu\,\frac{\partial A_z}{\partial y}\right)\ . \tag{7}$$

Voor *stationaire* magnetische velden is de beschrijvende vergelijking voor de vectorpotentiaal dan

$$\mathcal{L}(A_z) = -\sigma\frac{\partial \phi}{\partial z}\,. \tag{8}$$

In *tijdsharmonische* formuleringen veronderstelt men dat de elektromagnetische grootheden sinusoïdaal variëren in de tijd met een pulsatie ω. Voor een gegeven elektromagnetische grootheid $F(x, y, t)$ kan men dan schrijven dat

$$F(x, y, t) = \Re[\,\widehat{F}(x, y)\exp(j\omega t)\,]\,, \tag{9}$$

waarbij $\Re$ het reëel deel aanduidt en $\widehat{F}$ de amplitude. Voor deze formuleringen herleidt vergelijking (4) zich tot de volgende vergelijking voor $\widehat{A}_z$

$$\mathcal{L}(\widehat{A}_z) + j\,\omega\,\sigma\,\widehat{A}_z = -\sigma\frac{\partial \widehat{\phi}}{\partial z}\,. \tag{10}$$

2.2 Modellen van elektrische geleiders

Voor de beschrijving van de elektromagnetische wisselwerking worden in deze thesis, behalve lokale veldgrootheden, ook geïntegreerde globale elektrische grootheden ingevoerd. Hiertoe beschouwen we een elektrische geleider met lengte ℓ_z en dwarsdoorsnede Ω. Indien deze geleider een stationaire stroom voert, zijn de spanningsval $\triangle V$ en de stroom I gerelateerd door de wet van Ohm in integraalvorm

$$\triangle V = R\, I\,, \tag{11}$$

met R de ohmse weerstand van de geleider. De grootheid $G = 1/R$ noemt men de admittantie. De constitutieve relatie (11) dient veralgemeend te worden naar zowel tijdsafhankelijke stromen als naar situaties waarin geleiders met elkaar verbonden zijn in een circuit.

In de veralgemening van de wet van Ohm naar tijdsafhankelijke stromen, speelt de verhouding tussen de indringdiepte δ en de straal van de geleider een rol. Naargelang deze verhouding groot of klein is, onderscheidt men respectievelijk *volle* (solid) en *gewikkelde* (stranded) geleiders. Indices *sol* en *str* zullen worden ingevoerd om de symbolen verbonden met de volle en gewikkelde geleiders aan te duiden.

Voor een volle geleider veronderstelt men de spanningsval constant over de dwarsdoorsnede Ω_{sol}. Door het integreren van

$$\widehat{J}_z = \frac{\sigma}{\ell_z}\,\widehat{\triangle V} - j\,\omega\,\sigma\,\widehat{A}_z \tag{12}$$

over Ω_{sol}, bekomt men de volgende constitutieve relatie

$$\widehat{I}_{sol} = G_{sol}\,\widehat{\triangle V}_{sol} - j\,\omega \int_{\Omega_{sol}} \sigma\,\widehat{A}_z\,d\Omega\,. \tag{13}$$

De tweede term in het rechterlid van deze relatie is de magnetisch geïnduceerde stroom.

De dwarsdoorsnede van een gewikkelde geleider bestaat uit de vereniging van dwarsdoorsneden van verschillende geleiders die elk een te kleine straal hebben om apart gemodelleerd te worden. Over de individuele dwarsdoorsneden wordt de stroomdichtheid constant verondersteld. Stelt men het aantal in serie geschakelde individuele geleiders gelijk aan N_t, dan kan men de spanningsval $\triangle V_{str}$ over de gewikkelde geleider schrijven als N_t keer de gemiddelde spanningsval $\triangle V_{av}$ over Ω_{str}. Men bekomt zo als constitutieve relatie

$$\widehat{\triangle V}_{str} = R_{str}\,\widehat{I}_{str} + j\,\omega\,\frac{N_t\,\ell_z}{S_{str}} \int_{\Omega_{str}} \widehat{A}_z\,d\Omega\,. \tag{14}$$

De tweede term in het rechterlid van deze vergelijking is de magnetisch geïnduceerde spanning.

2.3 Stationaire elektrische circuits

De beschrijvende vergelijkingen voor een elektrisch circuit zijn

- de Kirchhoff spanningswet (KVL)
- de Kirchhoff stroomwet (KCL)
- de stroom-spanningrelatie voor elke component in het circuit.

Om een maximaal lineair onafhankelijk stel van deze vergelijkingen te bekomen, maakt deze thesis gebruik van een topologische methode. Deze methode associeert een graaf met het circuit. In deze graaf wordt een boom T getraceerd. Hierbij worden prioriteitsregels in acht genomen die danig zijn dat spanningsbronnen en volle geleiders in de boom verschijnen, en stroombronnen en gewikkelde geleiders in het complement L ervan. De takken van de boom worden opgesplitst in spanningsbronnen (met index ν), volle geleiders en overige takken in de boom T_0

$$T = \{sol\} \cup \{\nu\} \cup T_0 \,. \tag{15}$$

Analoog worden takken in het complement opgesplitst in stroombronnen (met index i), gewikkelde geleiders en de overige takken L_0

$$L = \{str\} \cup \{i\} \cup L_0 \,. \tag{16}$$

Weerstanden kunnen zowel in de boom als in het complement voorkomen. Gegeven de boom, kan men een fundamentele kringloopmatrix B_f en een fundamentele doorsnedematrix D_f voor het circuit opstellen. Met behulp van deze matrices en de vectoren van onbekende spanningen $\triangle V$ en stromen I kan met een maximaal lineair onafhankelijk stel van KVLs en KCLs respectievelijk schrijven als

$$B_f \ \triangle V = 0 \,, \tag{17}$$

en

$$D_f \, I = 0 \,. \tag{18}$$

Om dit paar stelsels van vergelijkingen samen te smelten tot een vierkant stelsel, worden de rijen en kolommen van de matrices en vectoren verdeeld volgens (15) en (16), en worden de spannings- en stroombronnen naar het rechterlid gebracht. Zo bekomt men de volgende vergelijkingen voor een stationair circuit

$$S \begin{pmatrix} I_{str} \\ I_{L_0} \\ \triangle V_{sol} \\ \triangle V_{T_0} \end{pmatrix} = \begin{pmatrix} B_{str,\,\nu} \ \triangle V_\nu \\ B_{L_0,\,\nu} \ \triangle V_\nu \\ -D_{sol,\,i} \, I_i \\ -D_{T_0,\,i} \, I_i \end{pmatrix} , \tag{19}$$

waarbij de matrix S gelijk is aan

$$S = \begin{pmatrix} -\mathrm{diag}[R_{str}] & 0 & -B_{str,\,sol} & -B_{str,\,T_0} \\ 0 & -\mathrm{diag}[R_{L_0}] & -B_{L_0,\,sol} & -B_{L_0,\,T_0} \\ D_{sol,\,str} & D_{sol,\,L_0} & \mathrm{diag}[G_{sol}] & 0 \\ D_{T_0,\,str} & D_{T_0,\,L_0} & 0 & \mathrm{diag}[G_{T_0}] \end{pmatrix} . \qquad (20)$$

Ze is symmetrisch en niet-singulier door constructie.

2.4 Veld-circuit koppeling

Voor de formulering van het veld-circuit gekoppelde probleem, veronderstelt men dat het rekendomein bestaat uit een elektrisch geleidend en niet-geleidend deel. Het geleidende deel wordt verder opgedeeld in de unie van de dwarsdoorsneden van de volle $\Omega_{sol,\,q}$ en gewonden geleiders en $\Omega_{str,\,p}$. Als het niet-geleidende deel aangeduid wordt met Ω_{core}, kan men stellen dat

$$\Omega = \left(\bigcup_p \Omega_{str,\,p} \right) \cup \left(\bigcup_q \Omega_{sol,\,q} \right) \cup \Omega_{core} \, . \qquad (21)$$

Het magnetisch veldprobleem op Ω kan dan worden geformuleerd als

$$\begin{aligned} \mathcal{L}(\widehat{A}_z) + j\,\omega\,\sigma\,\widehat{A}_z &= \frac{\sigma}{\ell_z}\,\widehat{\Delta V}_{sol,\,q} && \text{op } \Omega_{sol,\,q} \\ \mathcal{L}(\widehat{A}_z) &= \frac{N_{t,\,p}}{S_{str,\,p}}\,\widehat{I}_{str,\,p} && \text{op } \Omega_{str,\,p} \\ \mathcal{L}(\widehat{A}_z) &= 0 && \text{op } \Omega_{core} \, . \end{aligned} \qquad (22)$$

Indien alle spanningsvallen over de volle geleiders en alle stromen over de gewikkelde geleiders gekend zijn, dan kan dit differentiaalprobleem opgelost worden. In het algemeen echter dient het differentiaalprobleem aangevuld te worden met elektrische circuitvergelijkingen. Voor tijdsharmonische stromen bekomt men deze vergelijkingen door de magnetisch geïnduceerde stromen en spanningen op te tellen bij het linkerlid van (19)

$$S \begin{pmatrix} \widehat{I}_{str} \\ \widehat{I}_{L_0} \\ \widehat{\Delta V}_{sol} \\ \widehat{\Delta V}_{T_0} \end{pmatrix} + \begin{pmatrix} -\widehat{\Delta V}_{ind} \\ 0 \\ \widehat{I}_{ind} \\ 0 \end{pmatrix} = \begin{pmatrix} B_{str,\,\nu}\,\widehat{\Delta V}_{\nu} \\ B_{L_0,\,\nu}\,\widehat{\Delta V}_{\nu} \\ -D_{sol,\,i}\,\widehat{I}_i \\ -D_{T_0,\,i}\,\widehat{I}_i \end{pmatrix} . \qquad (23)$$

In veld-circuit gekoppelde problemen worden het differentiaalprobleem (22) en het algebraïsch stelsel (23) simultaan opgelost voor de vectorpotentiaal, de spanning over de volle en de stroom door de gewikkelde geleiders.

3 Eindige-elementendiscretisatie

3.1 Variationele formulering

De eindige-elementendiscretisatie van de PDEs afgeleid in het vorige hoofdstuk vertrekt van een variationele formulering. In deze formulering is het oorspronkelijk differentiaalprobleem, i.e., de differentiaalvergelijking aangevuld met randvoorwaarden, equivalent met het oplossen van een probleem van de vorm

$$\text{vind } u \in V \text{ zó dat } \forall v \in V \text{ geldt dat } \mathtt{A}(u,\, v) = \mathtt{F}(v)\,. \tag{24}$$

Hierbij is V voor de stationaire vergelijking (8) een reële vectorruimte en zijn $\mathtt{A}$ en $\mathtt{F}$ respectievelijk een bilineaire en lineaire vorm op V, als volgt gedefinieerd

$$\mathtt{A}(w,\, v) = \int_\Omega \nu\, \nabla w \cdot \nabla v\, d\Omega \tag{25}$$

en

$$\mathtt{F}(v) = -\int_\Omega \sigma\, \frac{\partial \phi}{\partial z}\, v\, d\Omega\,. \tag{26}$$

Voor de tijdsharmonische vergelijking (10) is V een complexe functieruimte en zijn $\mathtt{A}$ en $\mathtt{F}$ respectievelijk een sesquilineaire en lineaire vorm als volgt gedefinieerd

$$\mathtt{A}(w,\, v) = \int_\Omega [\nu\, \nabla w \cdot \nabla v + j\, \omega\, \sigma\, w\, v]\, d\Omega \tag{27}$$

en

$$\mathtt{F}(v) = -\int_\Omega \sigma\, \frac{\partial \widehat{\phi}}{\partial z}\, v\, d\Omega\,. \tag{28}$$

Voor de differentiaalvergelijking (22) dienen deze definities vervangen te worden door

$$\mathtt{A}(w,\, v) = \int_\Omega [\nu\, \nabla w \cdot \nabla v + \Lambda_e\, w\, v]\, d\Omega \tag{29}$$

en

$$\mathtt{F}(v) = \int_\Omega \Upsilon_s\, v\, d\Omega\,, \tag{30}$$

waarbij Λ_e en Υ_s respectievelijk de geïnduceerde en opgelegde stroomfuncties zijn. Het bestaan en de uniciteit van de oplossing van deze variationele formuleringen wordt aangetoond aan de hand van de stelling van Lax-Millgram [90].

3.2 Discretisatie

In deze thesis worden de oplossingen van de differentiaalvergelijkingen benaderd door lineaire veeltermen, stuksgewijs gedefinieerd over driehoeken. Hiertoe wordt een ongestructureerd rooster dat het rekendomein bedekt geconstrueerd. Een voorbeeld van zo'n rooster voor een permanent-magneetmotor wordt getoond in Figuur 1. Een typische maat voor de maaswijdte noemt men h. Men stelt X^h gelijk aan de verzameling eindige-elementenbasisfuncties en ontwikkelt de discrete benadering in deze basis

$$u_h = \sum_k c_k \, \varphi_k \, . \tag{31}$$

De discrete variationele formulering

$$\text{vind } u_h \in X^h \text{ zó dat } \forall v_h \in X^h \text{ geldt dat } \mathtt{A}(u_h, \, v_h) = \mathtt{F}(v_h) \tag{32}$$

vertaalt zich dan naar een stelsel van lineaire vergelijkingen voor de coëfficiënten c_k in (31)

$$A \, c = f \, . \tag{33}$$

De uitdrukkingen voor de elementen van de matrix $A_{k\ell}$ en het rechterlid f_k bekomt men door de bilineaire (of sesquilineaire) en lineaire vorm van de variationele formulering te evalueren in de basisfuncties als volgt

$$A_{k\ell} = \mathtt{A}(\varphi_k, \, \varphi_\ell) \tag{34}$$

en

$$f_k = \mathtt{F}(\varphi_k) \, . \tag{35}$$

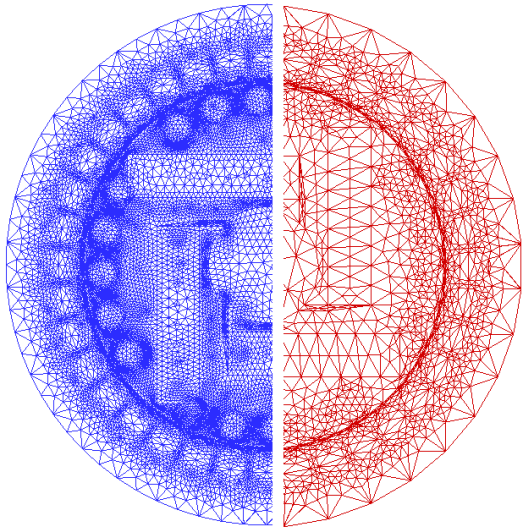
In tijdsharmonische problemen met circuitvergelijkingen dient het stelsel (33) aangevuld te worden met gediscretiseerde circuitvergelijkingen. Deze laatste bekomt men door het substitueren van de benadering (31) in de uitdrukking van de geïnduceerde stroom en spanning in (23). Het discreet veld-circuit gekoppelde systeem kan men dan schrijven als

$$\mathcal{A} \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} 0 \\ \chi \, g \end{pmatrix} \, , \tag{36}$$

waarbij we de matrix $\mathcal{A}$ hebben ingevoerd

$$\mathcal{A} = \begin{pmatrix} A & -F \\ -F^T & \chi S \end{pmatrix} \, , \tag{37}$$

waarbij de deelmatrices $\chi \, S$ en F en de deelvectoren d en χg respectievelijk het elektrisch circuit, de koppelingstermen en de onbekende en opgelegde stromen en spanningen voorstellen.



Figuur 1: Grof en fijn rooster voor een permanent-magneetmotor.

3.3 Eigenschappen van de coëfficiëntenmatrix

Voor de oordeelkundige keuze van een iteratieve oplossingsprocedure voor het stelsel (33) baseert men zich op de eigenschappen van de matrix A.

In stationaire problemen is de matrix A ijl symmetrisch positief definiet. Het is een M-matrix als geen anti-periodische randvoorwaarden in het differentiaalprobleem aanwezig zijn. De verwerking van anti-periodische randvoorwaarden leidt tot positieve buitendiagonaalelementen van dezelfde orde van grootte dan de diagonaalelementen. In tijdsharmonische problemen is de coëfficiëntenmatrix complex symmetrisch. In problemen zonder circuitrelaties heeft het reëel deel van de matrix de eigenschappen van een stationaire matrix. Het spectrum van de matrix ligt in het eerste kwadrant van het complexe vlak. In problemen met circuitrelaties introduceert de koppeling van de gediscretiseerde PDE met het elektrisch circuit eigenwaarden met mogelijk een negatief reëel of imaginair deel.

4 Matrixsplitsings- en Krylov-deelruimte-methoden

4.1 Matrixsplitsingsmethoden

De meest klassieke iteratieve methoden voor het oplossen van het stelsel (33) zoals de Jacobi-, Gauss-Seidel- en (Symmetrische-) Successieve-Overrelaxatie-methoden, zijn gebaseerd op een splitsing van de matrix A in de vorm

$$A = M - N\,. \tag{38}$$

Deze splitsing geeft aanleiding tot het iteratieve schema

$$x_{m+1} = M^{-1}\,N\,x_m + M^{-1}\,b\,. \tag{39}$$

Deze klassieke schema's zijn vaak niet efficiënt. Ze doen dienst als eenvoudige preconditioners voor Krylov-deelruimtemethoden en als bouwstenen voor meer geavanceerde multiroostermethoden.

4.2 Krylov-deelruimtemethoden

Onder Krylov-deelruimtemethoden classificeert men een familie van niet-stationaire iteratieve methoden die, gegeven een startoplossing x_0 en het hiermee overeenkomstig initieel residu r_0, iteranden x_m bepalen danig dat

$$x_m \in x_0 + K^m(A, r_0)\,. \tag{40}$$

Hierbij is $K^m(A, r_0)$ de Krylov-deelruimte van dimensie m gedefinieerd als

$$K^m(A, v) = \mathrm{span}\,\{v, A\,v, \ldots, A^{m-1}\,v\}\,. \tag{41}$$

Gegeven de matrix V_m waarvan de kolommen een basis vormen voor de ruimte $K^m(A, r_0)$ en een vector van coëfficiënten y_m kan de voorwaarde (41) ook geschreven worden als

$$x_m = x_0 + V_m\,y_m\,. \tag{42}$$

Krylov-deelruimtemethoden verschillen in de manier waarop ze de basis V_m construeren en de coëfficiënten y_m bepalen. In de methode van Arnoldi wordt een gegeven basis van $K^m(A, r_0)$ uitgebreid tot een basis van $K^{m+1}(A, r_0)$ door het berekenen van de vector $A\,V_m(:, m)$ en de resulterende vector te orthogonaliseren ten opzichte van V_m. Deze orthogonalisatie leidt tot lange recursiebetrekkingen die rekenintensief zijn. In de bi-Lanczos-methode worden deze lange recursiebetrekkingen vermeden door $A\,V_m(:, m)$

te orthogonaliseren ten opzichte van een basis W_m van de Krylov-deelruimte $K^m(A^H, r_0)$. Deze zogenoemde bi-orthogonalisatieprocedure kan niet onmiddellijk worden verdergezet indien basisvectoren $V_m(:,m)$ en $W_m(:,m)$ worden gegenereerd die orthogonaal zijn. Voor hermitische matrices herleiden de Arnoldi- en de bi-Lanczos-methode zich tot het Lanczos-algoritme. Dit algoritme gebruikt korte recursiebetrekkingen en vereist slechts één matrix-vectorvermenigvuldiging per iteratie.

Voor het bepalen van de coëfficiënten y_m onderscheidt men algoritmen gebaseerd op het (benaderend) minimaliseren van de residunorm enerzijds en op residuprojectiemethoden anderzijds.

Voor complex symmetrische matrices A kan een basis voor $K^m(A, r_0)$ worden geconstrueerd aan de hand van het Lanczos-algoritme indien men de bilineaire vorm

$$< v_m, w_m >_T = v_m^T w_m \,. \tag{43}$$

hanteert in plaats van het standaard inwendig product. Indien men dit algoritme combineert met een residuprojectiemethode voor het bepalen van y_m, bekomt men een Krylov-deelruimtesolver waarvan de residunorm mogelijk sterke oscillaties vertoont tijdens het convergentieproces. Met deze oscillaties gaan numerieke instabiliteiten gepaard. Om deze instabiliteiten tegen te gaan, verdient het de voorkeur y_m te bepalen met een criterium waarin de residunorm benaderend geminimaliseerd wordt.

In praktische ingenieurstoepassingen dienen Krylov-deelruimtemethoden steeds te worden gecombineerd met een preconditioner om efficiënte algoritmen te bekomen. De convergentiesnelheid van Krylov-deelruimtemethoden wordt bepaald door de eigenwaardendistributie van de coëfficiëntenmatrix van het lineaire stelsel. Het idee van preconditioning bestaat erin het gegeven spectrum te transformeren in een spectrum dat gunstiger is voor de convergentie van de Krylov-iteratie. Met de methoden besproken in Sectie 4.1 als preconditioner is de convergentie van Krylov-methoden nog steeds traag. In de twee volgende hoofstukken worden efficiëntere preconditioners besproken.

5 Basis van geometrische multiroostermethoden

5.1 De geometrische multiroostermethode voor het stationaire probleem

De convergentiegeschiedenis van de basis matrixsplitsingsmethoden besproken in Sectie 4.1 vertoont typisch initieel een sterke reductie van de resi-

dunorm. Na deze eerste fase van snelle convergentie, stagneert de convergentie. Dit convergentiegedrag kan verklaard worden door het feit dat deze methoden efficiënt de hoogfrequente componenten van de fout dempen. De convergentie vertraagt indien de fout is opgebouwd uit enkel laagfrequente componenten. De fout wordt niet klein, maar wel glad (smooth). In de context van multiroostermethoden worden deze basis iteratieve methoden dan ook *smoothers* genoemd.

Gladde fouten verschijnen opnieuw als hoogfrequent indien ze worden voorgesteld op een grover rooster. Op deze vaststelling zijn multiroostermethoden gebaseerd. In deze methoden wordt namelijk de smoother aangewend op een hiërarchie van steeds grovere roosters om de foutcomponenten te dempen overeenkomstig met de frequentie.

Eerst zullen we in deze sectie de multiroostermethode voor het oplossen van het stationaire probleem formeel uitleggen. De veralgemening voor het oplossen van het tijdsharmonische probleem zal in de volgende sectie worden besproken.

In de *twee*roostermethode veronderstelt men dat de PDE (8) gediscretiseerd wordt op een fijn rooster met maaswijdte h en op een grof rooster met maaswijdte H. Deze discretisaties resulteren in respectievelijk een fijn- en grof-roosterstelsel

$$A^h x^h = b^h \tag{44}$$

en

$$A^H x^H = b^H \,. \tag{45}$$

Functies kunnen tussen het fijne en het grove rooster worden getransfereerd met behulp van de restrictie- en interpolatie-operator I_h^H en I_H^h. Gegeven een benaderende oplossing x_m^h voor de fijn-roostervergelijkingen, worden eerst de hoogfrequente componenten van de fout $e_m^h = x^h - x_m^h$ gedempt door het toepassen van enkele (typisch één of twee) iteraties van de smoother

$$x_m^h \to \overline{x}_m^h \qquad \text{met} \qquad \overline{x}_m^h = x_m^h - (Q^h)^{-1}(A^h\, x_m^h - b^h)\,, \tag{46}$$

waarbij de matrix Q^h het splitsingsschema voorstelt. Deze matrix is gelijk aan de gescaleerde diagonaal van A^h in het geval van gedempte Jacobi-smoothing en aan het benedendriehoeksdeel van A^h is het geval van Gauss-Seidel-smoothing. In termen van de fout betekent (46) dat

$$\overline{e}_m^h = S^h\, e_m^h \qquad \text{waarbij} \qquad S^h = (I^h - (M^h)^{-1} A^h)\,. \tag{47}$$

De laagfrequente componenten van de fout worden gedempt door de grof-roostercorrectie. Deze correctie bestaat uit het transfereren van het residu

na smoothing $r_m^h = b^h - A^h\,\overline{x}_m^h$ naar het grover rooster, het oplossen van de defectvergelijkingen

$$A^H\,e_m^H = I_h^H\,r_m^h\,, \tag{48}$$

op het grover rooster, het interpoleren van de grof-roostercorrectieterm naar het fijn rooster, en het optellen van deze term op het fijne rooster bij de bestaande benadering. Voor de opeenvolgende iteranden betekent dit dat

$$\begin{aligned} \overline{x}_m^h \rightarrow \overline{\overline{x}}_m^h \quad \text{waarbij } \overline{\overline{x}}_m^h &= \overline{x}_m^h + I_H^h\,e_m^h \\ &= \overline{x}_m^h + I_H^h\,(A^H)^{-1}\,I_h^H\,r_m^h \\ &= \overline{x}_m^h + I_H^h\,(A^H)^{-1}\,I_h^H\,(b^h - A^h\,\overline{x}_m^h)\,. \end{aligned} \tag{49}$$

Dit impliceert dat de grofcorrectie kan geschreven worden als

$$e_{m+1} = K_{h,H}\,e_m \quad \text{waarbij} \quad K_{h,H} = I^h - I_H^h\,(A^H)^{-1}\,I_h^H\,A^h\,. \tag{50}$$

Omdat het optellen van de correctieterm bij de bestaande benadering mogelijk opnieuw hoogfrequente foutcomponenten introduceert, worden gewoonlijk enkele post-smoothing stappen uitgevoerd. De combinatie van de smoother met de grofroostercorrectie resulteert in de tweeroosteroperator

$$M_{h,H}(\nu_1,\,\nu_2) = (S_2^h)^{\nu_2}\,K_{h,H}\,(S_1^h)^{\nu_1}\,, \tag{51}$$

waarbij onderindices 1 en 2 worden gehanteerd om de pre- en post-smoother van elkaar te onderscheiden. De indices ν_1 en ν_2 duiden het aantal pre- en post-smoothing stappen aan.

In *multi*roostermethoden op meer dan twee roosters wordt de grofroostercorrectie (48) opgelost door het recursief toepassen van het tweeroosteralgoritme. Multiroostermethoden zijn optimaal in de zin dat de norm van de iteratiematrix kan begrensd worden door een constante onafhankelijk van de fijn-roostermaaswijdte h. Dit impliceert dat de multiroosterconvergentie onafhankelijk is van h.

5.2 De geometrische multiroostermethode voor het tijdsharmonische probleem

In de veralgemening van multiroostermethoden voor tijdsharmonische problemen, wordt de constructie van de interroosteroperatoren gebaseerd op het reëel deel van de coëfficiëntenmatrix. Deze keuze wordt gemotiveerd door het feit dat de Galerkin-discretisatie op het grof rooster

$$A^H = I_h^H\,A^h\,I_H^h \tag{52}$$

resulteert in een grof-roostermatrix A^H met dezelfde structuur en eigenschappen als de fijn-roostermatrix A^h. De smoother wordt gebaseerd op het geheel van de coëfficiëntenmatrix en is complex. Een Fourier-analyse van dit algoritme werd gegeven in [62].

Dit algoritme werd geïmplementeerd voor een modelprobleem op een vierkant. Numerieke resultaten tonen dat voor een gegeven, vaste frequentie enerzijds, de typische h-onafhankelijke convergentie wordt teruggevonden. Voor een gegeven, vaste fijn-roosterdiscretisatie anderzijds, neemt de snelheid van de multiroosterconvergentie af naarmate de frequentie toeneemt, totdat de frequentie een zekere drempelwaarde bereikt. Als men de frequentie laat toenemen voorbij deze drempelwaarde, versnelt het algoritme weer. In het modelprobleem bekomt men een snelle convergentie zelfs bij de drempelfrequentiewaarde. Deze resultaten zijn in overeenstemming met de resultaten van Fourier-analyse.

6 De algebraïsche multiroostermethode

6.1 Componenten van de algebraïsche multiroostermethoden

Algebraïsche multiroostermethoden (AMG) werden ontwikkeld als een alternatief voor het oplossen van problemen waarin de meer klassieke, geometrische multiroostermethoden niet of nauwelijks toepasbaar zijn. In deze sectie en de twee volgende zullen we de Brandt-Ruge-Stüben-aanpak van AMG voor het oplossen van het stationaire probleem bespreken [18, 67, 108]. De veralgemening van AMG voor het oplossen van het tijdsharmonische probleem zal in Sectie 6.4 worden besproken.

In het oplossen van het stelsel vergelijkingen (44) met behulp van AMG onderscheidt men twee fasen. In de *opstart*fase wordt een hiërarchie van grof-roosterdiscretisaties automatisch geconstrueerd. Hiervoor gebruikt AMG enkel informatie bevat in de fijn-roostermatrix A^h. In de *oplossings*fase wordt deze hiërarchie gebruikt voor het oplossen van het stelsel door middel van multiroosteriteraties zoals in geometrische multiroostermethoden.

In de opstartfase selecteert AMG een deelverzameling C^h van de fijnroosterpunten Ω^h. De deelverzameling C^h induceert een partitie van Ω^h, namelijk

$$\Omega^h = C^h \cup F^h \,, \tag{53}$$

die de C/F-splitsing van Ω^h wordt genoemd. Het volgend grove‘ rooster Ω^H wordt geïdentificeerd met C^h. Matrixafhankelijke interpolatie- en restrictieoperatoren I_H^h en I_h^H worden geconstrueerd. Voor symmetrische problemen

wordt de restrictie gedefinieerd als de getransponeerde van de interpolatie

$$I_h^H = (I_h^H)^T \,. \tag{54}$$

De grof-roostermatrix A^H wordt geconstrueerd aan de hand van de Galerkin-formule (52). Deze constructie van het grof-roosterprobleem wordt recursief toegepast vertrekkende van de matrix A^H. De recursie stopt indien de opvulling (fill-in) in de coëfficiëntenmatrix op het grofste rooster te groot, of het aantal punten van het grofste rooster te klein wordt.

Daar de restrictie-operator en de grof-roosterdiscretisatie vastgelegd worden door (54) en (52), dient enkel de constructie van de C/F-splitsing en van de interpolatie nog gedetailleerd te worden om de beschrijving van de opstartfase te vervolledigen.

In de oplossingsfase wordt steeds een eenvoudige punt-Gauss-Seidel-smoother aangewend. De foutcomponenten die niet door deze smoother worden gedempt, dienen wel gedempt te worden door de grof-roostercorrectie. Hiertoe wordt deze correctie aangepast aan de lokale eigenschappen van de smoother. Om deze laatste uitspraak te preciseren, worden in wat volgt gladde fouten algebraïsch gekarakteriseerd, i.e., zonder referentie naar een rooster.

6.2 Algebraïsch gladde fouten

In deze sectie en de volgende zullen de indices h en H weggelaten worden indien de betekenis van de symbolen duidelijk kan worden opgemaakt uit de context.

Een fout e wordt *algebraïsch glad* genoemd indien de smoother S gedefinieerd in (47) traag convergeert op de fout, i.e., als

$$Se \approx e \,. \tag{55}$$

Voor reële symmetrisch positief definiete M-matrices kan men, door het vergelijken van de fout in verschillende matrixnormen, tonen dat voor een algebraïsch gladde fout en het hiermee overeenkomstige residu $r = Ae$ geldt dat

$$\sum_i r_i^2/a_{ii} \ll \sum_i r_i \, e_i \,. \tag{56}$$

Heuristisch gaat men er dan van uit dat men voor algebraïsch gladde fouten kan verwachten dat

$$|r_i| \ll a_{ii}|e_i| \qquad \forall \, i. \tag{57}$$

De *omgeving* N_i van een roosterpunt i wordt gedefinieerd als

$$N_i = \{ j \neq i \,|\, a_{ij} \neq 0 \} \,. \tag{58}$$

Uit (57) leidt men af dat voor algebraïsch gladde fouten, de waarde e_i kan berekend worden als een functie van naburige waarden e_j, $j \in N_i$, door het evalueren van

$$(r_i =)\; a_{ii}\, e_i + \sum_{j \in N_i} a_{ij}\, e_j = 0 \,. \tag{59}$$

Dit is een belangrijke karakterisatie van gladde fouten daar het de basisinformatie zal leveren voor de constructie van de interpolatie. Hier zal verder worden op ingegaan in de volgende sectie.

6.3 Constructie van de interpolatie

Veronderstel dat een $C/F-$splitsing van de punten in Ω^h gegeven is. Voor de interpolatie associeert AMG met elk fijn-roosterpunt $i \in F^h$ een verzameling van interpolerende connecties $P_i \subset C^h$ en interpolatiegewichten $\{ w_{ij} \,|\, j \in P_i \}$. Men zegt dat de interpolatie gebeurt langsheen enkel directe connecties indien $P_i \subset N_i \cap C^h$. In dit geval wordt de interpolatie gedefinieerd door

$$e_i^h = (I_H^h e^H)_i = \begin{cases} e_i^H & \text{if } i \in C^h \\ \sum_{j \in P_i} w_{ij}^h e_j^H & \text{if } i \in F^h \end{cases} \,. \tag{60}$$

Om een efficiënt multiroosteralgoritme te bekomen is het nodig dat men gladde foutcomponenten nauwkeurig interpoleert. Voor symmetrische M-matrices kan de convergentie van de tweeroostermethode aangetoond worden indien na interpolatie aan (59) is voldaan $\forall i \in F^h$ [108]. Dit kan worden verwezenlijkt indien men stelt

$$\forall i \in F^h \qquad P_i = N_i \qquad \text{en} \qquad w_{ij} = -a_{ij}/a_{ii} \,. \tag{61}$$

Het stellen van $P_i = N_i$ is echter in de praktijk een te strenge eis. De verzameling C^h moet immers groot genoeg zijn om alle N_i te omvatten. Het impliceert een trage vergroving (coarsening) van Ω^h en een grote opvulling (fill-in) in de Galerkin-grof-roostermatrices (52). Daar de smoother gebaseerd is op een splitsing van de Galerkin-matrix, groeit de rekenkost van de smoothing-operatie. De kost van de multiroostercyclus wordt in grote mate bepaald door de smoothing-operatie, en de keuze (61) leidt tot multiroostercycli met een te hoge rekenkost.

De kost van de multiroostercycli kan beperkt worden door een snelle vergroving toe te staan, i.e., door de grootte van de verzameling C^h te beperken. Door de vergroving te snel uit te voeren, verliest men echter de

(snelle) convergentie van de multiroosteriteratie. In het verkleinen van de verzameling van interpolerende connecties, tracht men de convergentiesnelheid en de rekenkost per cyclus met elkaar in evenwicht te brengen.

De convergentie wordt verzekerd door te eisen dat elk fijn-roosterpunt voldoende gekoppeld is met zijn interpolerende connecties, i.e., door te eisen dat $\sum_{j \in P_i} a_{ij}$ voldoende groot is. Aan deze eis kan gemakkelijk worden voldaan door een trage vergroving toe te staan, i.e., door toe te staan dat veel punten naar het grof rooster gaan. Om de convergentie te versnellen wordt een maximaal-onafhankelijke-verzamelingvoorwaarde opgelegd aan de grof-roosterpunten. Deze voorwaarde verhindert dat twee grof-roosterpunten sterk met elkaar gekoppeld zijn. De hoger vermelde eisen op de interpolatie en de vergroving werken elkaar tegen en zijn danig dat de vergroving gebeurt in de richting van sterke koppelingen.

6.4 AMG voor tijdsharmonische problemen

In de uitbreiding van AMG voor tijdsharmonische problemen wordt de constructie van de $C/F-$splitsing en van de interpolatie-operator gebaseerd op het reëel deel van de coëfficiëntenmatrix. De Galerkin-grof-roosterdiscretisatie gebeurt met reële interpolatie- en restrictie-operatoren en met een complexe fijn-roostermatrix. We hebben twee manieren gevonden om dit algoritme te implementeren zonder daarbij bestaande AMG-codes drastisch te moeten wijzigen.

De eerste manier bestaat erin de complexe differentiaalvergelijking te schrijven als een stelsel van twee gekoppelde reële vergelijkingen. Dit stelsel kan na discretisatie opgelost worden met een AMG-code voor stelsels van differentiaalvergelijkingen. Door een geschikte keuze te maken van de parameters die de stelsel-AMG-code aansturen, bekomt men het algoritme dat hierboven werd beschreven.

De tweede manier bestaat erin een laag van routines te implementeren bovenop een AMG-code voor scalaire problemen. Deze toplaag van routines geeft het reëel deel van de matrix door aan AMG. AMG construeert de $C/F-$splitsing en de interpolatie-operator. Deze operator wordt door de toplaag van routines gebruikt als invoer voor het berekenen van het Galerkin-product. Deze procedure kan recursief worden herhaald. Eens de grof-roosterhiërarchie geconstrueerd is, kan de multiroosteriteratie in complexe variabelen worden uitgevoerd door de toplaag van routines. Dit tweede alternatief werd gedeeltelijk gerealiseerd door het ontwikkelen van een interface die toelaat de AMG-code aan te roepen vanuit `PETSc`. Deze interface zal verder worden beschreven in Hoofdstuk 8.

6.5 Numerieke resultaten

In deze sectie geven we enkele numerieke resultaten van de toepassing van AMG voor het oplossen van stationaire en tijdsharmonische problemen.

Numerieke resultaten voor het stationaire probleem

Als voorbeeld van een stationaire berekening, beschouwen we een niet-lineair model van een permanent-magneetmotor. In dit voorbeeld start men van een rooster met 1092 knopen en bekomt men na 13 adaptieve verfijningsstappen een rooster met 174.140 knopen. Op elk van deze 13 roosters lost men een niet-lineair probleem op gebruik makend van een gedempte Newton-iteratie. In elke Newton-iteratie wordt een stelsel van vergelijkingen met een Jacobiaan als coëfficiëntenmatrix opgelost. We vergelijken de performantie van de volgende iteratieve methoden voor het oplossen van deze stelsels: de Symmetrische Successieve Overrelaxatie-gepreconditioneerde CG-methode (SSOR/CG), en de AMG-solvers `AMG1R5` en `SAMG`, geïntroduceerd in Sectie 1.3. In het gebruik van AMG als solver beschouwen we de V(1,1)-, de V(2,2)-, alsook de W(1,1)-cyclus. In het gebruik van AMG als preconditioner, beschouwen we enkel de V(1,1)-cyclus.

Figuur 2(a) toont het gemiddeld aantal iteraties van `AMG1R5` per Newton-stap als een functie van de adaptieve verfijning. Figuur 2(b) toont analoge resultaten voor `SAMG`. Deze figuren tonen dat het aantal iteraties als functie van de probleemgrootte begrensd blijft en duiden dus op een multiroostergedrag. Als de AMG-codes gebruikt worden als preconditioner, daalt het aantal iteraties en is het aantal iteraties minder gevoelig aan de probleemgrootte. Dit gedrag kan worden verklaard door de convergentie van de Ritz-waarden naar de eigenwaarden van de iteratiematrix in CG-proces te analyseren. Een vergelijking van Figuur 2(a) en Figuur 2(b) toont dat voor elk van de beschouwde cyclussen `SAMG` minder iteraties vergt dan `AMG1R5`. Dit wordt verklaard door de hogere kwaliteit van de interpolatie in `SAMG`.

In Figuur 3 wordt de CPU-tijd vereist door `AMG1R5` als solver en als preconditioner vergeleken met deze vereist door de SSOR/CG-solver. Deze figuur toont dat `AMG1R5` als solver sneller is dan SSOR/CG en dat de snelheidswinst toeneemt met de probleemgrootte. Op het fijnste rooster levert het gebruik van `AMG1R5` een snelheidswinst op met een factor 7,5. Het gebruik van `AMG1R5` als preconditioner levert een bijkomende snelheidswinst op. De extra rekenkost van de uitwendige CG-iteratie is verwaarloosbaar klein ten opzichte van de winst in het aantal iteraties. Op het fijnste rooster is `AMG1R5` als preconditioner twee maal sneller dan `AMG1R5` als solver, i.e., 15 maal sneller als de SSOR-gebaseerde solver (die voorheen in de ESAT/ELEN-code was geïmplementeerd).

In Figuur 4 worden de CPU-tijden van `AMG1R5` en `SAMG` met elkaar verge-

leken. Het gebruik van SAMG als preconditioner levert het snelste algoritme. Op het fijnste rooster is SAMG 21 maal sneller dan het SSOR/CG-algoritme.

Numerieke resultaten voor het tijdsharmonische probleem

Als voorbeeld van een tijdsharmonische berekening, beschouwen we een lineair model van een 400kW-inductiemotor. In dit voorbeeld bestaat het initiële rooster uit 1028 knopen en wordt na vier verfijningsstappen een rooster met 75.951 knopen bekomen. In dit voorbeeld lossen we complexe stelsels op met een AMG-code voor stelsels differentiaalvergelijkingen die we gebruiken als een preconditioner voor het BiCGSTAB-algoritme. Dit algoritme is een Krylov-deelruimtemethode voor niet-hermitische matrices gebaseerd op korte recursiebetrekkingen en met een zacht convergentieverloop. In Figuur 5 wordt de CPU-tijd van deze solver vergeleken met deze van een ILU-gepreconditioneerde CCG-solver voor complex symmetrische matrices. Deze figuur toont dat het AMG-algoritme sneller is en dat de snelheidswinst toeneemt met de probleemgrootte, zoals in het vorige voorbeeld. Op het fijnste rooster is het AMG-algoritme 6 maal sneller.

7 De oplossing van veld-circuit gekoppelde systemen

7.1 Het multiroosterschema

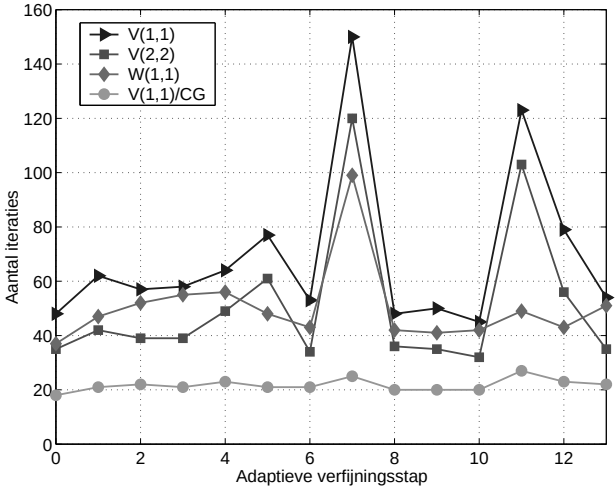
Voor de beschrijving van de veralgemening van AMG voor het oplossen van de discrete veld-circuit gekoppelde stelsels beschreven in Sectie 3.2, veronderstellen we dat de fijn-roosterdiscretisatie van het gekoppelde probleem resulteert in het volgende stelsel van vergelijkingen

$$\mathcal{A}^h \begin{pmatrix} x_h \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ g \end{pmatrix} . \tag{62}$$

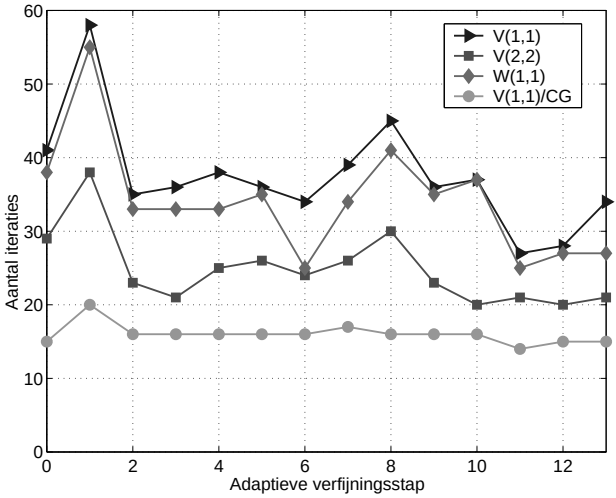
De matrix $\mathcal{A}^h$ is complex symmetrisch en heeft volgende blokstructuur

$$\mathcal{A}^h = \begin{pmatrix} A^h & B^h \\ (B^h)^T & C \end{pmatrix} . \tag{63}$$

In deze notatie komen de deelmatrices A^h, B^h en C overeen met respectievelijk de gediscretiseerde veldvergelijkingen, de koppelingstermen en het elektrisch circuit. De verzameling van vrijheidsgraden op het fijne rooster Ω^h omvat zowel de magnetische variabelen Ω^h_M als de elektrische variabelen

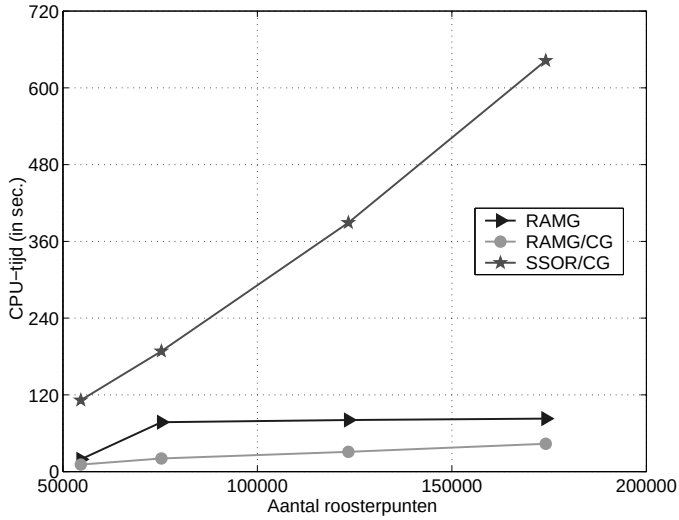


(a) Resultaten voor `AMG1R5`.

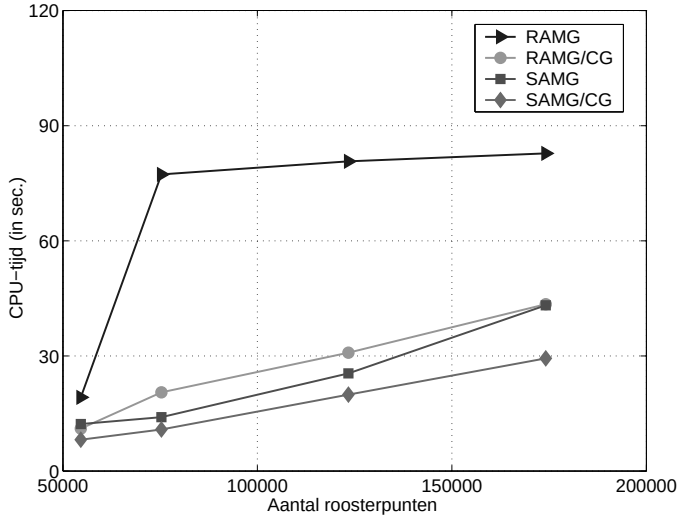


(b) Resultaten voor `SAMG`.

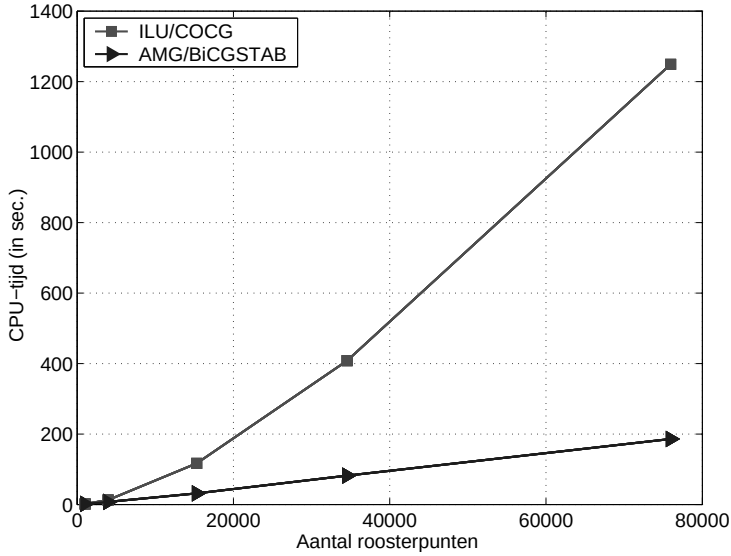
Figuur 2: Aantal iteraties van `AMG1R5` en `SAMG` versus de adaptieve verfijningsstap in de berekening van een permanent-magneetmotor. Het aantal iteraties dat getoond wordt voor een bepaalde verfijningsstap is het gemiddelde van het aantal iteraties in de individuele Newton-stappen in die beschouwde verfijningsstap.



Figuur 3: Gemiddelde CPU-tijd in elke Newton-stap van `AMG1R5` en van SSOR/CG versus het aantal roosterpunten in de laatste vier adaptieve verfijningsstappen in het voorbeeld van de permanent-magneetmotor. Resultaten voor `AMG1R5` aangewend als solver en als preconditioner worden getoond.



Figuur 4: Gemiddelde CPU-tijd in elke Newton-stap van `AMG1R5` en van `SAMG` gebruikt als solver en als preconditioner versus het aantal roosterpunten in de laatste vier adaptieve verfijningsstappen in het voorbeeld van de permanent-magneetmotor.



Figuur 5: CPU-tijd van `SAMG` als preconditioner voor het BiCGSTAB-algoritme en van ILU als preconditioner voor het COCG-algoritme versus het aantal roosterpunten in het voorbeeld van 400kW-inductiemotor.

Ω_E^h, i.e.,

$$\Omega^h = \Omega_M^h \cup \Omega_E^h \,. \tag{64}$$

Elke variabele in Ω_M^h stemt overeen met een knooppunt in het eindige-elementenrooster. Het aantal elementen van Ω_E^h is gelijk aan het aantal kringloop- en doorsnedevergelijkingen.

Het vergroven van de verzameling Ω^h gebeurt op zo'n manier dat de vrijheidsgraden in Ω_E^h tot het grof rooster behoren, i.e.,

$$\Omega_E^h \subset \Omega^H \,. \tag{65}$$

De magnetische vrijheidsgraden worden gesplitst in vrijheidsgraden op het grove en fijne rooster respectievelijk genoteerd als C_M^h en F_M^h. AMG construeert deze splitsing en de magnetische interpolatie I_H^h gebruik makend van informatie vervat in het reëel deel van de eerste diagonaalblok van $\mathcal{A}^h$. Het volgende grove rooster Ω^H en de interpolatie-operator $\mathcal{I}_h^H$ voor het gekoppelde probleem kunnen dan ingevoerd worden. De verzameling Ω^H is als volgt gedefinieerd

$$\Omega^H = \Omega_M^H \cup \Omega_E^h \,. \tag{66}$$

Dit impliceert dat de interpolatie-operator $\mathcal{I}_h^H$ de volgende blok-diagonaal-structuur heeft

$$\mathcal{I}_H^h = \begin{pmatrix} I_H^h & 0 \\ 0 & I \end{pmatrix}, \tag{67}$$

waarbij de tweede diagonaalblok de eenheidsmatrix op Ω_E voorstelt. De symmetrie van $\mathcal{A}^h$ motiveert de restrictie $\mathcal{I}_h^H$ te definiëren als de getransponeerde van de interpolatie. Het grof-roosterequivalent van $\mathcal{A}^h$ wordt berekend door het Galerkin-product

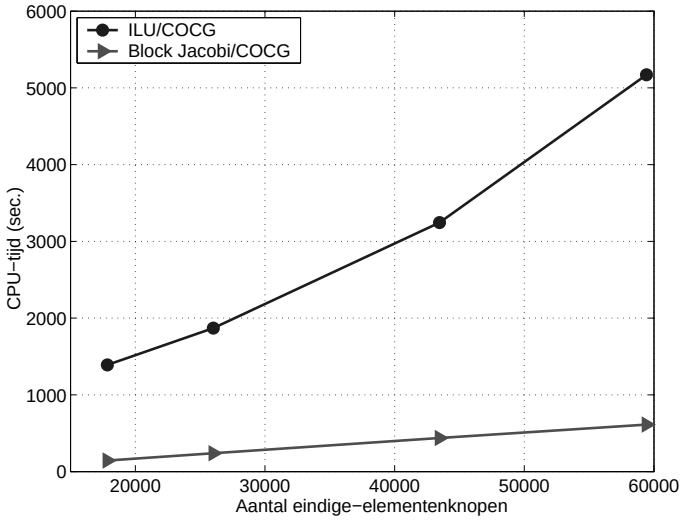
$$\mathcal{A}_H = \mathcal{I}_h^H \, \mathcal{A}_h \, \mathcal{I}_H^h = \begin{pmatrix} A_H & B^H \\ (B^H)^T & C \end{pmatrix}, \tag{68}$$

waarbij $A_H = I_h^H \, A_h \, I_H^h$ en $B^H = I_h^H \, B^h$.

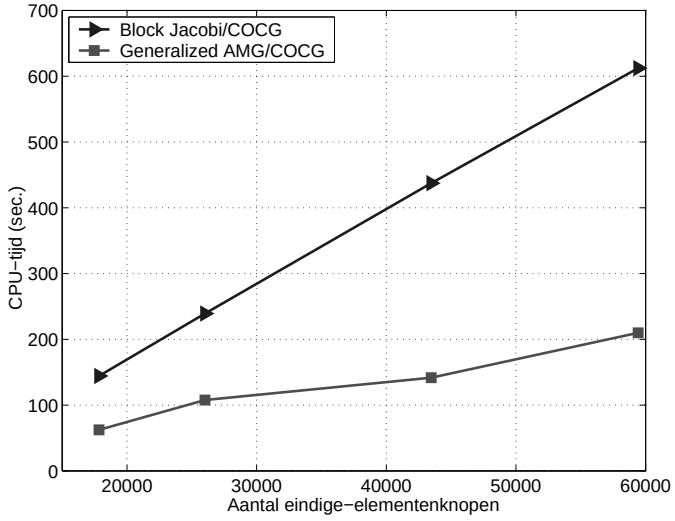
In de oplossingsfase wordt de smoothing uitgevoerd op enkel de magnetische variabelen. De smoothing laat de elektrische variabelen invariant. Gegeven een rechterlidvector (f^h, g) en een startoplossing (x_0^h, y_0) voor het stelsel (62), bestaat de smoothing uit het berekenen van een gewijzigd magnetisch rechterlid $\overline{f}^h = f^h - B^h \, y_0$ en het toepassen van Gauss-Seidel-relaxatie op het stelsel $A^h \, x^h = \overline{f}^h$. De grof-roostercorrectie wordt berekend door het oplossen van een lineair stelsel met (68) als coëfficiëntenmatrix.

7.2 Numerieke resultaten

Als testvoorbeeld voor het veralgemeende AMG-schema voorgesteld in de vorige sectie, beschouwen we een model van een 45kW-inductiemotor met 148 circuitvergelijkingen. Het finale rooster werd bekomen na 3 adaptieve verfijningsstappen en bevat in totaal 59.574 knopen. Voor dit model hebben we de CPU-tijd van drie preconditioners voor de CG-methode voor complex symmetrische matrices met elkaar vergeleken. Deze drie preconditioners zijn: de ILU- en blok-Jacobi-methode en het veralgemeende AMG-schema. In elke stap van de blok-Jacobi-methode worden de discrete veldvergelijkingen benaderend opgelost door toepassing van slechts één AMG-cyclus. De resultaten zijn weergegeven in Figuur 6 en Figuur 7. Figuur 6 toont dat de blok-Jacobi-methode sneller is dan de ILU-methode. Op het fijnste rooster levert de eerstgenoemde een winst op met een factor 8. Figuur 6 toont dat het veralgemeende AMG-schema nog sneller is dan de blok-Jacobi-methode. Op het fijnste rooster levert het AMG-schema een winst op met een factor 3. Vergeleken met de ILU-methode, is het veralgemeende AMG-schema dus sneller met een factor 24.



Figuur 6: CPU-tijd van de CG-methode voor complex symmetrische matrices met ILU en blok-Jacobi als preconditioner versus het aantal eindige-elementenroosterpunten in het voorbeeld van een 45kW-inductiemotor.



Figuur 7: CPU-tijd van de CG-methode voor complex symmetrische matrices met blok-Jacobi en het veralgemeende AMG-schema als preconditioner versus het aantal eindige-elementenroosterpunten in het voorbeeld van een 45kW-inductiemotor.

8 De AMG-PETSc interface

PETSc is een vrij beschikbare softwarebibliotheek voor het oplossen van gediscretiseerde PDEs. Het pakket omvat een breed gamma van Krylov-deelruimtesolvers en preconditioners. Zo biedt PETSc in het bijzonder een raamwerk aan voor blokpreconditioners en voor multiroostermethoden.

De uitbreidbaarheid van PETSc is danig dat functionaliteit voorzien is die de gebruiker toelaat nieuwe Krylov-deelruimtesolvers en preconditioners toe te voegen aan het pakket. Van deze functionaliteit hebben we gebruik gemaakt om de AMG-codes en PETSc te koppelen. In eerste instantie hebben we ervoor gezorgd dat het mogelijk werd om de opstart- en oplossingsfase van AMG aan te sturen vanuit PETSc. Deze koppeling noemen we de *niveau-1* AMG-PETSc koppeling. Ze laat toe om bijvoorbeeld AMG te versnellen met de Krylov-deelruimtesolvers beschikbaar in PETSc, om de convergentie van de Ritz-waarden in de Krylov-iteratie te volgen en om AMG aan te wenden als lokale solver in blokpreconditioneringsstrategieën.

In de niveau-1-koppeling neemt AMG de multiroosteriteratie voor zijn rekening. Om een flexibele omgeving te bekomen waarin we de multiroosteriteratie konden aanpassen aan onze noden, hebben we de niveau-1-interface danig uitgebreid dat PETSc de oplossingsfase van AMG overneemt. Deze koppeling noemen we de *niveau-2* AMG-PETSc koppeling. De niveau-1-koppeling biedt alle functionaliteit van de niveau-2-koppeling. Daarenboven laat de niveau-2-koppeling bijvoorbeeld toe om de grof-roostermatrices en interpolatie-operatoren geconstrueerd door AMG te visualiseren.

De niveau-2-interface bood ons een vertrekpunt voor de implementatie van het multiroosteralgoritme voor het veld-circuit gekoppelde probleem beschreven in Hoofdstuk 7. In deze uitbreiding wordt de AMG-opstartfase opgeroepen met als invoer de diagonaalblok van de coëfficiëntenmatrix die overeenkomt met de gediscretiseerde veldvergelijkingen. PETSc werd uitgebreid met routines voor de multiroosteriteratie voor het gekoppelde probleem.

9 Besluiten

Deze thesis werd gemotiveerd door de nood aan snelle procedures voor het oplossen van stelsels van lineaire vergelijkingen in een simulatiepakket voor elektrische energieomzetters. Door de introductie van algoritmen gebaseerd op de algebraï"sche multiroostermethode, zijn we er voor een groot deel in geslaagd aan deze nood tegemoet te komen.

In deze thesis werden tweedimensionale modellen beschouwd van stationaire en van quasi-stationaire tijdsharmonische magnetische velden en van

tijdsharmonische veld-circuit gekoppelde problemen. Voor de stelsels van lineaire vergelijkingen die voortvloeien uit de eindige-elementendiscretisatie van deze modellen, werd zowel het aspect van de multiroosteriteraties als dat van de Krylov-deelruimteversnelling uitvoerig bestudeerd. De voorgestelde algoritmen werden geïmplementeerd in het simulatiepakket dat in deze thesis wordt beschouwd. Op die manier konden de algoritmen uitvoerig getest worden in praktische ingenieursproblemen. Stationaire problemen werden opgelost met de oorspronkelijke AMG-code `AMG1R5` en zijn opvolger `SAMG`. Voor tijdshamonische problemen werd gebruik gemaakt van de uitbreiding van `SAMG` voor stelsels van gekoppelde vergelijkingen. Voor de veralgemening van AMG voor veld-circuit gekoppelde problemen, hebben we een interface ontwikkeld tussen de AMG-code en `PETSc`.

Vergeleken met de methoden waarmee de stelsels werden opgelost vóór de aanvang van deze thesis, levert het gebruik van AMG in de drie klassen van problemen een snelheidswinst die typisch toeneemt met de probleemgrootte. In een veld-circuit gekoppelde berekening van een inductiemotor bijvoorbeeld, reduceert de AMG-gebaseerde solver de CPU-tijd op het fijnste rooster met een factor 24.

Contents

Notations

Lower Case Greek Symbols

α		coercivity constant
δ		skin depth
ϵ		electric permittivity
	ϵ_{str}	measure of strong connectivity in AMG
	ϵ^{+}_{str}	measure of strong positive connectivity
η		damping factor in Newton's linearization
θ		angle in cylindrical coordinate system
λ		eigenvalue
μ		magnetic permeability
ν		magnetic resistivity
π		
	π_m	residual polynomial
σ		electric conductivity
ρ		electric charge density / radius / spectral radius of matrix
ϕ		electric potential
χ		symmetrization factor
ψ		
	ψ_m	search direction polynomial
ω		pulsation / relaxation parameter

Upper Case Greek Symbols

Γ		boundary of computational domain
	Γ_1	part of boundary connected to Γ_2 by (anti) periodic boundary conditions
	Γ_D	Dirichlet boundary
	Γ_m	diagonal scaling matrix in the Lanczos
Λ		
	Λ_e	eddy current function
Ξ		

	Ξ_h	set of finite element nodes
	Ξ_h^0	set of finite element nodes not lying on the boundary
Υ		
	Υ_s	source function
Ω		computational domain
	Ω^H	set of nodes on coarse grid
	Ω^h	set of nodes on fine grid

Lower Case Arabic Symbols

c		coefficients of expansion of discrete approximation in finite element basis
	$c^{[k]}$	coefficients in k-th successive substitution or Newton linearization step
	c_A	algebraic complexity of AMG
	c_G	geometric complexity of AMG
	$\mathrm{cond}(A)$	condition number of matrix A
	$\mathrm{cond}_2(A)$	condition number of matrix A in Euclidean norm
e		tree edge
f		frequency
j		complex unit, $j^2 = -1$
ℓ_z		length of conductor
n		outward normal
r		residual vector
s		
	$\mathrm{spec}(A)$	spectrum of matrix A
v		tree vertex
w_{ij}		AMG interpolation weight

Upper Case Arabic Symbols

A		coefficient matrix of linear system
	A_{ij}	ij-th component of A
	A^h	matrix of fine level discretization
	A^H	matrix of coarse level discretization / hermetian transpose of A
	A^T	transpose of A
	A_R	real part of A
	A_I	imaginary part of A
	A_*	non-symmetric equivalent real of A
	A_{**}	symmetric equivalent real of A
	A_z	z-component of vector potential
	$\widehat{A}_z$	amplitude of A_z

B		field-circuit coupling matrix
	B_f	fundamental loop matrix
	B_T	rows of B_f corresponding to the tree
	B_L	rows of B_f corresponding to the cotree
C		continuity constant of linear form / scaled electrical circuit matrix
	C_Ω	Poincar constant
	C^h	set of coarse grid points
D		diagonal of A
	D_f	fundamental cutset matrix
	D_T	rows of D_f corresponding to the tree
	D_L	rows of D_f corresponding to the cotree
F		field-circuit coupling matrix / gradient of discrete energy functional
	F^h	set of fine grid points
G		electrical conductance
H		
	$H^1(\Omega)$	function space of square integrable functions on Ω with square integrable first derivatives
	$H^1_0(\Omega)$	subspace of $H^1(\Omega)$ consisting of functions vanishing on the Dirichlet boundary
	$H^1_\parallel(\Omega)$	subspace of $H^1(\Omega)$ consisting of functions subject to (anti) periodic boundary conditions
	$H_{m,m}$	Hessenberg matrix in Arnoldi algorithm
I		current
	I_h^H	interpolation operator
	I_H^h	restriction operator
J		discrete energy functional
K		finite element triangle
	$K^m(A,v)$	Krylov subspace
	$K_{h,H}$	coarse grid correction operator
L		cotree / lower triangular part of A
	$L^2(\Omega)$	space of square integrable functions on Ω
	$L^\infty(\Omega)$	space of essentially bonded functions on Ω
	$L^m(A^H,v)$	second Krylov subspace
M		preconditioner
	$M_{h,H}$	two-grid iteration operator
N		
	N_t	number of winding of stranded conductor
	N_i	neigborhood of grid point i
P		
	P_i	interpolatory set of grid point i

R		electrical resistance
S		electrical circuit matrix / smoother
	S_{str}	cross-section of stranded conductor
	S_i	set of strong connections of grid point i
	S_A	Schur complement of A
T		tree
		tridiagonal matrix in (bi-) Lanczos algorithm
U		upper triangular part of A
V		
	V_m	basis of Krylov subspace
W		
	W_m	basis of second Krylov subspace
X		generic real finite dimensional function space
	X_h	real finite dimensional function space associated with discretization parameter h
	X_h^0	subspace of X_h consisting of functions vanishing on the Dirichlet boundary
Y		generic complex finite dimensional function space

Upper Case Bold Symbols

$\mathbf{A}$	magnetic vector potential
$\mathbf{B}$	magnetic induction
$\mathbf{D}$	electric displacement current
$\mathbf{E}$	electric field
$\mathbf{H}$	magnetic field
$\mathbf{J}$	electric current density

Calligrafic Symbols

$\mathcal{A}$	block structured matrix of field-circuit coupled problems
$\mathcal{B}^h$	basis of finite element functions
$\mathcal{F}$	conformal mapping
$\mathcal{G}$	graph
$\mathcal{L}$	differential operator
$\mathcal{P}_1$	space of polynomials of degree less than or equal to one
$\mathcal{T}_h$	triangulation

Other Symbols

$\mathbb{R}$	space of real numbers
$\mathbb{C}$	space of complex numbers
`A`	bilinear or sesquilinear form
`F`	linear form
`J`	continuous energy functional

Abbreviations

AMG	algebraic multigrid
BiCG	bi-conjugate gradient
BiCGSTAB	bi-conjugate gradient stabilised
CG	conjugate gradient
CGS	conjugate gradient squared
COCG	conjugate orthogonal conjugate gradient
FE	finite element
FGMRES	Flexible GMRES
FOM	full orthogonalisation method
GMRES	generalised minimal residual
GS	Gauss-Seidel
IC	incomplete Cholesky
JAC	Jacobi
KCL	Kirchhoff current law
KVL	Kirchhoff voltage law
MINRES	minimal residual
PDE	partial differential equation
QMR	quasi-minimal residual
RCM	reverse Cuthill-McKee
SOR	successive overrelaxation
SQMR	symmetric quasi-minimal residual
SSOR	symmetric successive overrelaxation

Chapter 1

Introduction

> Alors vous imaginez ma surprise, au lever du jour, quand une drôle de petite voix m'a réveillé. Elle disait:
> - S'il vous plaît ... dessine-moi un mouton!
> - Hein !
> - Dessine-moi un mouton ...[1]

1.1 Context of Research

In the design of electromagnetic devices, experimental work on prototypes is nowadays complemented with computer simulations. The computer models used in this context are simplified models based on the partial differential equations (PDEs) that govern the essential physics of the device under consideration. These equations are discretized to render an appropriate numerical solution possible. The computed solution is then compared with measurements. The original model is replaced by a more sophisticated one if deemed necessary. The long term goal in modeling electromagnetic devices is to avoid the need for expensive prototyping. Reaching this goal requires combining knowledge from different disciplines such as electrical engineering, applied mathematics and computer science.

This thesis contributes to the improvement of a package for the numerical simulation of quasi-stationary electromagnetic energy transducers by incorporating state of the art linear solver technology. This interdisciplinary research is part of a collaboration between two research groups of the Faculty of Applied Sciences of the Katholieke Universiteit Leuven. These groups are the Electrical Energy group of the Department of Electrical Engineering (ESAT/ELEN) and the Scientific Computing group of

[1] Antoine de Saint-Exupéry, Le Petit Prince

the Computer Science Department (CS/SciComp).

The ESAT/ELEN group has expertise in the study of electromagnetic devices such as electrical machines and transformers [30, 77, 56, 55]. Its objective is to increase the efficiency of these devices by improving their design and by minimizing undesired secondary phenomena such as the generation of heat or sound. This research might result in considerable savings in electrical energy and has as such potentially strong economical implications.

The CS/SciComp group has expertise in the design, analysis and implementation of algorithms for the fast solution of discretized PDEs [60, 47, 71]. The discretization of these equations yields systems of linear algebraic equations. The computational cost of solving these systems grows strongly with the accuracy of the discretization requested. This has stimulated research into algorithms that exploit properties of the coefficient matrix to obtain fast solution procedures. One distinguishes between direct and iterative solution methods. Three important groups of iterative techniques that are investigated by the SciComp group are: Krylov subspace [21, 64, 49, 100], multigrid [53, 75, 94, 85, 111] and domain decomposition [23, 104, 91] techniques.

1.2 The `Olympos` Package

The research in computer models for electromagnetic energy transducers in the ESAT/ELEN research group has since 1995 been concentrated around development of the `Olympos` package [86, 76, 35, 27]. Because of its importance in this thesis, we will briefly describe the package.

In `Olympos` a reduced set of the Maxwell equations is solved for the magnetic field. For the type of applications of interest to the ESAT/ELEN group, it is sufficient to consider quasi-stationary approximations, i.e., omitting any displacements currents. The package allows to simulate stationary, time-harmonic as well as transient magnetic fields. From the computed field, quantities of practical interest such as forces and torques can be derived.

The complex geometry of practical devices is in a first step approximated by modeling two-dimensional cross-sections or axisymmetric configurations. The magnetic vector potential is introduced as unknown. In the *stationary* case, the continuous mathematical model consists of a diffusion equation for the component of the vector potential perpendicular to the domain considered. This equation has to be supplied with appropriate boundary conditions. In the *time-harmonic* case, the amplitude of the vector potential is governed by the Helmholtz equation with a complex shift. In modeling saturation, the diffusion coefficient in the above PDEs becomes solution dependent and the differential equations are therefore non-linear. In practical applications, the computation of the magnetic field is impossible without

taking the properties and interconnection of the electrically conducting regions in the computational domain into consideration. In such cases, the differential equation for the magnetic field is coupled with additional equations modeling electrical circuit connections. This results in so-called *field-circuit* coupled problems.

In `Olympos` the continuous model is discretized by the finite element method. An unstructured grid of triangles covering the computational domain is constructed. On each triangle the solution is approximated by a polynomial of the first degree. The coefficients of the expansion of the solution in terms of the piecewise polynomials are given as the solution of a system of linear equations. Non-linear models are linearized and treated by iterating over a sequence of linear models. The meshes of triangles are constructed by an adaptive solution dependent refinement process. In linear problems, each refinement step requires solving a linear system. In non-linear problems, each refinement step requires solving a sequence of systems derived from the linearization.

`Olympos` is a complete finite element software package containing utilities for pre- and post-processing, mesh generation, assembling and solving the linear system and for visualizing the results. It has been successfully applied in numerous industrial and scientific research projects [86, 76, 35, 27].

Practical experience with `Olympos` has shown that solving the finite element linear systems constitutes the computational bottleneck. This bottleneck has made it extremely time-consuming to run computer simulations in which linear systems of high resolution discretizations have to be solved repeatedly. Examples of such simulations include optimization problems in which one studies the dependency of the magnetic field on a set of parameters and transient computations in which the solution is advanced from one time-step to the next. The aim of this thesis is to alleviate this bottleneck by providing numerical algorithms that are fast in CPU time and that have modest memory requirements. The algorithm central in this work is the *algebraic multigrid* (AMG) method. To introduce some terminology used further on, a brief introduction to this method is given next.

1.3 Algebraic Multigrid

The finite element discretization of PDEs yields systems with sparse coefficient matrices. This motivates looking into iterative solution procedures that have a computational kernel consisting of matrix-vector multiplications. One class of such procedures is formed by the classical matrix-splitting methods such as the Jacobi and Gauss-Seidel methods. These methods are conceptually simple, but unacceptably slow to converge on large problems. In *multigrid* enhancements of these methods, convergence

on a given fine grid is accelerated by computations on a sequence of coarser grids.

In a *two*-grid method, the classical matrix-splitting methods play the role of *smoother*. After applying a few steps of the classical method, smooth error components are restricted to the coarser grid. On the coarse grid, a correction is computed. This coarse grid correction is interpolated to the fine grid and added to the available approximation. In true *multi*grid methods, the two-grid method is applied recursively in order to compute the coarse grid correction. The recursion terminates when the cost of solving the coarsest grid problem by a direct method becomes negligible.

Multigrid algorithms in which the choice for the smoother is adapted to the coarsening strategy are called *geometric* multigrid algorithms. These algorithms constitute fast solvers for model elliptic problems and have been successfully applied in a variety of engineering disciplines. They are however cumbersome to implement for the problems with discontinuous coefficients discretized on unstructured grids that typically arise in the simulation of quasi-stationary electromagnetic energy transducers. As an alternative to geometric multigrid techniques, *algebraic* multigrid methods have therefore been developed.

In algebraic multigrid, the construction of the coarser grids is adapted to the local properties of the smoother. In the application of AMG, one distinguishes two phases. In the *set-up* phase, AMG exploits information on the strength of coupling between variables to construct a hierarchy of coarser level discretizations. This information is coded in the fine level coefficient matrix. In the *solve* phase, AMG uses this hierarchy to solve the system by multigrid cycling as in geometric multigrid. Ruge and Stüben developed the first AMG code called `AMG1R5` [95]. Stüben later enhanced `AMG1R5` in several ways, resulting in the successor of `AMG1R5` called `SAMG` [109]. The fact that AMG requires as input only the fine grid linear system, makes it attractive as a plug-in multigrid solver in finite element simulation packages. The performance of AMG can be accelerated by using it in combination with *Krylov subspace* solvers.

Krylov subspace methods for solving linear systems are non-stationary iterative methods. They produce iterands lying in a nested sequence of Krylov subspaces generated by the coefficient matrix and the initial residual vector. The convergence of these methods is determined by the eigenvalue distribution of the coefficient matrix and can be speeded up considerably by using a stationary method as preconditioner.

1.4 Contributions of the Thesis

This work has contributed to algorithmic developments for the numerical simulation of the stationary problem, the time-harmonic field problem and

the time-harmonic field-circuit coupled problem. It also led to software developments. Algorithms developed in this thesis were incorporated and tested in `Olympos`. We also developed an interface between the AMG codes and the Portable, Extensible Toolkit for Scientific Computations (`PETSc`) [5]. The latter is a general software library for solving discretized PDEs.

The discretization of *stationary* magnetic field models in `Olympos` yields real symmetric positive definite systems. For such matrices, the Conjugate Gradient method [58] is the Krylov subspace method of choice. In models with Dirichlet, Neumann and periodic boundary conditions the matrices are M-matrices. They fall into the class of matrices for which `AMG1R5` and `SAMG` were developed. For both codes we will show by numerical examples that the required number of iterations remains bounded. These examples also demonstrate that the use of AMG as a solver results in a substantial speedup compared with the one-level preconditioned conjugate gradient solver used before in `Olympos`. The amount of speedup grows with the problem size. Using AMG as a preconditioner further increases the speed and robustness of the algorithm.

The discretization of *anti-periodic* boundary conditions introduces large positive off-diagonal entries in the coefficient matrix. We discovered that these entries are not properly treated in `AMG1R5`, preventing the algorithm from converging if used without Krylov acceleration. This problem is fixed in `SAMG` by taking the large positive off-diagonal entries into account in the construction of the interpolation.

The discretization of *time-harmonic* field models yields complex symmetric systems. These systems can be solved efficiently by the Symmetric Quasi Minimal Residual Method [39]. This Krylov subspace method combines a constant amount of work and memory storage per iteration step with a smooth convergence history. These advantages are obtained by combining a quasi minimal residual approach with a Lanczos method to construct the Krylov search space.

For time-harmonic problems in which no circuit relations are present, an algebraic multigrid algorithm is constructed by basing the set-up on the strength of coupling defined by the *real* part of the coefficient matrix. In the solve phase, multigrid cycling in complex arithmetic can be performed. For the implementation of this algorithm we view the complex PDE as a coupled system of two PDEs, one for the real and one for the imaginary part of the field. The discrete system is solved using the version of `SAMG` for systems of PDEs accelerated by the BiCGSTAB method [113]. The latter is a smoothly converging Krylov subspace method for non-symmetric matrices based on short recurrences.

In problems in which *circuit relations* are present, the discretization yields complex symmetric 2×2 block-structured matrices. The first diagonal block represents the discretized PDE. The second diagonal block

models the electrical system in terms of the Kirchhoff laws. The entries in the off-diagonal blocks are coupling terms obtained by integrating the magnetic vector potential over the cross-section of the electrical conductors. We designed an algorithm that exploits this block-structure in order to reuse AMG to solve field-circuit coupled systems. In this algorithm the set-up phase uses as input the discretized PDE block of the system matrix only. The electrical circuit is taken into account in the solve phase.

For the implementation of AMG for the coupled problem, we developed an interface that allows to call `SAMG` from within `PETSc` and extended `PETSc`'s multigrid components. Our code was incorporated in the `Olympos` package. The resulting solver is such that `Olympos` handles the discretization and sets up the linear system. `SAMG` constructs the coarser level discretizations and `PETSc` is responsible for the multigrid cycling accelerated by Krylov subspace solvers. The coupling between these three codes has proven to be efficient and robust in computing models of practical relevance. Compared with the previously existing one-level preconditioned Krylov subspace solver, the new solver yielded a speedup with a factor between five and ten depending on the problem size.

1.5 Overview of the Thesis

This thesis consists of nine chapters. In Chapter 2 continuous models for magnetic field computation in electromagnetic devices are derived starting from the Maxwell equations. Particular attention is given to a concise mathematical description of field-circuit coupling. In Chapter 3 these models are discretized by the finite element method, and properties of the resulting linear systems are looked into. In Chapter 4 an overview of Krylov subspace methods is given, and their convergence properties are discussed. In Chapter 5 the fundamental concepts of multigrid in general and of geometric multigrid in particular are introduced. A model problem analysis of geometric multigrid for the time-harmonic field problem is given. In Chapter 6 algebraic multigrid is discussed in detail. The performance of the original Ruge-Stüben code `AMG1R5` and of its successor `SAMG` is validated on a set of test examples. Stationary and time-harmonic field problems are considered. In Chapter 7 several algorithms for field-circuit coupled problems are proposed and compared. Numerical results are presented showing that the generalized AMG algorithm we propose is fast both in number of iterations and in CPU time. Chapter 8 is devoted to the description of the interface between the AMG codes and `PETSc`. In Chapter 9 finally we draw some conclusions and suggest some directions for future research.

Chapter 2

Models for Magnetic Field Computations

2.1 Introduction

In this chapter we describe different partial differential equation models for the magnetic field in electromagnetic energy transducers. The continuous model is a simplification of the Maxwell equations obtained by taking the time behavior of the excitation, the spatial symmetry of the problem or particular domain characteristics into account. In Section 2.2 we introduce the Maxwell equations and in Section 2.3 we derive Ohm's law in integral form. In Sections 2.4 and 2.5 we consider stationary and quasi-stationary time-harmonic excitations and domains in two dimensions. Such two-dimensional domains represent cross-sections of the device under consideration. Assuming a stationary excitation, the magnetic field equations are decoupled from the electric ones. We introduce the magnetic vector potential as the unknown. In two-dimensional stationary formulations we arrive at a diffusion equation for the z-component of the magnetic vector potential. In the low-frequency time-harmonic formulation, an additional complex Helmholtz term appears in the partial differential equation. In some applications the computation of the time-dependent magnetic field is impossible without taking the connections of the electrical conductors present in the model into account. In Section 2.6 we present a model of the electrical circuit in terms of integrated electrical quantities, and in Sections 2.7 through 2.10 we describe how such a model can be coupled with the differential equation for the magnetic vector potential.

Non-linear magnetic constitutive relations introduced in Section 2.11 cause the diffusion coefficient to be dependent on the vector potential, giving

rise to non-linear diffusion operators. Suitable conditions on the boundary of the domain are detailed in Section 2.12. Periodic boundary conditions can be applied to reduce the area of the computational domain and a conformal mapping technique allows the treatment of problems posed on unbounded domains.

2.2 Maxwell Equations

The starting point for deriving partial differential equation models for magnetic field computations are the Maxwell equations [72, 105]. These equations relate five vector fields: the magnetic field $\mathbf{H}$, the magnetic flux density (or induction) $\mathbf{B}$, the electric field $\mathbf{E}$, the electric displacement $\mathbf{D}$ and the electric current density $\mathbf{J}$. They can be written as

$$\nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t}\,, \tag{2.1}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}\,, \tag{2.2}$$

$$\nabla \cdot \mathbf{B} = 0\,, \tag{2.3}$$

$$\nabla \cdot \mathbf{D} = \rho\,, \tag{2.4}$$

where ρ is the electric charge density. Equations (2.1) and (2.2) are Ampère's law and Faraday's law respectively. Equations (2.3) and (2.4) are conservation laws. The Maxwell equations have to be completed with constitutive material equations

$$\mathbf{B} = \mu\,\mathbf{H}\,, \tag{2.5}$$

$$\mathbf{D} = \epsilon\,\mathbf{E}\,, \tag{2.6}$$

$$\mathbf{J} = \sigma\,\mathbf{E}\,, \tag{2.7}$$

where μ, ϵ and σ are the magnetic permeability and the electric permittivity and conductivity respectively. Equations '(2.5) and (2.6) are the magnetic and dielectric relations, and equation (2.7) is Ohm's law. In the general case, the quantities μ, ϵ and σ are tensors. We will neglect any material anisotropy and assume the material coefficients to be scalars. Neither do we consider any permanent magnetic effects.

In the following sections the full set of Maxwell equations are reduced to simpler models by making assumptions on the time-behavior of the field and the dimensions of the problem.

2.3 Ohm's Law in Integral Form

Besides magnetic field quantities, which are local in nature, we will also make use of integrated electrical quantities, that are global in nature.

To introduce these quantities, we consider a cylindrical electrical conductor along the z-axis with length ℓ_z and cross-section Ω in the xy plane. The shape of the cross-section Ω is assumed to be z-independent. Consider a *stationary* electrical current flowing along the z-axis through the conductor. For stationary fields Faraday's law (2.2) implies that the electric field $\mathbf{E}$ is curl free. This condition can be ensured by writing $\mathbf{E}$ in terms of the electric potential ϕ as

$$\mathbf{E} = -\nabla\phi\,. \tag{2.8}$$

Substituting (2.8) into Ohm's law (2.7) yields

$$\mathbf{J} = -\sigma\,\nabla\phi\,. \tag{2.9}$$

For currents flowing in the z-direction only, the above relation yields

$$J_z = -\sigma\,\frac{\partial\phi}{\partial z}\,. \tag{2.10}$$

Assuming the gradient $\partial\phi/\partial z$ to be constant over the cross-section Ω, we obtain Ohm's law by integrating (2.10) over Ω in integral form

$$\triangle V = R\,I\,, \tag{2.11}$$

where

$$\triangle V = -\ell_z\frac{\partial\phi}{\partial z} \tag{2.12}$$

and

$$I = \int_\Omega J_z\,d\,\Omega \tag{2.13}$$

are the voltage drop over and the current through the conductor and where

$$R = \ell_z\,/\left(\int_\Omega \sigma\,d\,\Omega\right) \tag{2.14}$$

is the ohmic resistance of the conductor. Equation (2.11) can be rewritten as

$$I = G\ \triangle V\,, \tag{2.15}$$

where

$$G = \frac{1}{R}\,, \tag{2.16}$$

is the conductance of the conductor.

Ohm's law (2.11) needs to be generalized in two directions: we will need to consider time-varying currents and interconnections of various conductors. To take time-varying currents into account, we will distinguish different types of electrical conductors in Section 2.6. Interconnections of conductors will be considered in Sections 2.8 and 2.9.

2.4 Two-Dimensional Magnetostatic Formulation

In a stationary regime, the magnetic field quantities are decoupled from the electric ones and governed by

$$\nabla \times \mathbf{H} = \mathbf{J}\,, \tag{2.17}$$

$$\nabla \cdot \mathbf{B} = 0\,, \tag{2.18}$$

$$\mathbf{B} = \mu\,\mathbf{H}\,. \tag{2.19}$$

Equation (2.17) states that electrical charges moving uniformly in time give rise to a static magnetic field. The divergence-free condition for the magnetic flux density $\mathbf{B}$ (2.18) is ensured by expressing $\mathbf{B}$ in terms of the magnetic vector potential $\mathbf{A}$ as

$$\mathbf{B} = \nabla \times \mathbf{A}\,. \tag{2.20}$$

The substitution of (2.20) and (2.19) into (2.17) yields

$$\nabla \times (\nu\,\nabla \times \mathbf{A}) = \mathbf{J}\,, \tag{2.21}$$

where $\nu = 1/\mu$ is the magnetic reluctivity. Equation (2.21) is a coupled system of three second order PDEs for the three components of $\mathbf{A}$. Once this system has been solved for $\mathbf{A}$, the technically relevant fields $\mathbf{B}$ and $\mathbf{H}$ can be calculated by relations (2.20) and (2.19).

To avoid the complexity of solving (2.21), two-dimensional approximations are often considered in engineering practice. We will consider two such approximations: a two dimensional Cartesian formulation and an axisymmetrical formulation in cylindrical coordinates. In many applications,

these formulations already give valuable information of the device under consideration.

In two-dimensional Cartesian formulations, the problem is posed on a domain Ω lying in the xy-plane and all field quantities are assumed to be z-independent. Here Ω models for example the cross-section perpendicular to the axis of an electrical machine. The current density $\mathbf{J}$ is assumed to be perpendicular to Ω

$$\mathbf{J} = (0, 0, J_z(x, y)) \; . \tag{2.22}$$

Due to symmetry, the field quantity $\mathbf{B}$ lies in the plane Ω

$$\mathbf{B} = (B_x(x, y), B_y(x, y), 0) \; . \tag{2.23}$$

Condition (2.20) can therefore be met assuming a potential $\mathbf{A}$ of the form

$$\mathbf{A} = (0, 0, A_z(x, y)) \; . \tag{2.24}$$

By this assumption, the system (2.21) reduces to a single second order PDE for the z-component of the magnetic vector potential

$$-\frac{\partial}{\partial x}\left(\nu \frac{\partial A_z}{\partial x}\right) - \frac{\partial}{\partial y}\left(\nu \frac{\partial A_z}{\partial y}\right) = J_z \; . \tag{2.25}$$

After solving this PDE, the components of $\mathbf{B}$ in (2.23) can be computed by

$$B_x = \frac{\partial A_z}{\partial y} \text{ and } B_y = -\frac{\partial A_z}{\partial x} \; . \tag{2.26}$$

In an axi-symmetrical formulation in cylinder coordinates (r, θ, z), the domain Ω lies in the rz-plane and all field quantities are assumed to be θ-independent. In analogy to the Cartesian formulation, the vector potential is assumed to be perpendicular to Ω

$$\mathbf{A} = (0, A_\theta(r, z), 0) \; . \tag{2.27}$$

The system (2.21) then reduces to the scalar equation

$$-\frac{\partial}{\partial r}\left(\frac{\nu}{r} \frac{\partial (r\, A_\theta)}{\partial r}\right) - \frac{\partial}{\partial z}\left(\nu \frac{\partial A_\theta}{\partial z}\right) = J_\theta \; . \tag{2.28}$$

The PDE in form (2.25) or (2.28) is the governing PDE for two-dimensional stationary magnetic field computations.

2.5 Two-Dimensional Quasi-Stationary Time-Harmonic Formulation

In order to derive the time-dependent magnetic field formulation, we eliminate the current density $\mathbf{J}$ and the displacement current $\mathbf{D}$ in the right-hand side of Ampère's law (2.1) using Ohm's law (2.7) and the dielectric relation (2.6) respectively. These manipulations yield

$$\nabla \times \mathbf{H} = \sigma\, \mathbf{E} + \epsilon \frac{\partial \mathbf{E}}{\partial t}\,. \tag{2.29}$$

We assume the electro-magnetic excitations, and thus also the induced electro-magnetic fields, to vary periodically in time with period T and with frequency $f = 1/T$. We introduce the dimensionless variable $t' = t/T = f\,t$. By rewriting the time derivative in the right-hand of (2.29) in this dimensionless variable, we obtain

$$\nabla \times \mathbf{H} = \sigma\, \mathbf{E} + \epsilon\, f\, \frac{\partial \mathbf{E}}{\partial t'}\,. \tag{2.30}$$

We assume the time-variations to be *quasi-stationary*. This assumption is justified by the fact that in technically relevant simulations of electromagnetic energy transducers a sufficiently large upper bound on the frequency is

$$f \leq 10^4 \text{ Hz} \ . \tag{2.31}$$

For the electrically conducting media that we will consider, the electric conductivity σ and permeability ϵ lie in the ranges

$$\sigma \in [0, 10^9]\, C^2\, N^{-1}\, m^{-2}\, s^{-1} \text{ and } \epsilon \in [\epsilon_0, 10\,\epsilon_0]\,, \tag{2.32}$$

where $\epsilon_0 = 8.85\,.\,10^{-12} C^2\, N^{-1}\, m^{-2}$ is the permittivity of vacuum. The above bounds on f, σ and ϵ allow us to neglect the contribution of the displacement current, i.e. the second term in the right-hand side of (2.30). Hence, we obtain

$$\nabla \times \mathbf{H} = \sigma\, \mathbf{E}\,. \tag{2.33}$$

In the presence of time-varying fields, the definition (2.20) for the vector potential $\mathbf{A}$ remains valid. Introducing $\mathbf{A}$ in the left-hand side of (2.33) yields

$$\nabla \times (\nu\, \nabla \times \mathbf{A}) = \sigma\, \mathbf{E}\,. \tag{2.34}$$

By Ohm's law (2.7), this equation can be written as

$$\nabla \times (\nu \nabla \times \mathbf{A}) = \mathbf{J} \,. \tag{2.35}$$

In the presence of time-varying fields, the integration of Faraday's law (2.2) yields

$$\mathbf{E} = -\nabla\phi - \frac{\partial \mathbf{A}}{\partial t} \,, \tag{2.36}$$

where the gradient of the electric potential $\nabla\phi$ is an integration constant. The contributions of the two terms in the right-hand side of (2.36) to the current distribution will be treated separately. Substituting (2.36) into the right-hand side of (2.7) yields

$$\mathbf{J} = \mathbf{J}_s + \mathbf{J}_e \qquad \mathbf{J}_s = -\sigma \nabla\phi \qquad \mathbf{J}_e = -\sigma \frac{\partial \mathbf{A}}{\partial t} \,, \tag{2.37}$$

where $\mathbf{J}_s$ and $\mathbf{J}_e$ are the source and induced (or eddy) current density respectively. Substituting (2.37) into the right-hand side of (2.34), we obtain the following partial differential equation for the vector potential

$$\nabla \times (\nu \nabla \times \mathbf{A}) + \sigma \frac{\partial \mathbf{A}}{\partial t} = \mathbf{J}_s \,. \tag{2.38}$$

Solving the PDE (2.38) numerically entails a time-stepping procedure of some kind. In practice, a great deal of useful information can already be gained by assuming that the electromagnetic fields vary sinusoidally in time. Assuming a pulsation $\omega = 2\pi f$, we can write a generic electro-magnetic quantity $F(\mathbf{x}, t)$ oscillating harmonically in time as

$$F(\mathbf{x}, t) = \Re[\, \widehat{F}(\mathbf{x}) \exp(j\omega t) \,] \,, \tag{2.39}$$

where the magnitude $\widehat{F}$ is a complex-valued quantity consisting of a real and imaginary part $\Re[\widehat{F}]$ and $\Im[\widehat{F}]$ respectively. In a time-harmonic formulation, the PDEs (2.35) and (2.38) reduce to the following PDE for the amplitude $\widehat{\mathbf{A}}$ of the vector potential

$$\nabla \times (\nu \nabla \times \widehat{\mathbf{A}}) = \widehat{\mathbf{J}} \tag{2.40}$$

and

$$\nabla \times (\nu \nabla \times \widehat{\mathbf{A}}) + j\,\omega\sigma\,\widehat{\mathbf{A}} = \widehat{\mathbf{J}}_s \tag{2.41}$$

respectively. By a reasoning analogous to the derivation of equation (2.25), we reduce the system of coupled PDEs (2.40) and (2.41) to a scalar PDE. In the former case we obtain the Poisson equation

$$-\frac{\partial}{\partial x}\left(\nu \frac{\partial \widehat{A}_z}{\partial x}\right) - \frac{\partial}{\partial y}\left(\nu \frac{\partial \widehat{A}_z}{\partial y}\right) = \widehat{J}_z \tag{2.42}$$

while in the latter a Helmholtz equation with complex shift

$$-\frac{\partial}{\partial x}\left(\nu\,\frac{\partial \widehat{A}_z}{\partial x}\right)-\frac{\partial}{\partial y}\left(\nu\,\frac{\partial \widehat{A}_z}{\partial y}\right)+j\,\omega\,\sigma\,\widehat{A}_z=\widehat{J}_{s,z}\,, \tag{2.43}$$

where $\widehat{J}_{s,z}$ denotes the z-component of the source current density $\widehat{\mathbf{J}}_s$. Equations (2.42) and (2.43) are the governing PDEs for two-dimensional time-harmonic magnetic field formulations. In Section 2.7 we will argue which circumstances motivate the choice for either one of these equations.

In (2.39) we assumed a single frequency oscillatory behavior. In more advanced formulations a small set a discrete frequencies is taken into account. Such formulations are called *multi-harmonic* [122, 123].

2.6 Conductor Models

Time-varying currents in a conductor magnetically induce eddy currents and voltages trying to counteract their source. The current will redistribute in such a way that the current density is larger towards the cross-section boundaries. This effect is known as the skin-effect and characterized by the skin-depth. The skin-depth δ is typical length scale measuring the size of the second order term relative to the term of order zero in the PDE (2.43) and is defined as

$$\delta=\sqrt{\frac{1}{\pi f\,\sigma\mu}}\,. \tag{2.44}$$

It is represented schematically in Figure 2.1. Technically speaking, about 60% of the current flows in the shaded area in this figure. Different conductor models are distinguished based on the ratio between the skin-depth and the conductor radius, i.e. on the importance of the skin-effect in modeling the conductor. In this thesis we will consider *solid* and *stranded* conductors. For both conductor models and for time-varying field, one obtains by substituting (2.12) into (2.37) the relation

$$\widehat{J}_z=\frac{\sigma}{\ell_z}\,\widehat{\triangle V}-j\,\omega\,\sigma\,\widehat{A}_z \tag{2.45}$$

or, equivalently,

$$\widehat{\triangle V}=\ell_z\left(\frac{\widehat{J}_z}{\sigma}+j\,\omega\,\widehat{A}_z\right)\,. \tag{2.46}$$

Subscripts *sol* and *str* will be used to indicate properties associated with the solid and stranded conductors respectively.

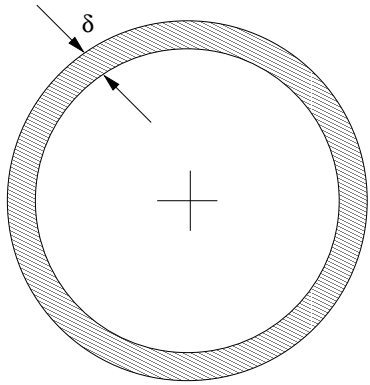


Figure 2.1: Cross-section of a conduction showing the skin-depth.

2.6.1 Solid Conductors

A solid conductor is a massive bar of electrically conducting material with homogeneous cross-section Ω_{sol}. The radius of a solid conductor is large compared with the skin-depth δ. The variation of the current density over Ω_{sol} can therefore not be neglected. The magnitude of the voltage drop $\widehat{\Delta V}_{sol}$ in contrast is constant over Ω_{sol}.

Rotor bars in an induction machine are typically modeled as solid conductors.

To generalize (2.11) for time-harmonic fields in a solid conductor, we integrate both sides of (2.45) over Ω_{sol}. Using the the fact that $\widehat{\Delta V}_{sol}$ is constant over Ω_{sol}, we find

$$\widehat{I}_{sol} = G_{sol}\,\widehat{\Delta V}_{sol} - j\,\omega \int_{\Omega_{sol}} \sigma\,\widehat{A}_z\,d\Omega\,, \tag{2.47}$$

where

$$G_{sol} = \frac{1}{\ell_z}\int_{\Omega_{sol}} \sigma\,d\Omega \tag{2.48}$$

is the Ohmic resistance of the solid conductor (see also e.g. [13]).

2.6.2 Stranded Conductors

The cross-section of a stranded conductor Ω_{str} is the union of several conductors, each of them too small to be modeled individually. The skin-depth is large compared to the radius of the individual conductors, allowing the

skin effect in the individual conductors to be neglected. The current density $\widehat{J}_z$ is assumed to be constant over Ω_{str}. Due to magnetically induced voltages, the variation of potential drop over Ω_{str} has to be taken into account. Compared with a solid conductor, the roles of current density and voltage drop are interchanged.

While the solid conductor is a physical entity, the stranded conductor is merely a convenient mathematical tool that allows one to model, for example, the windings in the stator of an induction machine or the coil of a transformer.

Assuming that the stranded conductor is made up of N_t conductors, each carrying a current $\widehat{I}_{str}$, the average current density is

$$\widehat{J}_z = \frac{N_t \, \widehat{I}_{str}}{S_{str}} \,, \tag{2.49}$$

where S_{str} is the area of Ω_{str}. The individual conductors forming the stranded conductor are assumed to be connected in series. By this assumption the potential difference $\widehat{\Delta V}_{str}$ over the latter is equal to the sum of the potential differences $\widehat{\Delta V}$ over the conductors. This sum is approximated by N_t times the average potential difference $\widehat{\Delta V}_{av}$ over an individual conductor

$$\widehat{\Delta V}_{str} = N_t \, \widehat{\Delta V}_{av} \,, \tag{2.50}$$

where

$$\widehat{\Delta V}_{av} = \frac{1}{S_{str}} \int_{\Omega_{str}} \widehat{\Delta V} \, d\Omega \,. \tag{2.51}$$

Substituting (2.46) into (2.51), taking into account that the current distribution (2.49) as well as the conductivity σ are constant over Ω, one obtains

$$\widehat{\Delta V}_{str} = R_{str} \, \widehat{I}_{str} + j \, \omega \, \frac{N_t \, \ell_z}{S_{str}} \int_{\Omega_{str}} \widehat{A}_z \, d\Omega \,, \tag{2.52}$$

see also e.g. [70, 13], where

$$R_{str} = \frac{N_t^2 \, \ell_z}{S_{str}} \int_{\Omega_{str}} \frac{1}{\sigma} \, d\Omega \tag{2.53}$$

is the Ohmic resistance of the stranded conductor. The insulation material surrounding the individual solid conductors and the air gaps between them can be taken into account by multiplying σ in (2.53) by a positive factor smaller than one. This factor is called the *slot fill* factor.

2.7 Magnetic Field in Conductors

Relations (2.47) and (2.52) generalize Ohm's law (2.11) to time-harmonic currents. The first and second term of the right-hand side of (2.47) represent the resistive and inductive currents respectively . The two terms in the right-hand side of (2.52) are the resistive and inductive voltage . The relations (2.47) and (2.52) couple magnetic field quantities with integrated electric ones and are called *electrical circuit relations* .

To compute the magnetic field over the cross-section of a conductor, appropriate field equations and circuit relations have to be coupled. The right-hand side in PDE (2.43) can be written in terms of the applied voltage $\widehat{\Delta V}_{sol}$ over a solid conductor using (2.37) and (2.12)

$$\widehat{J}_{s,z} = -\sigma \frac{\partial \phi}{\partial z} = \frac{\sigma}{\ell_z} \widehat{\Delta V}_{sol} \,, \tag{2.54}$$

and is thus constant over Ω_{sol}. It is therefore convenient to choose the PDE (2.43) in the form

$$-\frac{\partial}{\partial x}\left(\nu \frac{\partial \widehat{A}_z}{\partial x}\right) - \frac{\partial}{\partial y}\left(\nu \frac{\partial \widehat{A}_z}{\partial y}\right) + j\,\omega\,\sigma\,\widehat{A}_z = \frac{\sigma}{\ell_z}\widehat{\Delta V}_{sol} \tag{2.55}$$

for the computation of the magnetic field over Ω_{sol}. By a similar argument the magnetic field over the cross-section of a stranded conductor Ω_{str} is modeled by PDE (2.42). Upon eliminating the current density by (2.49), this PDE becomes

$$-\frac{\partial}{\partial x}\left(\nu \frac{\partial \widehat{A}_z}{\partial x}\right) - \frac{\partial}{\partial y}\left(\nu \frac{\partial \widehat{A}_z}{\partial y}\right) = \frac{N_t}{S_{str}}\widehat{I}_{str} \,. \tag{2.56}$$

Conductors can be excited by either a voltage or a current source. If a solid conductor is operated by a voltage source, the source term in the PDE (2.55) is a known constant, and the PDE can be solved. If instead the solid conductor is current driven, the source in the PDE has to be calculated from the electrical circuit relation (2.47). In this case, the PDE and the circuit relation have to be solved simultaneously for the magnetic vector potential and the applied voltage. To compute the magnetic field distribution of a voltage driven stranded conductor, the PDE (2.56) and the circuit relation (2.52) have to be solved simultaneously for the vector potential and the applied current.

2.8 Stationary Electrical Circuits

In this section and in the following one, we will discuss interconnections of conductors. Our aim is to provide information that will allow us to explain

how electrical circuits and magnetic fields can be coupled in Section 2.10. This in turn will enable us to describe the properties of the linear system resulting from the finite element discretization.

In engineering applications electrical conductors are typically interconnected and also connected to other elements in an electrical circuit. Such elements are current and voltage sources, resistances, self-inductances and capacitors connected in a network. In the discussion we will limit ourselves to circuits of conductors, resistances, current and voltage sources.

In this section we consider circuits with constant currents. In Section 2.9 we will generalize our discussion to circuits exposed to time-varying quantities.

The laws governing the electrical circuit are

- the Kirchhoff voltage law (KVL),
- the Kirchhoff current law (KCL) and
- the current-voltage relation for each component.

The KVL states that the sum of voltage drops over the components in a closed path in the circuit is zero. The KCL states that the net current entering and leaving a circuit node is zero. The current-voltage relation for a resistor is Ohm's law in either form (2.11) or (2.15).

Writing down the KVL and the generalized KCL for every possible closed path and interconnected part of the circuit results in an over-determined (but uniquely solvable) system. The method used in this thesis to obtain a *maximal set of linearly independent* KVLs and KCLs was developed in [28, 27]. This method does not impose any assumptions on the way the components in the circuit are connected and operates on a graph associated with the circuit. It is therefore called general and topological. The general circuit theory on which the method is built can be found in [25]. In outlining the method in the remainder of this section, we omit some technical details and rely on intuitive ideas of graph theory. For further details we refer the reader to the cited references.

The method associates a graph $\mathcal{G}$ with the circuit. In this graph the set of edges $\{e\}$ represents the electrical components, and the set of vertices $\{v\}$ the points at which the components are connected.

A tree T is traced through the graph and the set of edges not belonging to the tree are said to form the cotree $L = \{e\} \setminus T$. In tracing the tree through the graph, one requires the voltage sources and the solid conductors to be in the tree, and all current sources and stranded conductors to be in the cotree. Resistances are allowed to appear as both tree and cotree edges. Later on, this requirement will allow us to associate voltage and current unknowns to tree and cotree edges respectively.

In practice, simple circuits frequently occur for which the above requirement on the tree tracing procedure cannot be met. To analyze such circuits, additional so-called partial transformations have to be performed. In this thesis however, we assume the topology of the circuits to be such that these additional transformations are redundant.

We partition the tree edges into voltage sources (with index ν), solid conductors and remaining tree edges T_0

$$T = \{sol\} \cup \{\nu\} \cup T_0 \,. \tag{2.57}$$

Likewise, we partition the cotree edges into current sources (with index i), stranded conductors and remaining cotree edges L_0

$$L = \{str\} \cup \{i\} \cup L_0 \,. \tag{2.58}$$

Given the tree, fundamental loops and cutsets can be traced in the graph. A loop is a path through the graph with the same begin and end point. A *fundamental loop* is a loop consisting of a single cotree edge or of a cotree edge and one or more tree edges. A cutset is a set of edges having the property that removing the cutset from the graph results in a disconnected graph. A *fundamental cutset* is a cutset consisting of a single tree edge or of a tree edge and one or more cotree edges. To each cotree and tree edge corresponds a fundamental loop and fundamental cutset respectively.

The fundamental loop (cutset) matrix B_f (D_f) gives the edge-loop (edge-cutset) incidence for all fundamental loops (cutsets) in the graph for a given tree. The rows and columns of fundamental loop (cutset) matrices correspond to the loops (cutsets) and edges respectively. The rows of fundamental matrices are linearly independent.

No KVL is written for a current source edge, and no KCL is written for a voltage source edge. We denote by I the vector of currents through all edges except the voltage sources and by $\triangle V$ the vector of voltages over all edges except the current sources. A maximal set of linear independent KVLs and KCLs can be expressed as

$$B_f \ \triangle V = 0 \,, \tag{2.59}$$

and

$$D_f \, I = 0 \,. \tag{2.60}$$

Equations (2.59) and (2.60) are called the *loop* and *cutset equations* respectively.

The next step consists in rewriting (2.59) and (2.60) as a single square system of equations. To do so, we reorder the current and voltage unknowns

according to the tree-cotree partitioning

$$I = \begin{pmatrix} I_T \\ \hline I_L \end{pmatrix} \text{ and } \triangle V = \begin{pmatrix} \triangle V_T \\ \hline \triangle V_L \end{pmatrix} . \tag{2.61}$$

The partitionings (2.57) and (2.58) allow us to further differentiate the tree and cotree unknowns as follows

$$I = \begin{pmatrix} I_{sol} \\ I_{T_0} \\ \hline I_{str} \\ I_i \\ I_{L_0} \end{pmatrix} \text{ and } \triangle V = \begin{pmatrix} \triangle V_{sol} \\ \triangle V_\nu \\ \triangle V_{T_0} \\ \hline \triangle V_{str} \\ \triangle V_{L_0} \end{pmatrix} . \tag{2.62}$$

Likewise, we reorder rows and columns of the fundamental loop and cutset matrix

$$\begin{aligned} B_f &= [B_T \,|\, B_L] = [B_T \,|\, \mathbf{1}] \\ &= \left(\begin{array}{ccc|cc} B_{str,\,sol} & B_{str,\,\nu} & B_{str,\,T_0} & \mathbf{1} & 0 \\ B_{L_0,\,sol} & B_{L_0,\,\nu} & B_{L_0,\,T_0} & 0 & \mathbf{1} \end{array} \right) \end{aligned} \tag{2.63}$$

and

$$\begin{aligned} D_f &= [D_T \,|\, D_L] = [\mathbf{1} \,|\, D_L] \\ &= \left(\begin{array}{cc|ccc} \mathbf{1} & 0 & D_{sol,\,str} & D_{sol,\,i} & D_{sol,\,L_0} \\ 0 & \mathbf{1} & D_{T_0,\,str} & D_{T_0,\,i} & D_{T_0,\,L_0} \end{array} \right) . \end{aligned} \tag{2.64}$$

Bringing the voltage and current sources to the right-hand side, the loop equations (2.59) and cutset equations (2.60) can be rewritten as

$$\left\{ \begin{array}{rcl} B_{str,\,sol}\ \triangle V_{sol} + B_{str,\,T_0}\ \triangle V_{T_0} + \triangle V_{str} &=& -B_{str,\,\nu}\ \triangle V_\nu \\ B_{L_0,\,sol}\ \triangle V_{sol} + B_{L_0,\,T_0}\ \triangle V_{T_0} + \triangle V_{L_0} &=& -B_{L_0,\,\nu}\ \triangle V_\nu \end{array} \right. \tag{2.65}$$

and

$$\left\{ \begin{array}{rcl} I_{sol} + D_{sol,\,str}\, I_{str} + D_{sol,\,L_0}\, I_{L_0} &=& -D_{sol,\,i}\, I_i \\ I_{T_0} + D_{T_0,\,str}\, I_{str} + D_{T_0,\,L_0}\, I_{L_0} &=& -D_{T_0,\,i}\, I_i \end{array} \right. \tag{2.66}$$

respectively. The cotree voltages $\triangle V_{str}$ and $\triangle V_{L_0}$ can be eliminated from (2.65) using (2.11)

$$\begin{aligned} \triangle V_{str} &= R_{str}\, I_{str} \\ \triangle V_{L_0} &= R_{L_0}\, I_{L_0} , \end{aligned} \tag{2.67}$$

and the tree currents I_{sol} and I_{T_0} from (2.66) using (2.15)

$$\begin{aligned} I_{sol} &= G_{sol}\ \triangle V_{sol} \\ I_{T_0} &= G_{T_0}\ \triangle V_{T_0} . \end{aligned} \tag{2.68}$$

By multiplying equations (2.65) by -1, the loop and cutset equations (2.65) and (2.66) can thus be written into the following system of equations

$$S \begin{pmatrix} I_{str} \\ I_{L_0} \\ \triangle V_{sol} \\ \triangle V_{T_0} \end{pmatrix} = \begin{pmatrix} B_{str,\,\nu}\ \triangle V_\nu \\ B_{L_0,\,\nu}\ \triangle V_\nu \\ -D_{sol,\,i}\, I_i \\ -D_{T_0,\,i}\, I_i \end{pmatrix}, \tag{2.69}$$

where the matrix S is given by

$$S = \begin{pmatrix} -\mathrm{diag}[R_{str}] & 0 & -B_{str,\,sol} & -B_{str,\,T_0} \\ 0 & -\mathrm{diag}[R_{L_0}] & -B_{L_0,\,sol} & -B_{L_0,\,T_0} \\ D_{sol,\,str} & D_{sol,\,L_0} & \mathrm{diag}[G_{sol}] & 0 \\ D_{T_0,\,str} & D_{T_0,\,L_0} & 0 & \mathrm{diag}[G_{T_0}] \end{pmatrix}. \tag{2.70}$$

The blocks on the diagonal are diagonal matrices containing the resistance and admittance values. The matrix S is called the circuit system matrix. It is a square matrix whose dimension equals the number of loop equations plus the number of cutset equations. It is non-singular by construction. Due to the property [25]

$$D_{x,\,y} = -B^T_{y,\,x} \tag{2.71}$$

the matrix S is symmetric

$$S = S^T. \tag{2.72}$$

2.9 Transient Electrical Circuits

The electrical circuit system (2.69) needs to be generalized to situations where time-varying currents flow in the circuit. The magnetically induced effects in conductors must then be taken into account. To do so, we add the magnetically induced voltage over a stranded conductor and the magnetically induced current in a solid conductor, i.e., the second term in the right-hand side of equations (2.52) and (2.47) to their resistive and conductibe counterparts in the left-hand side of (2.69). Denoting by $\Omega_{str,\,p}$ and $\Omega_{sol,\,q}$ the p-th stranded and q-th solid conductor's cross-section, and by $\triangle\widehat{V}_{ind}$ and $\triangle\widehat{I}_{ind}$ the vectors of the magnitudes of the induced voltages and current with components

$$\widehat{\triangle V}_{ind,\,p} = j\,\omega\,\frac{N_{t,\,p}\,\ell_z}{S_{str,\,p}} \int_{\Omega_{str,\,p}} \widehat{A}_z\, d\Omega \qquad \text{for } p \in \{str\}, \tag{2.73}$$

and

$$\widehat{I}_{ind,\,q} = -j\,\omega \int_{\Omega_{sol,\,q}} \sigma_q\, \widehat{A}_z\, d\Omega \qquad \text{for } q \in \{sol\}, \tag{2.74}$$

we obtain for the magnitudes of the current and voltage the linear system

$$S \begin{pmatrix} \widehat{I}_{str} \\ \widehat{I}_{L_0} \\ \widehat{\Delta V}_{sol} \\ \widehat{\Delta V}_{T_0} \end{pmatrix} + \begin{pmatrix} -\widehat{\Delta V}_{ind} \\ 0 \\ \widehat{I}_{ind} \\ 0 \end{pmatrix} = \begin{pmatrix} B_{str,\,\nu}\, \widehat{\Delta V}_{\nu} \\ B_{L_0,\,\nu}\, \widehat{\Delta V}_{\nu} \\ -D_{sol,\,i}\, \widehat{I}_i \\ -D_{T_0,\,i}\, \widehat{I}_i \end{pmatrix} . \tag{2.75}$$

This circuit system has to be coupled with field equations for the integrand $\widehat{A}_z$ in the right-hand side of (2.73) and (2.74). This yields the general field-circuit coupled problem given in the following section.

2.10 Field-Circuit Coupling

In some situations the computation of the magnetic field is not possible without taking the electrical circuit connection into account. We already discussed simple examples of this situation in Section 2.7. An example of an engineering application is the simulation of an induction machine where the magnetic field cannot be computed without taking the connection of the stator winding with the drive and the interconnection of the rotor bars into consideration.

To formulate a general field-circuit coupled problem, we consider a two-dimensional domain $\Omega \subset \mathbb{R}^2$ partitioned into electrically conducting and non-conducting regions. The conducting region is the union of the cross-sections of solid and stranded conductors $\Omega_{sol,\,q}$ and $\Omega_{str,\,p}$. Denoting the non-conducting region by Ω_{core}, we have

$$\Omega = \left(\bigcup_p \Omega_{str,\,p} \right) \cup \left(\bigcup_q \Omega_{sol,\,q} \right) \cup \Omega_{core} \,. \tag{2.76}$$

By introducing the notation

$$\mathcal{L}(\widehat{A}_z) = -\frac{\partial}{\partial x} \left(\nu \frac{\partial \widehat{A}_z}{\partial x} \right) - \frac{\partial}{\partial y} \left(\nu \frac{\partial \widehat{A}_z}{\partial y} \right) , \tag{2.77}$$

the magnetic field problem on Ω can be stated as

$$\begin{aligned} \mathcal{L}(\widehat{A}_z) + j\,\omega\,\sigma\,\widehat{A}_z &= \frac{\sigma}{\ell_z}\, \widehat{\Delta V}_{sol,\,q} && \text{on } \Omega_{sol,\,q} \\ \mathcal{L}(\widehat{A}_z) &= \frac{N_{t,\,p}}{S_{str,\,p}}\, \widehat{I}_{str,\,p} && \text{on } \Omega_{str,\,p} \\ \mathcal{L}(\widehat{A}_z) &= 0 && \text{on } \Omega_{core} \,, \end{aligned} \tag{2.78}$$

supplied with appropriate boundary conditions. Given a subdomain $\Omega_0 \subset \Omega$, we introduce for future convenience the indicator function $\mathbb{I}_{\Omega_0}$ with values

$$\mathbb{I}_{\Omega_0}(\mathbf{x}) = \begin{cases} 1 \text{ for } \mathbf{x} \in \Omega_0 \\ 0 \text{ for } \mathbf{x} \in \Omega \setminus \Omega_0 \end{cases} . \tag{2.79}$$

The eddy current and source functions Λ_e and Υ_s are defined on Ω in the following way

$$\Lambda_e(\mathbf{x}) = j\,\omega \sum_q \sigma_q(\mathbf{x})\, \mathbb{I}_{\Omega_{sol,\,q}} \tag{2.80}$$

$$\Upsilon_s(\mathbf{x}) = \sum_p \frac{N_{t,\,p}}{S_{str,\,p}}\, \widehat{I}_{str,\,p}\, \mathbb{I}_{\Omega_{str,\,p}} + \sum_q \frac{\sigma_q(\mathbf{x})}{\ell_z}\, \widehat{\triangle V}_{sol,\,q}\, \mathbb{I}_{\Omega_{sol,\,q}} . \tag{2.81}$$

Using these functions, the set of differential equations (2.78) can be written as the single PDE

$$-\frac{\partial}{\partial x}\left(\nu \frac{\partial \widehat{A}_z}{\partial x}\right) - \frac{\partial}{\partial y}\left(\nu \frac{\partial \widehat{A}_z}{\partial y}\right) + \Lambda_e(\mathbf{x})\, \widehat{A}_z = \Upsilon_s(\mathbf{x}) \text{ on } \Omega . \tag{2.82}$$

If none of the conductors are interconnected, and if all solid conductors are driven by voltage source and all stranded conductors by a current source, then the source terms $\widehat{\triangle V}_{sol}$ and $\widehat{I}_{str}$ are known constants and the partial differential problem (2.78) can be solved. If instead the conductors are interconnected, (some of) the constants $\widehat{\triangle V}_{sol}$ and $\widehat{I}_{str}$ are generally unknown and the source term in the PDE (2.82) is undetermined. In this case a *magnetic field-electrical circuit coupled system* consisting of the the PDE (2.82) and linear system (2.75) have to be solved simultaneously for the magnetic vector potential and the unknown cotree currents and tree voltages.

2.11 Material Parameters

In deriving the PDEs in the previous sections, we neglected anisotropy in both the magnetic reluctivity ν and the electric conductivity σ and assumed ν and σ to be real-valued scalar quantities. In general, these quantities can be functions of the spatial coordinates over the domain Ω. The conductivity in particular can vary over the cross-section of a solid conductor. The reluctivity is strictly positive, whereas the conductivity is positive and zero over non-conducting parts of Ω. Both material parameters can have sharp discontinuities across the interface between two different materials. The magnetic permeability of air and iron, for example, differ by a factor of several thousand.

In many practical applications it is required to take the non-linear characteristic of the magnetic material into account. For such materials the magnetic constitutive equation (2.5) is a non-linear relation. In the constitutive models we consider in this work the reluctivity ν is supposed to be a function of the square of the magnitude of the magnetic induction $\mathbf{B}$. This magnitude, denoted by $|B|^2$, is in stationary formulations defined as the inner product of $\mathbf{B}$ with itself

$$|B|^2 = \frac{1}{2}\,\mathbf{B}\cdot\mathbf{B}\,, \tag{2.83}$$

and in time-harmonic formulations as the inner product of $\mathbf{B}$ with its complex conjugate

$$|B|^2 = \frac{1}{2}\,\mathbf{B}\cdot\overline{\mathbf{B}}\,. \tag{2.84}$$

For two-dimensional cartesian stationary formulation, the relations (2.26) imply that the definition (2.83) can be written as

$$|B|^2 = \frac{1}{2}\left[\left(\frac{\partial A_z}{\partial x}\right)^2 + \left(\frac{\partial A_z}{\partial y}\right)^2\right]. \tag{2.85}$$

A similar expression can be derived in two-dimensional cartesian time-harmonic formulations.

2.12 Boundary Conditions

The partial differential equations introduced before need to be supplied with boundary conditions. In this work we will make use of homogeneous Dirichlet, Neumann and periodic boundary conditions. When used in combination with a conformal mapping technique, the latter allow the modeling of infinite domains.

2.12.1 Periodic Boundary Conditions

Let Γ_1 and Γ_2 be two disjoint parts of the boundary of the computational domain as shown in Figure 2.2 on page 25 for example. Let n_1 and n_2 denote the outward normal on Γ_1 and Γ_2 respectively. Periodic boundary conditions connect both the vector potential and its normal derivative on Γ_1 and Γ_2. Periodic boundary conditions state that

$$A_z|_{\Gamma_1} = A_z|_{\Gamma_2} \tag{2.86}$$

$$\nu\,\frac{\partial A_z}{\partial n_1}|_{\Gamma_1} = -\nu\,\frac{\partial A_z}{\partial n_2}|_{\Gamma_2}\,. \tag{2.87}$$

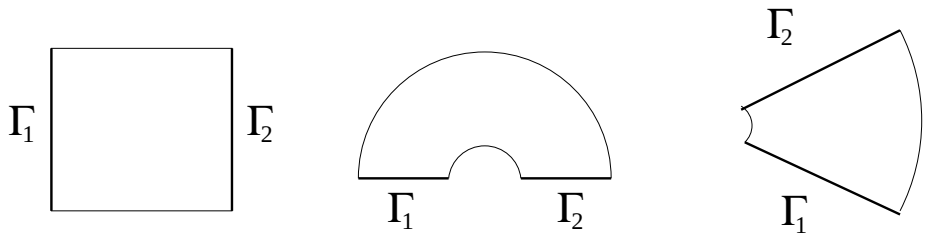


Figure 2.2: Illustration of computational domains and part of their boundary on which periodic boundary conditions can be applied. These conditions relate the values of the field on Γ_1 and Γ_2.

Anti-periodic boundary conditions state that

$$A_z|_{\Gamma_1} = -A_z|_{\Gamma_2} \tag{2.88}$$

$$\nu \frac{\partial A_z}{\partial n_1}|_{\Gamma_1} = \nu \frac{\partial A_z}{\partial n_2}|_{\Gamma_2} \tag{2.89}$$

see e.g. [91, 36]. By imposing periodic boundary conditions one is able to reduce the area of the computational domain. We illustrate this point by an example. Consider the four-pole alternating current machine depicted in Figure 2.3 on page 26. Due to symmetry in field and geometry, it is sufficient to model only two, or even just one pole of the machine. Figure 2.4 shows a two-pole half model of the motor being considered. In this model periodic boundary conditions relate the values on the straight edges at the bottom of the model. In the one-pole quarter model shown in Figure 2.4, anti-periodic boundary conditions relate the values of the vertical and horizontal straight edge.

The reduction of the computational domain entails a saving in computational resources required to solve the problem or allows to obtain higher accuracy for the same problem size.

2.12.2 Unbounded Domains

In some practical magnetic field computations the domain Ω is unbounded. Different types of so-called *open boundary problems* exist. In a first one the accurate estimation of the field in a region of interest might require taking the surrounding unbounded domain into account. In a second type one is interested in far field effects, i.e. in the strength of the field at a large distance from the source. An example of the former is the computation of the magnetic field in the vicinity of a transformer. An example of the latter is the computation of the magnetic field excited by high voltage lines at ground level.

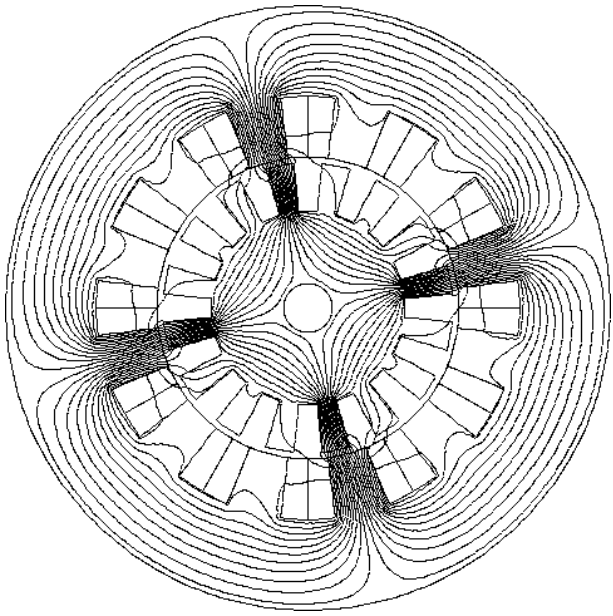


Figure 2.3: Model of an four pole alternating current machine.

In numerical computations the unbounded domain needs to be truncated somehow. An overview of truncation techniques proposed in the literature can be found in [88].

In the application software considered in this work, a conformal mapping technique is used. To describe this technique formally, consider equation (2.25) posed on the domain $\Omega = \mathbb{R}^2$. The PDE has to be supplied with the condition that the potential remains finite at infinity. By properly scaling the potential, one can assume the value at infinity to be zero, i.e.

$$A_z(x,y) \to 0 \qquad \text{as } \sqrt{x^2+y^2} \to \infty\,. \tag{2.90}$$

The part of Ω that is of interest is enclosed in a disk Ω_D centered in $(0,0)$. The radius ρ of Ω_D is required to be large enough so that the source term J_z is zero on $(\Omega_D)^c = \mathbb{R}^2 \setminus \Omega_D$. We further assume ν to be constant on $(\Omega_D)^c$. Hence, the function A_z satisfies the Laplace equation on $(\Omega_D)^c$. Let A_D and A_{D^c} denote the restriction of A_z to Ω_D and $(\Omega_D)^c$ and n_D and n_{D^c} the outward normal on Ω_D and $(\Omega_D)^c$ respectively. The original problem

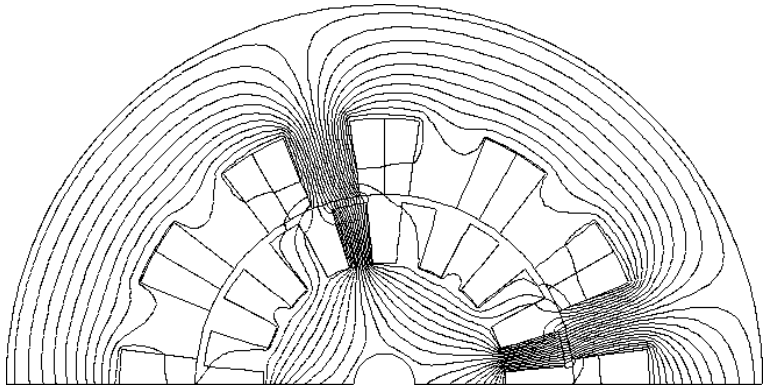


Figure 2.4: Half model of an four pole alternating current machine with periodic boundary conditions.

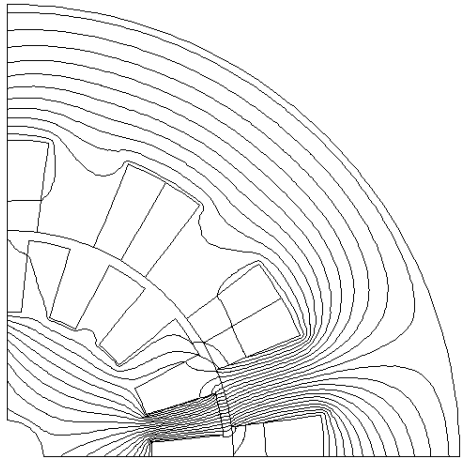


Figure 2.5: Quarter model of an four pole alternating current machine with periodic boundary conditions.

on $\mathbb{R}^2$ is equivalent with

$$\begin{aligned} -\frac{\partial}{\partial x}\left(\nu\,\frac{\partial A_D}{\partial x}\right) - \frac{\partial}{\partial y}\left(\nu\,\frac{\partial A_D}{\partial y}\right) &= J_z \qquad \text{on } \Omega_D \\ -\frac{\partial^2 A_{D^c}}{\partial x^2} - \frac{\partial^2 A_{D^c}}{\partial y^2} &= 0 \qquad \text{on } (\Omega_D)^c\,, \end{aligned} \tag{2.91}$$

supplied with conditions ensuring the continuity of A_z and its normal flux across the boundary $\partial\Omega_D$. These continuity conditions can be stated as

$$A_D|_{\partial\Omega_D} = A_{D^c}|_{\partial\Omega_D} \quad \text{and} \quad \nu\,\frac{\partial A_D}{\partial n_D}|_{\partial\Omega_D} = -\nu\,\frac{\partial A_{D^c}}{\partial n_{D^c}}|_{\partial\Omega_D}\,. \tag{2.92}$$

After setting $z = x + j\,y$, one applies the conformal mapping

$$\mathcal{F}(z) = 1/z \tag{2.93}$$

to $(\Omega_D)^c$. The latter is mapped onto a disk Ω_E with radius $1/\rho$

$$\mathcal{F}[(\Omega_D)^c] = \Omega_E\,. \tag{2.94}$$

Properties of conformal transformations imply that the function A_{D^c} is a harmonic function on the image disk Ω_E if A_{D^c} is harmonic on the domain $(\Omega_D)^c$. The conformal mapping thus transforms the differential problem (2.91)-(2.92) into

$$\begin{aligned} -\frac{\partial}{\partial x}\left(\nu\,\frac{\partial A_D}{\partial x}\right) - \frac{\partial}{\partial y}\left(\nu\,\frac{\partial A_D}{\partial y}\right) &= J_z \qquad \text{on } \Omega_D \\ -\frac{\partial^2 A_{D^c}}{\partial x^2} - \frac{\partial^2 A_{D^c}}{\partial y^2} &= 0 \qquad \text{on } \Omega_E \end{aligned} \tag{2.95}$$

supplied with the conditions

$$A_D|_{\partial\Omega_D} = A_{D^c}|_{\partial\Omega_E} \quad \text{and} \quad \nu\,\frac{\partial A_D}{\partial n_D}|_{\partial\Omega_D} = -\rho^2\,\nu\,\frac{\partial A_{D^c}}{\partial n_E}|_{\partial\Omega_E}\,, \tag{2.96}$$

where n_E denotes the outward normal on Ω_E. The factor ρ^2 appearing in (2.96) is due to the local magnification $1/|\mathcal{F}'|$ evaluated on $\partial\Omega_D$ by conformal mapping. Using a conformal mapping we have thus transformed the original unbounded domain problem into two problems on bounded domains: a Poisson type problem on Ω_D and a Laplace problem on Ω_E. The condition (2.90) is assured by imposing the potential to be zero in the center of Ω_E

$$A_z = 0 \qquad \text{in center of } \Omega_E\,. \tag{2.97}$$

Chapter 3

Finite Element Discretization

3.1 Introduction

We start Chapter 3 explaining how the continuous models from Chapter 2 are discretized to render their numerical solution possible. The continuous differential problem is cast in its variational form and discretized in space by the finite element method. The choice for this method is motivated by the fact that is has evolved into a powerful method to solve non-linear diffusion equations posed on complicated geometries. We employ linear elements on unstructured grids of triangles. The finite element discretization yields systems of linear algebraic equations for the value of the discrete vector potential at the finite element nodes.

Next we derive properties of the coefficient matrix of the linear system based on the knowledge of certain properties of its variational formulation equivalent. The discretized stationary formulation results in sparse real symmetric positive definite matrices. The discretized time-harmonic formulation results in sparse complex symmetric matrices with symmetric positive definite real part. The discretized coupled magnetic-electric formulations yield block-structured matrices in which a large complex symmetric diagonal part is bordered by small dense blocks.

We conclude this chapter by describing the electromagnetic simulation software developed inside the ESAT/ELEN research group and by giving some examples of practical engineering applications used in the following chapters.

3.2 Variational Formulation

The starting point for the finite element discretization of a differential problem is its reformulation into variational form [90, 107].

In the continuous variational formulation we will here only consider *homogeneous* Dirichlet and Neumann boundary conditions. We assume that problems with inhomogeneous Dirichlet conditions are reformulated into problems with homogeneous Dirichlet conditions by a transformation of the dependent variable. We postpone the treatment of periodic and antiperiodic boundary conditions introduced in Section 2.12.1 until Section 3.2.2.

3.2.1 Magnetostatic Variational Formulation

In this subsection we cast PDE (2.25) describing two-dimensional magnetostatic fields into variational form. Assume Ω to be a subset of $\mathbb{R}^2$ with boundary $\Gamma = \partial\Omega$. We introduce the space of square integrable functions $L^2(\Omega)$, its subspace of functions with square integrable first order derivatives $H^1(\Omega)$ and the space $L^\infty(\Omega)$ of essentially bounded functions on Ω [69]. Let Γ_D denote the part of the boundary Γ on which homogeneous Dirichlet boundary conditions are imposed, and $H_0^1(\Omega)$ the subset of functions in $H^1(\Omega)$ that vanish on Γ_D. For the magnetostatic variational formulation we consider these function spaces as vector spaces over the field of real numbers.

The space $L^2(\Omega)$ is endowed with the scalar product

$$(u,\, v) = \int_\Omega u(\mathbf{x})\, v(\mathbf{x})\, d\mathbf{x}\,, \tag{3.1}$$

and the associated norm

$$\| u \|_0 = (u,\, u)^{1/2}\,. \tag{3.2}$$

The space $L^\infty(\Omega)$ is endowed with norm

$$\| u \|_\infty = \text{ess sup}\{|u(\mathbf{x})| \;|\mathbf{x} \in \Omega\}\,. \tag{3.3}$$

The gradient

$$\nabla u = (\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}) \tag{3.4}$$

of a function $u \in H^1(\Omega)$ belongs to $L^2(\Omega) \times L^2(\Omega)$. Denoting by $\cdot$ the Euclidean scalar product, we introduce on the latter space the inner product

$$(\underline{u},\, \underline{v}) = \int_\Omega \underline{u}(\mathbf{x}) \cdot \underline{v}(\mathbf{x})\, d\mathbf{x}\,, \tag{3.5}$$

and the associated norm

$$\| \underline{u} \|_0 = (\underline{u}, \underline{u})^{1/2} . \tag{3.6}$$

The space $H^1(\Omega)$ is endowed with the norm

$$\| u \|_1 = \left(\| u \|_0^2 + \| \nabla u \|_0^2 \right)^{1/2} . \tag{3.7}$$

In the PDE (2.25), we assume the magnetic reluctivity ν to be bounded

$$\nu \in L^\infty(\Omega) , \tag{3.8}$$

and strictly positive, i.e., we assume the existence of a strictly positive constant ν_0 such that

$$0 < \nu_0 \leq \nu(\mathbf{x}) \qquad \forall \mathbf{x} \in \Omega . \tag{3.9}$$

We also assume the z-component of the current distribution function J_z to be square integrable

$$J_z \in L^2(\Omega) . \tag{3.10}$$

On $H_0^1(\Omega)$ we introduce the linear form

$$\mathtt{F} : H_0^1(\Omega) \to \mathbb{R} : \mathtt{F}(v) = (J_z, v) , \tag{3.11}$$

and on $H_0^1(\Omega) \times H_0^1(\Omega)$ the bilinear form

$$\mathtt{A} : H_0^1(\Omega) \times H_0^1(\Omega) \to \mathbb{R} : \mathtt{A}(w, v) = (\nu \nabla w, \nabla v) . \tag{3.12}$$

Condition (3.10) implies the continuity of $\mathtt{F}$

$$|\mathtt{F}(v)| \leq C \ \| v \|_0 \qquad \forall v \in H_0^1(\Omega) \tag{3.13}$$

where $C = \| J_z \|_0$, while (3.8) implies the continuity of $\mathtt{A}$

$$|\mathtt{A}(w, v)| \leq \gamma \ \| w \|_1 \| v \|_1 \qquad \forall (w, v) \in H_0^1(\Omega) \times H_0^1(\Omega) , \tag{3.14}$$

where $\gamma = \| \nu \|_\infty$. The bilinear form (3.14) is said to be coercive if there exists a constant α such that

$$\mathtt{A}(v, v) \geq \alpha \ \| v \|_1^2 \qquad \forall v \in H_0^1(\Omega) . \tag{3.15}$$

The following proposition recalls a classical result. We include its proof in order to generalize it later in time-harmonic formulations.

Proposition 3.2.1 *The bilinear form defined in (3.12) is coercive, i.e. there exists a constant $\alpha > 0$ such that (3.15) is satisfied.*

Proof. By (3.9) we have that

$$\mathtt{A}(v,\, v) = (\nu\, \nabla v,\, \nabla v) \geq \nu_0\, (\nabla v,\, \nabla v) = \nu_0\ \|\, \nabla v\, \|_0^2\ . \tag{3.16}$$

By the Poincaré inequality (see e.g. [90] Theorem 1.3.3) on the other hand there exists a constant C_Ω such that

$$\|\, v\, \|_1^2 = \|\, v\, \|_0^2 + \|\, \nabla v\, \|_0^2 \leq (1 + C_\Omega)\ \|\, \nabla v\, \|_0^2\ . \tag{3.17}$$

The substitution of (3.17) into (3.16) yields

$$\mathtt{A}(v,\, v) \geq \frac{\nu_0}{1 + C_\Omega}\ \|\, v\, \|_1^2\ . \tag{3.18}$$

Setting

$$\alpha = \frac{\nu_0}{1 + C_\Omega} \tag{3.19}$$

concludes the verification of the proposition. □

The variational formulation of problem (2.25) reads

$$\text{find } u \in H_0^1(\Omega) \text{ such that } \mathtt{A}(u,\, v) = \mathtt{F}(v) \qquad \forall\, v \in H_0^1(\Omega)\,. \tag{3.20}$$

The existence and uniqueness of the solution of (3.20) is implied by the Lax-Millgram theorem (see e.g. [90], theorem 5.1.2).

The variational formulation (3.20) is equivalent to the minimization of the functional

$$\mathtt{J}(v) = \frac{1}{2}\, \mathtt{A}(v,\, v) - \mathtt{F}(v)\,. \tag{3.21}$$

The function v corresponds to the z-component of the magnetic vector potential $\mathbf{A}$. The equivalent of $\mathtt{J}$ for three dimensional magnetostatic formulations is given by

$$\begin{aligned} \mathtt{J}(\mathbf{A}) &= \frac{1}{2} \int_\Omega \nu\, (\nabla \times \mathbf{A}) \cdot (\nabla \times \mathbf{A})\, d\Omega + \int_\Omega \mathbf{J} \cdot \mathbf{A}\, d\Omega \\ &= \frac{1}{2} \int_\Omega \mathbf{H} \cdot \mathbf{B}\, d\Omega + \int_\Omega \mathbf{J} \cdot \mathbf{A}\, d\Omega \end{aligned} \tag{3.22}$$

In this notation it is clear that the functional $\mathtt{J}$ corresponds with the magnetic energy stored in Ω.

Choosing a finite dimensional subspace $X \subset H_0^1(\Omega)$ and solving

$$\text{find } u \in X \text{ such that } \mathtt{A}(u,\, v) = \mathtt{F}(v) \qquad \forall\, v \in X\,, \tag{3.23}$$

will result in a spatial discretization of (3.20).

3.2.2 Variational Formulation of Periodic Boundary Conditions

In this section we discuss the variational formulation of differential problems with periodic and anti-periodic boundary conditions introduced in Section 2.12.1. The periodic boundary conditions on the function, i.e., the conditions (2.86) and (2.88) are imposed in strong form [36]. This is done by imposing a restriction on the function space in which the solution is sought. For the periodic boundary conditions (2.86)-(2.87) for example, this function space $H^1_{\parallel}(\Omega)$ is defined as

$$H^1_{\parallel}(\Omega) = \{v \in H^1_o \,|\, v|_{\Gamma_1} = v|_{\Gamma_2}\}\,. \tag{3.24}$$

The conditions on the normal derivative of the function, i.e., conditions (2.87) and (2.89) are imposed in weak form. For the periodic boundary conditions (2.86)-(2.87) for example, the solution u in $H^1_{\parallel}(\Omega)$ is required to satisfy

$$\int_{\Gamma_1} \nu \,\frac{\partial u}{\partial n_1}\, v \, d\Gamma_1 = -\int_{\Gamma_2} \nu \,\frac{\partial u}{\partial n_2}\, v \, d\Gamma_2 \qquad \forall\, v \in H^1_{\parallel}(\Omega)\,. \tag{3.25}$$

The bilinear form for 'both 'problems with periodic and anti-periodic boundary conditions is defined as

$$\begin{aligned} &\mathtt{A} : H^1_{\parallel}(\Omega) \times H^1_{\parallel}(\Omega) \to \mathbb{R} : \\ &\qquad \mathtt{A}(w,\, v) = (\nu\, \nabla w,\, \nabla v) + \int_{\Gamma_1} \nu \,\frac{\partial w}{\partial n_1}\, v \, d\Gamma_1 + \int_{\Gamma_2} \nu \,\frac{\partial w}{\partial n_2}\, v \, d\Gamma_2\,. \end{aligned} \tag{3.26}$$

The differential problem can now be stated variationally as in the previous subsection.

3.2.3 Time-Harmonic Variational Formulation

For the variational formulation of the two-dimensional time-harmonic PDE (2.43), we consider the function spaces $L^2(\Omega)$, $H^1_0(\Omega)$ and $L^\infty(\Omega)$ introduced previously as vector spaces over the set of complex numbers $\mathbb{C}$.

The scalar product on $L^2(\Omega)$ over $\mathbb{C}$ is defined by

$$(u,\, v) = \int_\Omega u\, \overline{v}\, d\Omega\,, \tag{3.27}$$

where $\overline{v}$ denotes the complex conjugate of v.

In the PDE (2.43), we assume the magnetic reluctivity ν and the electric conductivity σ to be bounded over Ω

$$\nu\,,\, \sigma \in L^\infty(\Omega)\,, \tag{3.28}$$

the reluctivity to be strictly positive, the conductivity to be non-negative

$$\sigma \geq 0\,, \tag{3.29}$$

and the amplitude of the z-component of the applied current distribution to be square integrable

$$\widehat{J}_{s,z} \in L^2(\Omega)\,. \tag{3.30}$$

On $H_0^1(\Omega)$ we introduce the linear form

$$\mathtt{F} : H_0^1(\Omega) \to \mathbb{C} : \mathtt{F}(v) = (\widehat{J}_{s,z},\, v)\,, \tag{3.31}$$

and on $H_0^1(\Omega) \times H_0^1(\Omega)$ the sesquilinear form

$$\mathtt{A} : H_0^1(\Omega) \times H_0^1(\Omega) \to \mathbb{C} : \mathtt{A}(w,\, v) = (\nu\, \nabla w,\, \nabla v) + j\,\omega\,(\sigma\, w,\, v) \tag{3.32}$$

Conditions (3.28) and (3.30) imply the continuity of $\mathtt{A}$ and $\mathtt{F}$ respectively. The sesquilinear form (3.32) is said to be coercive if there exists a positive constant $\alpha > 0$ such that (see e.g. [90], Remark 5.1.2)

$$|\mathtt{A}(v,\, v)| \geq \alpha \;\|\, v \,\|_1^2 \qquad\qquad \forall\, v \in H^1(\Omega)\,. \tag{3.33}$$

Proposition 3.2.2 *The sesquilinear form defined in (3.32) is coercive, i.e., there exists a constant $\alpha > 0$ such that (3.33) is satisfied.*

Proof. We have that

$$|\mathtt{A}(v,\, v)| = |\,(\nu\, \nabla v,\, \nabla v) + j\,\omega\,(\sigma\, v,\, v)\,| \geq |\,(\nu\, \nabla v,\, \nabla v)\,|\,. \tag{3.34}$$

The remainder of the proof is analogous to that of Proposition 3.2.1. □

The time-harmonic variational formulation

$$\text{find } u \in H_0^1(\Omega) \text{ such that } \mathtt{A}(u,\, v) = \mathtt{F}(v) \qquad \forall\, v \in H_0^1(\Omega)\,, \tag{3.35}$$

therefore has a unique solution by the Lax-Millgram theorem . Choosing a finite dimensional subspace $Y \subset H_0^1(\Omega)$ and solving

$$\text{find } u \in Y \text{ such that } \mathtt{A}(u,\, v) = \mathtt{F}(v) \qquad \forall\, v \in Y\,, \tag{3.36}$$

will result in the discretization of (3.35).

For the variational formulation of the PDE (2.82) in field-circuit coupled problems, the linear and sesquilinear forms $\mathtt{F}$ and $\mathtt{A}$ are defined in terms of the eddy and source current functions in the following way

$$\mathtt{F}(v) = (\Upsilon_s,\, v) \tag{3.37}$$

and

$$\mathtt{A}(w,\, v) = (\nu\, \nabla w,\, \nabla v) + (\Lambda_e\, w,\, v)\,. \tag{3.38}$$

The existence and uniqueness of the solution to the variational formulation in terms of (3.37) and (3.38), i.e. find $u \in Y$

$$\text{find } u \in Y \text{ such that } \mathtt{A}(u,\, v) = \mathtt{F}(v) \qquad \forall\, v \in Y\,, \tag{3.39}$$

can be shown by the Lax-Millgram theorem .

3.3 Finite Element Discretization

3.3.1 Triangulation and Finite Element Basic

In the finite element (FE) method [90, 107], the solution is approximated by piecewise polynomials. In this work we consider approximations by linear polynomials defined on triangles.

Assume Ω to be a polygonal domain and denote by $\mathcal{T}_h$ a triangulation of Ω, i.e. a finite set of triangles such that the intersection of two triangles is either empty, or a vertex or an edge, and such that

$$\overline{\Omega} = \cup_{K \in \mathcal{T}_h} K\,. \tag{3.40}$$

The real parameter h is a characteristic measure of the mesh width of the triangulation. Techniques to construct such triangulations are described in e.g. [44]. When Ω has smooth curved boundaries, these boundaries are only resolved in the limit that $h \to 0$.

We denote by $\mathcal{P}_1$ the space of polynomials in the variables x and y with total degree less than or equal to one. With any triangulation $\mathcal{T}_h$, we associate the finite dimensional subspace

$$X_h := \{v_h \in \mathcal{C}^0(\overline{\Omega}) \,|\, v_{h|K} \in \mathcal{P}_1\,,\ \ \forall\, K \in \mathcal{T}_h\}\,, \tag{3.41}$$

where $v_{h|K}$ is the restriction of v_h to K. Moreover, X_h^0 is the subspace

$$X_h^0 := \{v_h \in X_h \,|\, v_h = 0 \text{ on } \Gamma_D\}\,. \tag{3.42}$$

We take the FE degrees of freedom to be the values of the unknown PDE solution in the vertices of the triangles K. We denote the vertices of triangles K by $\mathbf{a}_K^i$, $i \in \{1, 2, 3\}$ and the set of global FE nodes by Ξ_h

$$\Xi_h = \{\mathbf{a}_K^i \,|\, i \in \{1, 2, 3\}\,, K \in \mathcal{T}_h\}\,, \tag{3.43}$$

and its subset Ξ_h^0

$$\Xi_h^0 = \{\mathbf{a}_K^i \,|\, i \in \{1,2,3\}\,, K \in \mathcal{T}_h\,, \mathbf{a}_K^i \notin \Gamma_D\}\,. \tag{3.44}$$

In a global enumeration over all triangles we denote the FE nodes by $\mathbf{a}_k$. To each FE node $\mathbf{a}_k \in \Xi_h^0$ we associate the function $\varphi_k \in X_h^0$ such that

$$\varphi_k(\mathbf{a}_\ell) = \delta_{k\ell} \qquad \forall \ell : \mathbf{a}_\ell \in \Xi_h \tag{3.45}$$

where $\delta_{k\ell}$ is the Kronecker delta. The set $\mathcal{B}^h$

$$\mathcal{B}^h = \{\varphi_k \,|\, k = 1, \ldots, N^h\} \tag{3.46}$$

then forms a basis for X_h^0.

3.3.2 Magnetostatic Discretization

The magnetostatic variational formulation (3.23) is discretized by setting the space X in (3.23) equal to the space X_h^0 defined in (3.42) and approximating the solution u by a sum of basis functions of X_h^0

$$u_h = \sum_{k\,:\,\mathbf{a}_k \in \Xi_h^0} c_k\, \varphi_k\,. \tag{3.47}$$

The real-valued coefficients c_k represent the values of u_h in the FE nodes. These coefficients are the solution of a linear algebraic system

$$A\,c = f\,, \tag{3.48}$$

obtained by substituting the discrete potential (3.47) into the variational formulation (3.23). The entries of the matrix A and f are given by

$$A_{k\ell} = \mathtt{A}(\varphi_k,\, \varphi_\ell) = \int_\Omega \nu\, \nabla\varphi_k(\mathbf{x}) \cdot \nabla\varphi_\ell(\mathbf{x})\, d\mathbf{x}\,, \tag{3.49}$$

and

$$f_k = \mathtt{F}(\varphi_k) = \int_\Omega J_z(\mathbf{x})\, \varphi_k(\mathbf{x})\, d\mathbf{x} \tag{3.50}$$

respectively.

3.3.3 Time-Harmonic Discretization

In the time-harmonic variational formulation (3.35) the space X^0 is a vector space over $\mathbb{C}$. The discrete vector potential (3.47) is expanded as a linear

combination of the real-valued FE basis functions (3.46) with complex coefficients c_i. These coefficients are the solution of the linear system

$$A\,c = f\,, \tag{3.51}$$

with

$$A = A_R + jA_I\,, \tag{3.52}$$

where the elements of the real and imaginary part A_R and A_I of the matrix A are given by

$$A_{R,\,k\ell} = \Re[\mathtt{A}(\varphi_k,\,\varphi_\ell)] = \int_\Omega \nu\,\nabla\varphi_k(\mathbf{x})\cdot\nabla\varphi_\ell(\mathbf{x})\,d\mathbf{x} \tag{3.53}$$

and

$$A_{I,\,k\ell} = \Im[\mathtt{A}(\varphi_k,\,\varphi_\ell)] = \omega\int_\Omega \sigma\,\varphi_k(\mathbf{x})\,\varphi_\ell(\mathbf{x})\,d\mathbf{x} \tag{3.54}$$

and where the components of the right-hand side vector f are given by

$$f_k = \int_\Omega \hat{J}_s(\mathbf{x})\,\varphi_k(\mathbf{x})\,d\mathbf{x}\,. \tag{3.55}$$

3.3.4 Coupled Field-Circuit Discretization

In field-circuit coupled problems, the variational formulation of the PDE is discretized by substituting approximation (3.47) into (3.39), resulting in the linear system

$$A\,c = f\,, \tag{3.56}$$

where the components of the real and imaginary part A_R and A_I of the matrix are given by

$$A_{R,\,k\ell} = \Re[\mathtt{A}(\varphi_k,\,\varphi_\ell)] = \int_\Omega \nu\,\nabla\varphi_k(\mathbf{x})\cdot\nabla\varphi_\ell(\mathbf{x})\,d\mathbf{x} \tag{3.57}$$

and

$$A_{I,\,k\ell} = \Im[\mathtt{A}(\varphi_k,\,\varphi_\ell)] = \int_\Omega \Lambda_s(\mathbf{x})\,\varphi_k(\mathbf{x})\,\varphi_\ell(\mathbf{x})\,d\mathbf{x} \tag{3.58}$$

and where the components of the right-hand side vector f are given by

$$f_k = \mathtt{F}(\varphi_k) = \int_\Omega \Upsilon_s(\mathbf{x})\,\varphi_k(\mathbf{x})\,d\mathbf{x}\,. \tag{3.59}$$

Substituting definition (2.81) of the source function Υ_s into the expression for the right-hand components, these components can be written as

$$\begin{aligned} f_k = & \sum_p \frac{N_{t,\,p}}{S_{str,\,p}} \widehat{I}_{str,\,p} \int_{\Omega_{str,\,p}} \varphi_k(\mathbf{x})\, d\Omega \\ & + \frac{1}{\ell_z} \sum_q \widehat{\Delta V}_{sol,\,q} \int_{\Omega_{sol,\,q}} \sigma_q(\mathbf{x})\, \varphi_k(\mathbf{x})\, d\Omega\,. \end{aligned} \tag{3.60}$$

By defining the small rectangular matrices P and Q via their entries as

$$P_{kp} = \frac{N_{t,\,p}}{S_{str,\,p}} \int_{\Omega_{str,\,p}} \varphi_k(\mathbf{x})\, d\Omega\,, \quad \text{for } k : \mathbf{a}_k \in \Xi_h^0\,,\ p \in \{str\}\,, \tag{3.61}$$

and

$$Q_{kq} = \int_{\Omega_{sol,\,q}} \frac{\sigma_q(\mathbf{x})}{\ell_z}\, \varphi_k(\mathbf{x})\, d\Omega\,, \quad \text{for } k : \mathbf{a}_k \in \Xi_h^0\,,\ q \in \{sol\}\,, \tag{3.62}$$

and the matrix F defined in terms of P and Q as

$$F = \begin{pmatrix} P & 0 & Q & 0 \end{pmatrix}, \tag{3.63}$$

the right-hand side of the linear system (3.56) can be rewritten as

$$\begin{aligned} f &= P\, \widehat{I}_{str} + Q\, \widehat{\Delta V}_{sol} \\ &= F \begin{pmatrix} \widehat{I}_{str} \\ \widehat{I}_{L_0} \\ \widehat{V}_{sol} \\ \widehat{V}_{T_0} \end{pmatrix} \end{aligned} \tag{3.64}$$

In the electrical circuit relations (2.75) the magnetically induced voltage (2.73) and current (2.74) need to be discretized. This is done by approximating the integrand $\widehat{A}_z$ in (2.73) and (2.74) by the finite element function (3.47), resulting in the following expressions

$$\widehat{V}_{ind,\,p} \approx j\,\omega\,\ell_z \sum_i P_{ip}\, c_i\,, \tag{3.65}$$

and

$$\widehat{I}_{ind,\,q} \approx -j\,\omega\,\ell_z \sum_i Q_{iq}\, c_i\,. \tag{3.66}$$

We thus obtain the following discretized electrical system

$$S \begin{pmatrix} \widehat{I}_{str} \\ \widehat{I}_{L_0} \\ \widehat{\Delta V}_{sol} \\ \widehat{\Delta V}_{T_0} \end{pmatrix} - j\,\omega \ell_z\, F^T\, c = \begin{pmatrix} B_{str,\,\nu}\, \widehat{\Delta V}_{\nu} \\ B_{L_0,\,\nu}\, \widehat{\Delta V}_{\nu} \\ -D_{sol,\,i}\, \widehat{I}_i \\ -D_{T_0,\,i}\, \widehat{I}_i \end{pmatrix} . \tag{3.67}$$

By multiplying both sides of (3.67) with the scalar χ given by

$$\chi = 1/j\,\omega \ell_z \tag{3.68}$$

we can finally combine (3.56), (3.64) and (3.67) into the single linear system

$$\mathcal{A} \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} 0 \\ \chi\, g \end{pmatrix} , \tag{3.69}$$

where we introduced the matrix

$$\mathcal{A} = \begin{pmatrix} A & -F \\ -F^T & \chi S \end{pmatrix} \tag{3.70}$$

and the vectors

$$d = \begin{pmatrix} \widehat{I}_{str} \\ \widehat{I}_{L_0} \\ \widehat{\Delta V}_{sol} \\ \widehat{\Delta V}_{T_0} \end{pmatrix} \text{ and } g = \begin{pmatrix} B_{str,\,\nu}\, \widehat{\Delta V}_{\nu} \\ B_{L_0,\,\nu}\, \widehat{\Delta V}_{\nu} \\ -D_{sol,\,i}\, \widehat{I}_i \\ -D_{T_0,\,i}\, \widehat{I}_i \end{pmatrix} . \tag{3.71}$$

3.3.5 Linear System Setup

Given an integrand $\varphi(\mathbf{x})$ defined over Ω, the triangulation (3.40) allows one to replace the integration over Ω by a sum of integrations over triangles $K \in \mathcal{T}_h$

$$\int_{\Omega} \varphi(\mathbf{x})\, d\mathbf{x} = \sum_{K \in \mathcal{T}_h} \int_K \varphi(\mathbf{x})\, d\mathbf{x} . \tag{3.72}$$

This element-wise computation of the entries of the matrices and right-hand sides in the previous subsections is important in both deriving properties of the system matrices and the computer implementation of the set-up of the linear system.

3.4 Discretization of Non-Linear Problems

In the case that the magnetic constitutive relation is non-linear, the discrete magnetic field equations are derived from an energy minimization principle. We will perform this minimization in the discrete setting only.

The discrete counterpart J of the energy functional defined in (3.22) is in stationary formulations given by

$$J(c) = \frac{1}{2} c^T A(c)\, c - f^T c\,, \tag{3.73}$$

where the matrix A and the vectors f and c are defined in (3.49), (3.50) and (3.47) respectively. The matrix A depends on the coefficient c through the reluctivity. In time-harmonic formulations, the matrix A and the vector f have to be replaced with (3.51) and (3.55) respectively. In field-circuit coupled formulations, the discrete energy functional is given in terms of the matrix and vectors introduced in (3.69).

The coefficients c for which the energy is minimal is sought by setting its gradient ∇J, also denoted by F, equal to zero. We thus obtain

$$\nabla J = 0 \Leftrightarrow F(c) = A(c)\, c - f = 0. \tag{3.74}$$

Let N denote the number of degrees of freedom in the finite element approximation. In the stationary formulations F maps from $\mathbb{R}^N$ to $\mathbb{R}^N$, while in time-harmonic formulations F maps from $\mathbb{C}^N$ to $\mathbb{C}^N$. In case of linear magnetic materials, the matrix A is independent of c, and we retrieve the equations derived previously. In the case of non-linear materials, (3.74) defines a system of non-linear equations that can be solved for example by Picard or Newton-Raphson iteration. Both methods handle the non-linearity by iterating over a sequence of linear problems.

In the following two subsections we describe both linearization methods formally. General references on this subject include [33, 84]. In the third subsection we apply Newton's method in stationary formulations. In the final subsection we will recall results showing that in time-harmonic formulations the function F defined in (3.74) is not analytic. T'herefore Newton's method has to applied to an equivalent real formulation of the system (3.74).

3.4.1 Picard Iteration

In the Picard iteration, the system (3.74) is linearized as follows: given $c^{[k]}$, the solution computed at iteration step k, the matrix $A(c^{[k]})$ is evaluated, and the linear system

$$A(c^{[k]})\, c^{[k+1]} = f\,, \tag{3.75}$$

is solved for the next approximate solution $c^{[k+1]}$. The iteration continues until the norm of the non-linear residual

$$r^{[k+1]} = f - A(c^{[k]})\, c^{[k+1]}\,, \tag{3.76}$$

drops below a prescribed tolerance. By viewing the Picard iteration as a fixed-point iteration, one can show that, for starting values $c^{[0]}$ sufficiently close to the solution of the non-linear problem, the convergence of the Picard iteration is linear. The convergence of the Picard iteration can be stabilized by introducing a damping parameter η with values between 0 and 1, and defining the iteration

$$c^{[k+1]} = \eta\, c^{[k+1]}_{\text{Picard}} + (1-\eta)\, c^{[k]}_{\text{Picard}}\,. \tag{3.77}$$

The parameter η can be chosen by a line-search strategy, see e.g., [76, 33].

3.4.2 Newton Iteration

In the Newton-Raphson linearization the iterands are defined through a Taylor series expansion up to first order of the function F around a point $c^{[k]}$ in $\mathbb{R}^N$ (if F is real valued) or in $\mathbb{C}^N$ (if F is complex valued)

$$F(c) = F(c^{[k]}) + \nabla F(c^{[k]})\,(c - c^{[k]}) + \mathcal{O}(c - c^{[k]})^2\,, \tag{3.78}$$

where ∇F is the Jacobian of F. The recursion formula for the Newton iterands is then given by

$$c^{[k+1]} = c^{[k]} - [\nabla F(c^{[k]})]^{-1}\, F(c^{[k]})\,, \tag{3.79}$$

For starting approximations $c^{[0]}$ sufficient close to the solution of the non-linear problem, the convergence of Newton-Raphson iteration is *quadratic*. As in the Picard iteration, the Newton iteration can be damped by introducing a parameter η.

3.4.3 Non-Linear Stationary Problems

In this section we apply Newton's method in two-dimensional stationary problems. Our aim is to derive an explicit expression for the Jacobian ∇F required in the iteration (3.79).

The i-th component of the function F defined in (3.74) is given by

$$F_i(c) = \sum_m A_{im}(c)\, c_m - f_i\,. \tag{3.80}$$

The ij-th component of ∇F evaluated in c^k is thus by (3.49) equal to

$$\begin{aligned}(\nabla F)_{ij}|_{c=c^{[k]}} &= \frac{\partial F_i}{\partial c_j}|_{c=c^{[k]}} \\ &= A_{ij}(c^{[k]}) + \sum_m c_m^{[k]} \frac{\partial A_{im}}{\partial c_j}|_{c=c^{[k]}} \\ &= A_{ij}(c^{[k]}) \\ &\quad + \sum_m c_m^{[k]} \int_\Omega \frac{\partial \nu}{\partial c_j}|_{c=c^{[k]}} \, (\nabla\varphi_i \cdot \nabla\varphi_m) \, d\Omega \, .\end{aligned} \tag{3.81}$$

A matrix entry A_{ij} is non-zero only if the support of the FE basis functions φ_i and φ_j overlap. The same holds true for a matrix entry P_{ij}. The matrices A and P have thus the same non-zero structure. To compute the derivative under the integral sign, we assume ν to be a function of the finite element approximation to $|B|^2$. An expression for this approximation, denoted by $|B_h|^2$, is obtained by substituting the finite element approximation (3.47) for A_z into (2.85). This yields

$$|B_h|^2 = \frac{1}{2} \sum_{m\,n} c_m \, c_n \, (\nabla\varphi_m \cdot \nabla\varphi_n) \, . \tag{3.82}$$

We therefore have that

$$\frac{\partial \nu}{\partial c_j} = \frac{d\,\nu}{d\,|B_h|^2} \frac{\partial |B_h|^2}{\partial c_j} = \frac{d\,\nu}{d\,|B_h|^2} \sum_n c_n \, (\nabla\varphi_j \cdot \nabla\varphi_n) \, . \tag{3.83}$$

Substituting this expression into (3.81), we finally obtain the following expression for the Jacobian

$$(\nabla F)_{ij}|_{c=c^{[k]}} = A_{ij} + P_{ij} \, , \tag{3.84}$$

where

$$P_{ij} = \sum_{m\,n} c_m^{[k]} \, c_n^{[k]} \int_\Omega \frac{d\,\nu}{d\,|B_h|^2}|_{c=c^{[k]}} \, (\nabla\varphi_i \cdot \nabla\varphi_m) \, (\nabla\varphi_j \cdot \nabla\varphi_n) \, d\Omega \, . \tag{3.85}$$

Numerical values for the derivative $d\,\nu/d\,|B_h|^2$ are obtained from given data of the material under consideration.

3.4.4 Non-Linear Time-Harmonic Problems

In this subsection we consider the non-linear system (3.74) in time-harmonic formulations. We here recall the main results of [68]. The function F defined

in (3.74) does not satisfy the Cauchy-Riemann equations and is therefore not analytical. In applying Newton's method in complex arithmetic, the entries of the Jacobian $\partial F_i/\partial c_j$ can therefore not be expressed in terms of $\partial F_i/\partial \xi_j$ and $\partial F_i/\partial \eta_j$, where ξ_j and η_j denote the real and imaginary part of c_j respectively. The latter is necessary however to take the magnetic constitutive relation into account. The solution proposed in [68] is to apply Newton's method to an equivalent real formulation of the non-linear system, i.e. to apply Newton's method separately to the real and imaginary part of F. The resulting Jacobian in this equivalent is non-symmetric.

Solving non-symmetric systems can be avoided by resorting to Picard's method. As the latter converges only linearly, more non-linear iterations will be required with this approach in general.

3.5 Linear Systems Properties

3.5.1 Numerical Linear Algebra Concepts

In order to discuss the properties of the linear system resulting from the finite element discretization, it is useful to recall some concepts from the theory of linear algebra [46].

We will use the subscript H for the complex conjugate transpose of a complex matrix. Let A be a square complex matrix of size N. The matrix A is said to be Hermitian if $A = A^H$.

Let $\lambda_i\,, i = 1, \ldots, N$ be the eigenvalues of A. The *spectral radius* of A is defined as

$$\rho(A) = \max\{\mid \lambda_i \mid \mid i = 1, \ldots, N\}\ . \tag{3.86}$$

Given a matrix norm $\| \ . \ \|$, the *condition number* of a non-singular matrix with respect to that norm is the value

$$\mathrm{cond}(A) = \| \ A \ \| \ \| \ A^{-1} \ \|\ . \tag{3.87}$$

For Hermitian matrices, the condition number in Euclidean norm is given by

$$\mathrm{cond}_2(A) = \mid \lambda_{\max} \mid / \mid \lambda_{\min} \mid . \tag{3.88}$$

A matrix is said to be *ill*-conditioned if its condition number is large.

The remainder of this section is restricted to *real* matrices. A square matrix A is said to be *positive definite* if

$$\mathbf{w}^T A \mathbf{w} > 0 \text{ for all } \mathbf{w} \in \mathbb{R}^N\ ,\ \mathbf{w} \neq 0 \tag{3.89}$$

Given $s \in \mathbb{R}$ and I the identity matrix, any matrix of the form

$$A = s\,I - B, \qquad s > 0\,, \qquad B_{ij} \geq 0\,, \tag{3.90}$$

for which $s \geq \rho(B)$ is called an *M-matrix* [14] .

3.5.2 Magnetostatic System Matrix Properties

In this section we describe the properties of the discrete magnetostatic system matrix in (3.48) [90]. Some properties of the system matrix (3.48) are induced by its continuous counterpart, i.e. the bilinear form $\mathtt{A}$ defined in (3.12). The coercivity and symmetry of $\mathtt{A}$ imply that

$$A \text{ is positive definite,} \tag{3.91}$$

and that

$$A \text{ is symmetric} \tag{3.92}$$

respectively. The finite element basis functions φ_k have local support. Hence

$$A \text{ is sparse.} \tag{3.93}$$

For problems without anti-periodic boundary conditions, we have that under suitable conditions of the triangulation detailed in e.g [89] that

$$A_{ii} > 0 \text{ and } A_{ij} \leq 0 \text{ if } i \neq j\,. \tag{3.94}$$

From (3.91) and (3.94) we have by Theorem 2.3 in [14] that

$$A \text{ is an M-matrix}. \tag{3.95}$$

In problems with anti-periodic boundary conditions, the treatment of the boundary conditions, leads to off-diagonal entries that are positive and of the same order of magnitude as the diagonal entries. Such matrices do not have the M-matrix property.

Under suitable conditions on the triangulation $\mathcal{T}_h$, one can show that

$$\mathrm{cond}_2(A) = \mathcal{O}(h^{-2})\,. \tag{3.96}$$

The condition number is not only affected by the size of the grid, but also by the shape of the domain, the quality of the triangulation and the smoothness of the material coefficients in the PDE.

3.5.3 Time-Harmonic System Matrix Properties

Let A be the coefficient matrix of the time-harmonic system (3.51). The relation (3.53) defining its real part A_R and the relation defining the entries of the coefficient matrix of stationary systems (3.49) are the same. Therefore A_R has all the properties a stationary system matrix has, i.e.

$$A_R \text{ is a sparse, symmetric and positive definite M-matrix}, \tag{3.97}$$

and

$$\text{cond}_2(A_R) = \mathcal{O}(h^{-2}). \tag{3.98}$$

For the imaginary part A_I, we have that the local support of the FE basis functions and the symmetry in the indices k and ℓ of the expression (3.54) imply that

$$A_I \text{ is sparse and symmetric}, \tag{3.99}$$

respectively. Next we prove that A_I is positive semi-definite.

Proposition 3.5.1 *Let λ be an eigenvalue of A_I: then $\lambda \geq 0$.*

Proof. The rows of the matrix A_I corresponding to nodes in the non-conducting parts of Ω are zero. Denoting the union of the conducting parts of Ω by Ω_σ, we obtain by ordering the nodes in Ω_σ first, the following block structure for A_I

$$A_I = \begin{pmatrix} \widetilde{A}_I & 0 \\ 0 & 0 \end{pmatrix}. \tag{3.100}$$

Let (λ, c) be an eigenvalue-eigenvector pair of the diagonal block $\widetilde{A}_I$, and consider the linear combination of FE basis functions defined through c as $v = \sum_i c_i \varphi_i$. The support of v lies in Ω_σ, and therefore

$$\lambda \parallel c \parallel^2 = c^T \widetilde{A}_I \, c = \omega \int_{\Omega_\sigma} \sigma \, v^2(\mathbf{x}) \, d\Omega \geq \omega \min_{\mathbf{x} \in \Omega_\sigma} \sigma(\mathbf{x}) \parallel v \parallel_0^2 > 0 \tag{3.101}$$

The proposition then follows from the fact that the eigenvalues of A_I are those of $\widetilde{A}_I$ and 0. □

For the entire matrix, (3.97) and (3.99) imply that

$$A \text{ is complex, sparse and symmetric}. \tag{3.102}$$

A is *not* Hermitian. We are able to prove that the spectrum of A lies in the first quadrant of the complex plane.

Proposition 3.5.2 *Let $\lambda = \lambda_R + j\,\lambda_I$ be an eigenvalue of A: then $\lambda_R > 0$ and $\lambda_I \geq 0$.*

Proof. Let (λ, c) be an eigenvalue-eigenvector pair of the matrix A. The symmetry of A_R and A_I implies that

$$\lambda \parallel c \parallel^2 = c_R^T A_R c_R + c_I^T A_R c_I + j[c_R^T A_I c_R + c_I^T A_I c_I]\,. \tag{3.103}$$

The proposition then follows from the fact that A_R is positive definite and A_I positive semi-definite. □

Remark In some cases more detailed information on the spectrum of A is available. Consider the model problem in which the PDE (2.43) is posed on $\Omega = [0,1] \times [0,1]$ discretized by a regular mesh, and in which σ is constant over Ω. In both the finite difference method and the finite element method with lumping (see e.g. [90]), the matrix A_I reduces to a constant diagonal matrix, i.e., $A_I = k\,I$. If (λ, c) is an eigenvalue-eigenvector pair of the matrix A_R, then $(\lambda + jk, c)$ is an eigenvalue-eigenvector pair of A. The spectrum of A is that of A_R shifted upward in the complex plane by a distance k.

As an illustration we plotted the spectrum of the matrix resulting from the discretization of a time-harmonic problem in Figure 3.1.

3.5.4 Coupled Field-Circuit System Matrix Properties

The field-circuit coupled system (3.69) has a block structure. The first set of equations represents the field-equations, while the second set represents the electrical circuit relations. The $(1,1)$-block of the system matrix $\mathcal{A}$ defined in (3.70) has all the properties that a matrix of the discrete harmonic system (3.51) has. Equation (2.72) implies the symmetry of the $(2,2)$-block of the system matrix $\mathcal{A}$. The matrix $\mathcal{A}$ is therefore symmetric. In practical applications the number of circuit relations (up to 500) will be much smaller than the number of FE degrees of freedom (up to 500.000). We therefore have

$$dim(S) \ll dim(A) \tag{3.104}$$

Numerical experiments demonstrate that the coupling with the electrical circuit introduces eigenvalues with possibly negative real or imaginary part.

As an illustration we plotted the spectrum of the matrix resulting from the discretization of a field-circuit coupled problem in Figure 3.2. This figure shows an eigenvalue with a negative imaginary part.

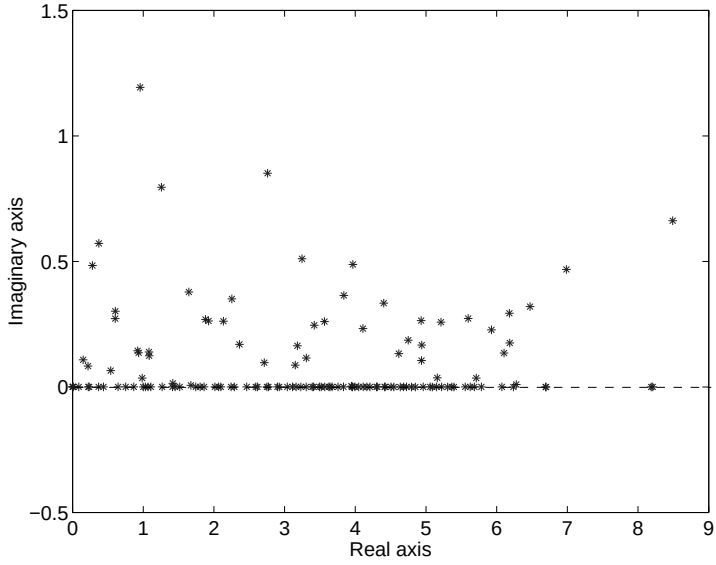


Figure 3.1: Spectrum of a matrix resulting from the discretization of a time-harmonic problem.

3.6 Software Implementation

A finite element simulation package called `Olympos` was developed by the ELEN/ESAT research group [86, 76, 35, 27]. The iterative solvers for the systems of linear equations developed in this study have been incorporated into that package. We therefore touch upon a number of features of the `Olympos` software package that influenced some of the choices made in this work.

3.6.1 Mesh Generation

To be able to model the complicated domains of realistic applications *unstructured* meshes of triangles are used in `Olympos`. These meshes are generated by *adaptive refinement* [79].

Given an initial triangulation of Ω, a finer mesh is constructed by inserting nodes in such a way that the finite element discretization error is decreased. A local error estimator uses the solution computed on the initial mesh to select those triangles in which some approximation to the discretization error is large. A refinement algorithm splits the elements with large error into smaller ones by adding nodes in the triangulation. The refine-

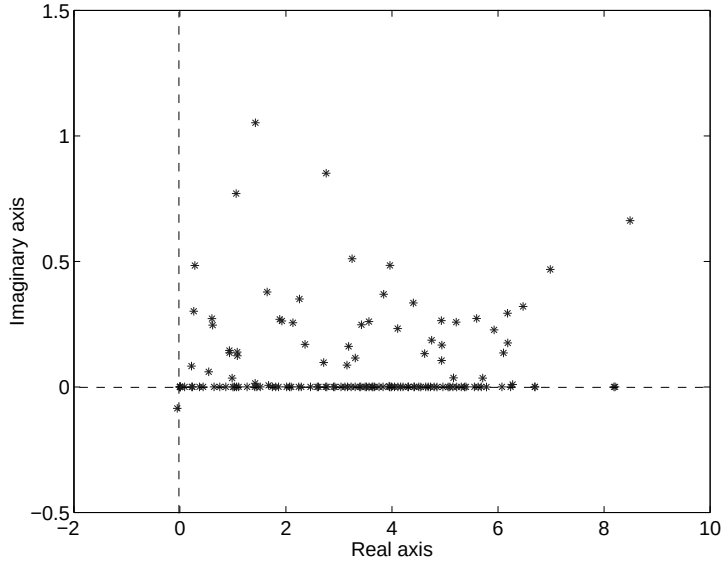


Figure 3.2: Spectrum of a matrix resulting from the discretization of a field-circuit coupled problem.

ment of elements is done by either inserting nodes in the interior of a triangle (element based refinement) or on the edges (edge based refinement). After insertion, nodes can be moved to improve mesh quality. The mesh refinement process is started with a minimal triangulation of the geometry and is repeated until the discretization error estimate obtained on the newly created triangulation drops below a prescribed tolerance.

The mesh construction process for the magnetic field computation in the vicinity of a configuration of high voltage wires is shown Figure 3.3 as an example. The upper three figures give a global view of the problem, while the lower three figures give a more detailed view of the triangulation close to the wires. The unboundedness of the domain is modeled by a conformal mapping as discussed in Section 2.12.2. The mapped unbounded domain is represented by the disk in the upper right corner in the first three figures.

The mesh refinement process in linear problems requires the solution of a sequence of linear systems, each corresponding to a different mesh. In non-linear problems a set of linear systems resulting from the Picard or Newton linearization has to be solved at each mesh refinement step.

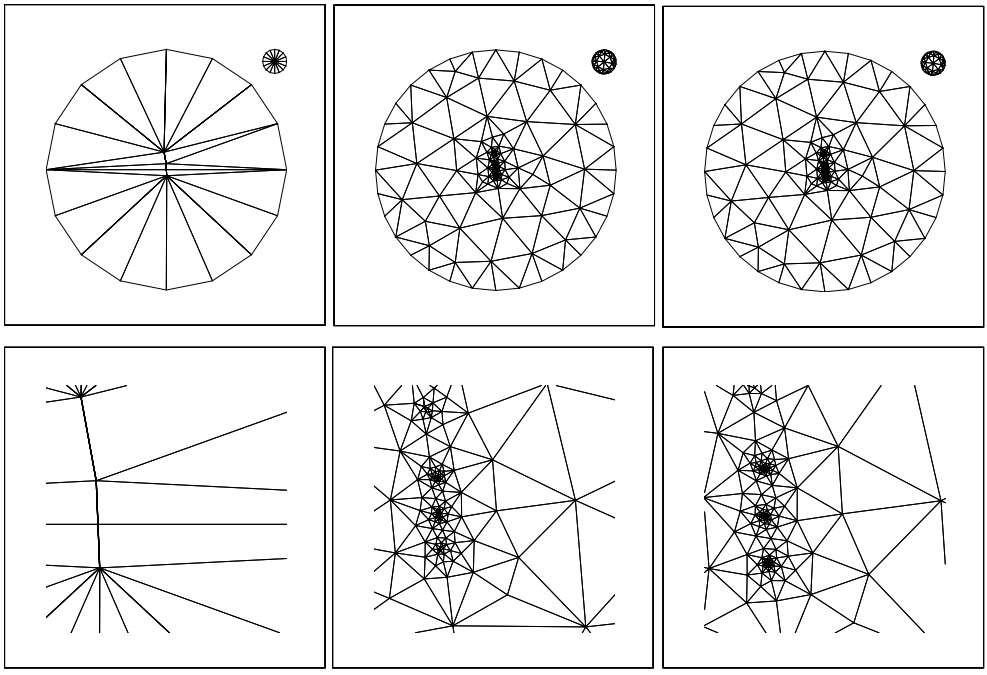


Figure 3.3: Illustration of the adaptive mesh construction process for the magnetic field computation in the vicinity of a configuration of high voltage wires. The upper three figures give a global view of the problem, while the lower three pictures give a more detailed view close to the wires. The mapped unbounded domain is represented by the disk in the upper right corner in the upper three figures.

3.6.2 Treatment of Non-Linearities

In `Olympos` the non-linear constitutive laws in stationary and time-harmonic formulations are treated by a Newton and Picard linearization respectively. On the coarsest mesh the linearization is started with a zero initial guess. On finer meshes an initial guess is obtained by projecting the solution computed on the next coarser mesh [78]. In every Newton or Picard iteration step, each linearised system is solved to full accuracy.

3.7 Example Problems

In this section we give some example problems that we will use as test cases in the following chapters.

3.7.1 Permanent Magnet Machine

In example problem **EP**-1, we consider the computation of the stationary magnetic field in the non-linear model for a permanent magnet machine. Figure 3.4 shows the geometry and the simulated equipotential lines of the solution. The shape and arrangement of the permanent magnets in the rotor is such that no symmetry is present in the model and no periodic boundary conditions of the type discussed in Section 2.12.1 can be used to reduce the model size. The mesh refinement process is started on a mesh with 1.092 nodes and 2.146 elements and results after 13 steps in a mesh with 174.140 nodes and 347.757 elements. Figure 3.6 reports the increase in the number of nodes during the mesh refinement process. The initial coarse mesh and some intermediate finer mesh are shown in Figure 3.5. The number of Newton steps required on each mesh is plotted in Figure 3.7. This figure illustrates that the number of non-linear steps on the initial grid is larger than on all finer grids. This is due to the fact that on all finer grids a better starting solution is available by the projection from the next coarser mesh.

3.7.2 Switched Reluctance Machine

In example problem **EP**-2 we look at the linear stationary magnetic field computation in a model where anti-periodic boundary conditions are present. As an example we take the quarter model of the switched reluctance machine used as an illustration in Section 2.12.1. In this problem six adaptive steps result in a mesh with 4141 nodes and 8064 elements. The mesh and the motor's geometry are plotted in Figure 3.8.

3.7.3 400 kW Induction Machine

In example problem **EP-3** we consider the linear time harmonic computation of a four pole 400 kW induction machine. The real and imaginary part of the computed solution are plotted in Figures 3.10 and 3.11 respectively. Thanks to symmetry of the geometry and the field, periodic boundary conditions can be used and only two poles have to be modeled. The solid rotor bars are driven by 0 V voltage sources (short circuited). The stator windings are excited by a three-phase alternating current system of 154 A. No external circuit relations are therefore necessary to model the electric load. The initial mesh consists of 1028 nodes and 1966 elements. After four refinement steps 75951 nodes and 151504 elements are obtained. Figure 3.9 shows the mesh obtained after two refinement steps. In Figure 3.12 we plotted the increase of the number of nodes during mesh refinement.

3.7.4 45 kW Induction Machine

In a fourth and last example, example problem **EP-4**, we take a four pole 45kW induction machine. Unlike in the previous example, we consider a full model. The final mesh was constructed after four adaptive refinement steps and contains a total of 118802 elements and 59574 nodes. The finite element model is coupled with an electrical circuit consisting of 148 equations. The equipotential lines of the real part of the computed magnetic vector potential are shown in Figure 3.13.

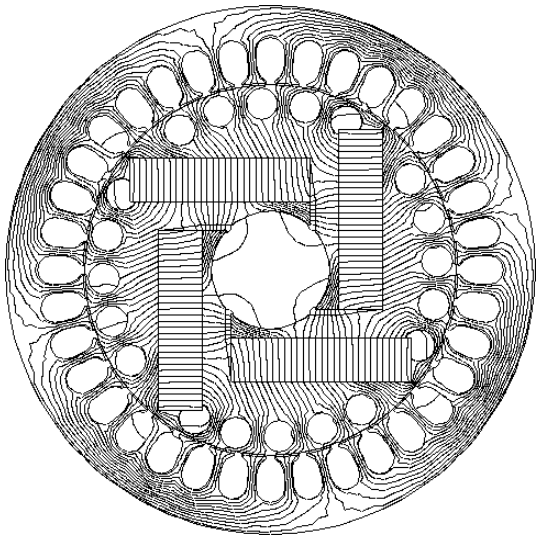


Figure 3.4: Computed equipotential lines of the vector potential in example problem **EP**-1.

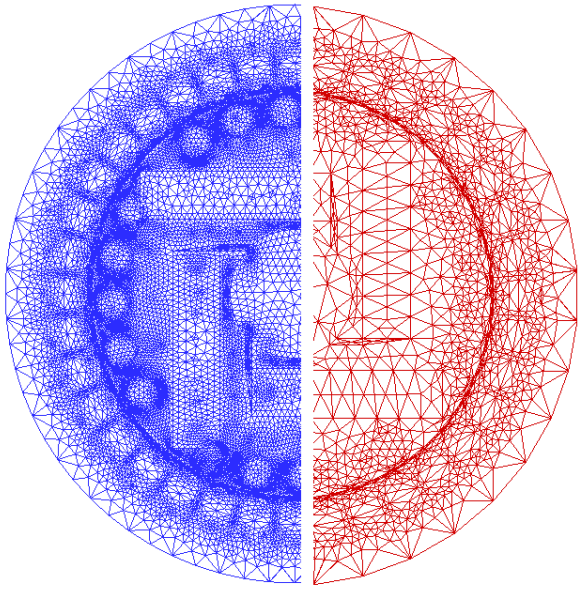


Figure 3.5: Coarse and fine mesh in example problem **EP**-1.

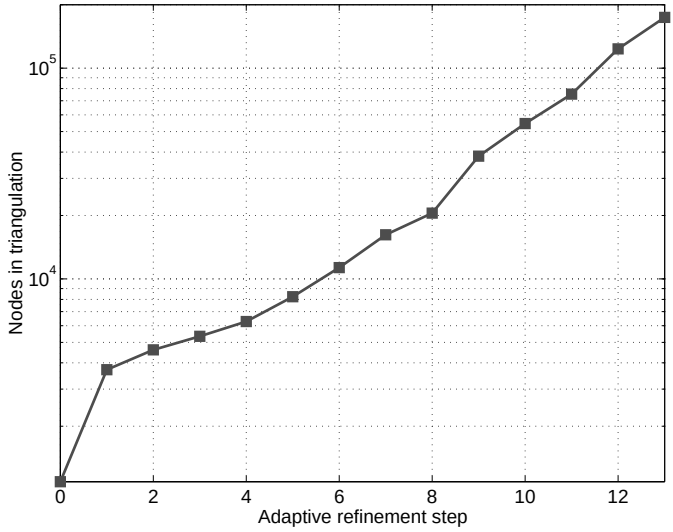


Figure 3.6: Number of nodes in triangulation versus the adaptive refinement step in example problem **EP**-1.

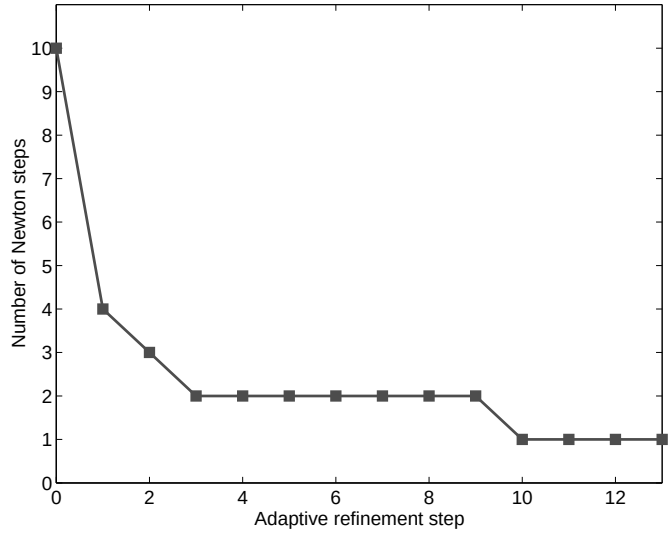


Figure 3.7: Number of Newton steps versus the adaptive refinement step in example problem **EP**-1.

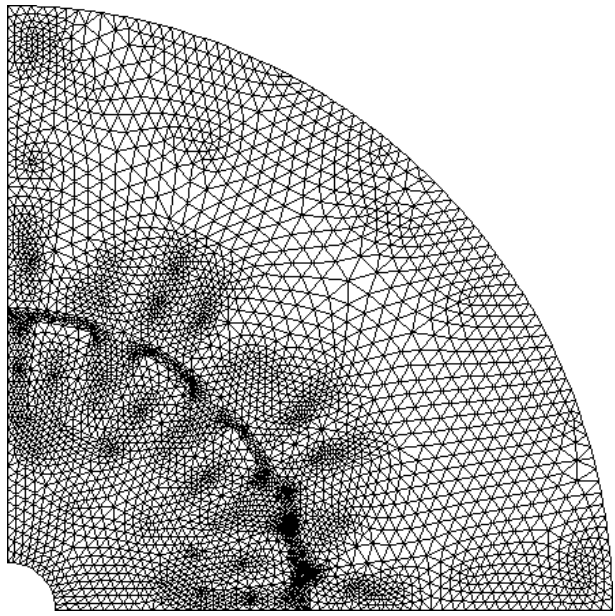


Figure 3.8: Final mesh obtained in example problem **EP**-2.

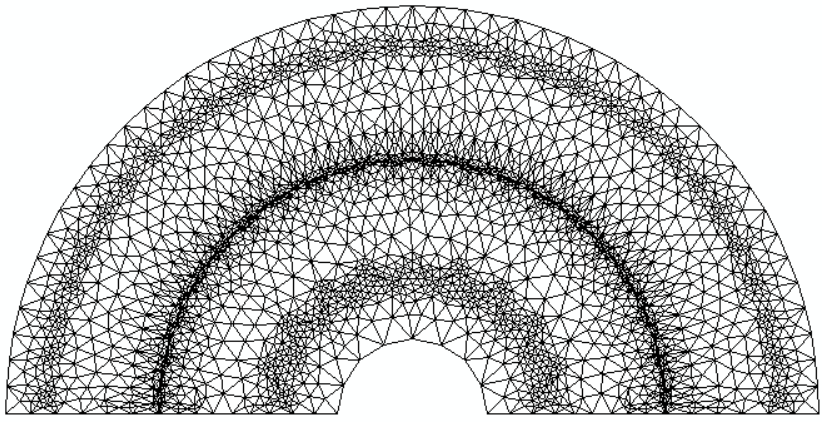


Figure 3.9: Mesh obtained after two refinement steps in example problem **EP**-3.

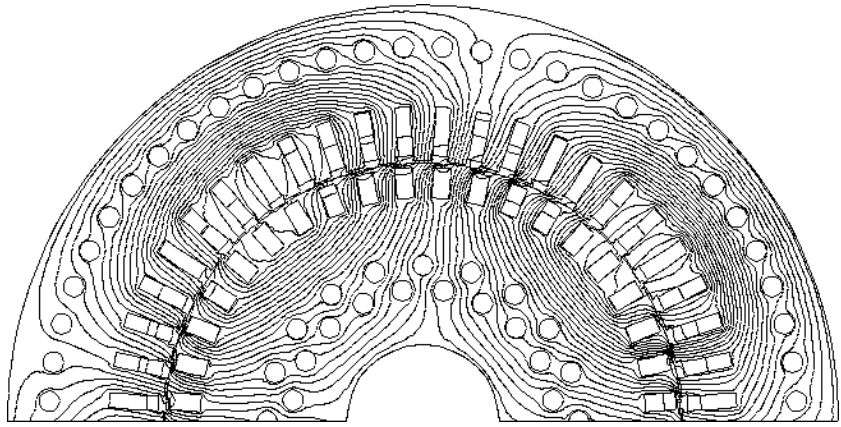


Figure 3.10: Equipotential lines of the real part of the vector potential in example problem **EP-3**.

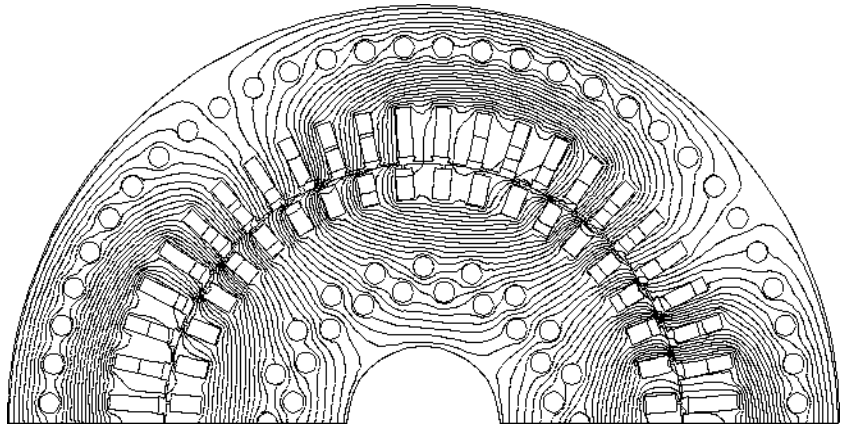


Figure 3.11: Equipotential lines of the imaginary part of the vector potential in example problem **EP-3**.

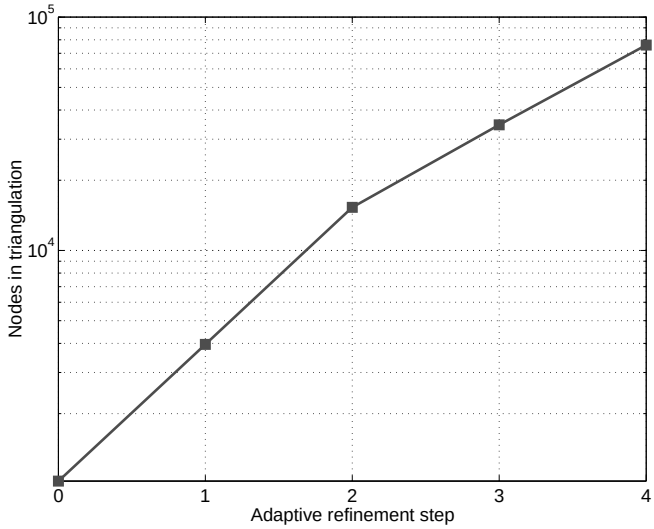


Figure 3.12: Number of nodes in triangulation versus the adaptive refinement step in example problem **EP**-3.

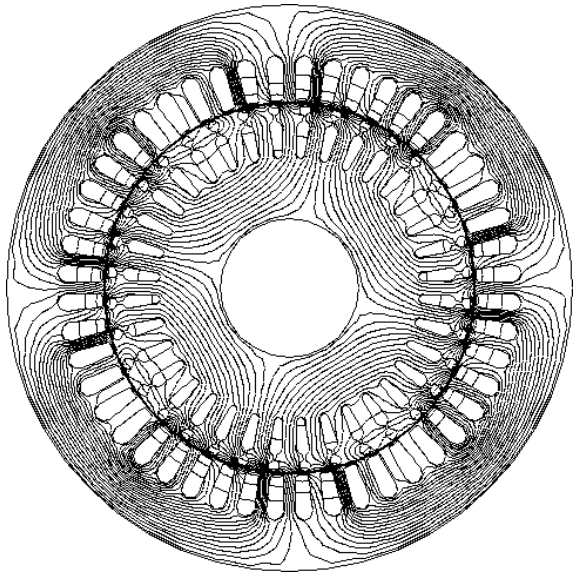


Figure 3.13: Equipotential lines of the real part of the computed vector potential in example problem **EP**-4.

Chapter 4

Basic Matrix Splitting and Krylov Subspace Methods

4.1 Introduction

In this chapter we discuss classes of iterative methods for solving the discrete systems obtained in Chapter 3. The choice for iterative instead of direct methods is motivated by the fact that the computational kernel of iterative methods is formed by the matrix-vector multiplication. For sparse matrices as those obtained in Chapter 3, the number of iterations required for this operation is only linear in the number of unknowns.

The methods we will discuss in this chapter operate on a single level of discretization. This is in contrast with the methods that will be described in Chapter 5 and Chapter 6. The methods in this chapter furthermore do not require any information about the geometry of the discretized problem.

We first describe the classical matrix splitting methods. Afterwards we focus on Krylov subspace methods which are non-stationary. We describe the principles underlying these methods and some commonly used examples. We draw attention to Krylov subspace methods for complex symmetric systems. The speed of convergence of the Krylov iteration is enhanced by combining it with a preconditioner. We give examples of some basic preconditioning schemes. This chapter is concluded with some numerical examples. They show that the performance of the above methods used by itself for solving engineering problems is rather poor. This will motivate the study of faster algorithms in subsequent chapters.

4.2 Matrix Splitting Methods

It this section we recall basic stationary iterative schemes for solving linear systems resulting from the finite difference or finite element discretization of elliptic PDEs [118, 124, 21, 100]. When used as stand-alone solvers, these schemes are often not efficient. However, they may serve as simple preconditioning schemes for Krylov subspace methods leading to significantly superior convergence, see Section 4.6, and as building blocks for more advanced multigrid methods, see Chapter 5.

Basic iterative schemes for solving the system

$$A\,x = b\,, \tag{4.1}$$

are based on splitting the coefficient matrix A into

$$A = M - N\,. \tag{4.2}$$

The matrix M should be non-singular, and the linear system

$$M\,x = y \tag{4.3}$$

should be solvable at low cost. Given the starting solution x_0, the splitting (4.2) induces for the following iterative scheme

$$x_{m+1} = M^{-1}\,N\,x_m + M^{-1}\,b\,, \tag{4.4}$$

or, equivalently, by setting N equal to $M - A$

$$x_{m+1} = x_m - M^{-1}\,(A\,x_m - b)\,. \tag{4.5}$$

In terms of the iteration error $e_m = x - x_m$ and the residual $r_m = b - A\,x_m$, (4.4) implies

$$e_{m+1} = (I - M^{-1}\,A)\,e_m \tag{4.6}$$

and

$$r_{m+1} = (I - A\,M^{-1})\,r_m\,. \tag{4.7}$$

The speed of convergence of the iterative scheme (4.6) is characterized by some norm or by the spectral radius of the iteration matrix $C = I - M^{-1}\,A$.

In order to give the matrix M in explicit form for certain classical iterative methods, it is useful to introduce the notation

$$A = D - L - U \tag{4.8}$$

where D, $-L$ and $-U$ denote the diagonal, the strictly lower and strictly upper triangular part of A respectively. Given a relaxation factor $\omega > 0$, the matrix M for the *Jacobi*, *Gauss-Seidel* and *Successive Overrelaxation* (SOR) methods is then given by

$$M_{JAC} = D\,, \tag{4.9}$$

$$M_{GS} = D - L\,, \quad \text{and} \tag{4.10}$$

$$M_{SOR} = \frac{1}{\omega}\,D - L \tag{4.11}$$

respectively. In the Gauss-Seidel and the SOR methods, the order in which the components of the iterand x_m are updated plays a role. A symmetric version of the SOR method is obtained by sweeping over the unknowns a second time in reversed order. The matrix M for the resulting *Symmetric* Successive Overrelaxation (SSOR) method is given by

$$M_{SSOR} = \frac{1}{\omega(2-\omega)}(D - \omega\,L)D^{-1}\,(D - \omega\,U)\,. \tag{4.12}$$

It is symmetric and positive definite if A is symmetric and positive definite.

The convergence of these methods is analyzed in detail in [118].

The basic iterative schemes described above update each component of the iterand separately. By grouping components of the iterand it is possible to update blocks of components simultaneously, giving rise to block versions of the above methods. Such blocks may correspond to the degrees of freedom associated with a part of the computational domain or, in problems of coupled scalar PDEs, to the set of degrees of freedom associated to a node in the mesh.

In the Gauss-Seidel method *coloring* can be introduced to enable the parallel update of blocks of components.

4.3 Krylov Subspace Methods

In the remainder of this chapter we will discuss a family of iterative methods that are more robust and, if used in combination with a preconditioner, generally faster to converge than those discussed in the previous section. The methods produce iterands in a nested sequence of Krylov subspaces of increasing dimension and are called Krylov subspace methods. General references on this subject include the overview papers [45, 40] and the books [46, 21, 64, 49, 100].

Unfortunately, there is no single Krylov subspace method efficient (with respect to convergence rate and memory usage) for a general class of problems [80]. The choice of the particular method is problem dependent. This motivates the study of these methods for particular problems.

In the following we explain how Krylov subspace methods compute the consecutive iterands. An overview of the methods that focuses on those used in this thesis is given in the following section.

Definition 4.3.1 *Given a (real or complex) matrix A and a vector v, the Krylov subspace of order m generated by A and v is defined as*

$$K^m(A, v) = span\{v, A\,v, \ldots, A^{m-1}\,v\}\,. \tag{4.13}$$

The dimension of $K^m(A, v)$ equals m, unless there exists an integer $\mu < m$ such that the space $K^\mu(A, v)$ is invariant under A.

Given an initial guess x_0 and the corresponding residual $r_0 = b - A\,x_0$, Krylov subspace methods for solving the linear system (4.1) compute iterands x_m, $m \geq 1$, of the form

$$x_m \in x_0 + K^m(A, r_0)\,. \tag{4.14}$$

Given a basis $\{v_1, \ldots, v_m\}$ for $K^m(A, v)$, the matrix V_m defined as

$$V_m = [v_1, \ldots, v_m]\,, \tag{4.15}$$

and a vector of coefficients $y_m \in \mathbb{C}^m$, condition (4.14) can be expressed as

$$x_m = x_0 + V_m\,y_m\,. \tag{4.16}$$

In terms of the residual r_m this implies

$$r_m = b - A\,x_m = r_0 - A\,V_m\,y_m. \tag{4.17}$$

Krylov subspace methods differ in the way they construct the basis V_m and in the way they determine the coefficients y_m. We elaborate on these two issues in the following two subsections.

4.3.1 Krylov Subspace Basis Construction

The two most common methods for constructing a basis for $K^m(A, r_0)$ are the *Arnoldi* and the *bi-orthogonal Lanczos* method. Both methods iterate over the dimension of the Krylov subspace. For Hermitian matrices both methods simplify to the *Lanczos* method .

The Arnoldi Method

The Arnoldi iteration is started using as initial vector

$$v_1 = r_0 \,/\, \| r_0 \| \,. \tag{4.18}$$

Let $\{v_1, \ldots, v_m\}$ be an orthonormal basis of $K^m(A, r_0)$ obtained after applying m steps of the Arnoldi method. The orthonormality can be expressed in terms of the matrix $V_m = [v_1, \ldots, v_m]$ as

$$V_m^H \, V_m = I_m \,. \tag{4.19}$$

The Arnoldi method expands this basis to an orthonormal basis of the space $K^{m+1}(A, r_0)$ by computing $A\, v_m$ and orthogonalizing this vector with respect to $\{v_1, \ldots, v_m\}$. Let the resulting vector be denoted by $\tilde{v}_m$. If the orthogonalization is carried out by a standard Gram-Schmidt procedure, we have that

$$\tilde{v}_m = A\, v_m - \sum_{i=1}^{m} h_{i,m}\, v_i \,, \tag{4.20}$$

where the coefficient $h_{i,m}$ are given by

$$h_{i,m} = v_i^H \, A\, v_m \text{ for } 1 \le i \le m \,. \tag{4.21}$$

The new basis vector v_{m+1} is then obtained by normalizing $\tilde{v}_m$

$$v_{m+1} = \tilde{v}_m \,/\, h_{m+1,m} \text{ where}\,, \tag{4.22}$$

$$h_{m+1,m} = \| \tilde{v}_m \| \,. \tag{4.23}$$

Let $H_{m,m}$ be the $m \times m$ upper Hessenberg matrix with elements $h_{i,j}$ defined by (4.21)-(4.23), and let $\widetilde{H}_{m+1,m}$ be its extension by the additional row $[\,0 \ldots 0\, h_{m+1\,m}\,]$ of length m, i.e.,

$$\widetilde{H}_{m+1,m} = \begin{pmatrix} H_{m,m} \\ 0 \ldots 0\, h_{m+1,m} \end{pmatrix} \,. \tag{4.24}$$

Using this matrix, the recursion for the Arnoldi basis vectors can be written in matrix-vector form as

$$\begin{aligned} A\, V_m &= V_m \, H_{m,m} + h_{m+1,m}\, v_{m+1}\, e_m^H \\ &= V_{m+1}\, \widetilde{H}_{m+1,m} \end{aligned} \tag{4.25}$$

where e_m^H denotes the row vector $[\,0, \ldots, 0, 1\,]$ of length m.

In the Arnoldi method the basis vector v_{m+1} is constructed by orthogonalizing with respect to all previous vectors $\{v_1, \ldots, v_m\}$, resulting in *long* recurrences. The computational cost and the memory requirements of these long term recurrences grow with m. All basis vectors need to be stored in memory.

By multiplying (4.25) to the left by V_m^H one obtains by the orthonormality of V_{m+1} the relation

$$V_m^H \, A \, V_m = H_{m,m} \, . \tag{4.26}$$

This implies that the matrix $H_{m,m}$ can be interpreted as the orthogonal projection of A onto the vector space $K^m(A, r_0)$. The eigenvalues of $H_{m,m}$ are called *Ritz* values and provide an approximation to the eigenvalues of A.

The Lanczos Method

Relation (4.26) implies that if A is Hermitian, the Hessenberg matrix $H_{m,m}$ is Hermitian as well, and therefore tridiagonal. We denote the matrices $H_{m,m}$ and $\widetilde{H}_{m+1,m}$ defined in (4.24) in the case of Hermitian problems by T_m and $\widetilde{T}_{m+1}$ respectively. The tridiagonal structure of T_m implies that the recurrence relation for the basis vectors V_{m+1} of $K^{m+1}(A, r_0)$, namely

$$\begin{aligned} A \, V_m &= V_m \, T_m + \| \, \tilde{v}_m \, \| \, v_{m+1} \, e_m^H \\ &= V_{m+1} \, \widetilde{T}_{m+1} \end{aligned} \tag{4.27}$$

are *short* term recurrences. The version of the Arnoldi method for Hermitian problems is called the Lanczos method.

The Bi-Orthogonal Lanczos Method

If the matrix A is non-Hermitian, computationally attractive *short* term recurrences for generating a basis V_m for $K^m(A, r_0)$ can be obtained by making the orthogonality conditions less stringent. This can be done by making V_m orthogonal to a basis W_m of a second Krylov subspace L^m of dimension m. In the bi-orthogonal Lanczos method the space L_m is the Krylov subspace generated by A^H and some vector r_0^*

$$L^m = K^m(A^H, r_0^*) \, . \tag{4.28}$$

In the bi-orthogonal Lanczos method the basis vectors can be scaled in various ways. Here we follow the conventions adapted in [43] and assume the basis vectors V_m and W_m to be normalized in such a way that

$$\| \, v_i \, \| = \| \, w_i \, \| = 1 \qquad 1 \leq i \leq m \, . \tag{4.29}$$

The bi-orthogonality condition can be expressed as

$$V_m^H \, W_m = D_m = \mathrm{diag}\,[d_1, \ldots, d_m]\,, \tag{4.30}$$

where

$$d_i = v_i^H \, w_i \qquad 1 \le i \le m\,. \tag{4.31}$$

The bi-orthogonal Lanczos recursion is started with the vector

$$v_1 = r_0 \,/\, \| \, r_0 \, \|\,, \tag{4.32}$$

and a vector w_1 with unit norm satisfying $v_1^H \, w_1 \neq 0$. The bases V_m and W_m are extended from step m to $m+1$ by computing basis vectors v_{m+1} and w_{m+1} using the following recurrences

$$\begin{aligned} \tilde{v}_{m+1} &= A \, v_m - \mu_m \, v_m - \nu_m \, v_{m-1}\,, \\ \rho_{m+1} &= \| \, \tilde{v}_{m+1} \, \|\,, \qquad v_{m+1} = \tilde{v}_{m+1} \,/\, \rho_{m+1}\,, \end{aligned} \tag{4.33}$$

$$\begin{aligned} \tilde{w}_{m+1} &= A^H \, w_m - \mu_m \, w_m - \frac{\rho_m}{\xi_m} \, \nu_m \, w_{m-1}\,, \\ \xi_{m+1} &= \| \, \tilde{w}_{m+1} \, \|\,, \qquad w_{m+1} = \tilde{w}_{m+1} \,/\, \xi_{m+1}\,, \end{aligned} \tag{4.34}$$

where we have introduced

$$\mu_m = \frac{w_m^H \, A \, v_m}{w_m^H \, v_m} \qquad \text{and} \qquad \nu_m = \frac{w_m^H \, v_m}{w_{m-1}^H \, v_{m-1}} \, \xi_m\,. \tag{4.35}$$

The coefficients defined by the bi-orthogonal Lanczos method are defined to be the elements of the $m \times m$ tridiagonal matrix

$$T_m = \mathrm{tridiag}[\, \rho_{i-1} \; \mu_i \; \nu_{i-1} \,] \qquad 1 \le i \le m\,, \tag{4.36}$$

its extension $\widetilde{T}_{m+1}$ by the $m+1$-st row $[\,0, \ldots, 0, \rho_{m+1}\,]$ as in (4.24), and the diagonal matrix

$$\Gamma_m = \mathrm{diag}[\, \gamma_1, \ldots, \gamma_m \,]\,, \tag{4.37}$$

where

$$\gamma_i = \begin{cases} 1 & \text{if} \quad i = 1 \\ \gamma_{i-1} \, \rho_i \,/\, \xi_i & \text{if} \quad i > 1 \end{cases}\,. \tag{4.38}$$

Using these matrices, the bi-orthogonal Lanczos recurrences (4.33)-(4.34) can be written in matrix-vector form as

$$\begin{aligned} A \, V_m &= V_{m+1} \, \widetilde{T}_{m+1} \\ &= V_m \, T_m + \rho_{m+1} \, v_{m+1} \, e_m^H \end{aligned} \tag{4.39}$$

$$\begin{aligned} A^H W_m &= W_{m+1}\,\Gamma_{m+1}^{-1}\,\widetilde{T}_{m+1}\,\Gamma_m \\ &= W_m\,\Gamma_m^{-1}\,T_m\,\Gamma_m + \xi_{m+1}\,w_{m+1}\,e_m^H\,. \end{aligned} \tag{4.40}$$

The tridiagonal structure of $\widetilde{T}_{m+1}$ implies that (4.39) and (4.40) are short recurrences. The bi-orthogonal Lanczos projection of the matrix A onto the space $K^m(A, r_0)$ is obtained by multiplying (4.39) to the left by W_m^H. By the bi-orthogonality of V_{m+1} and W_{m+1}, one obtains

$$W_m^H\,A\,V_m = D_m^H\,T_m\,. \tag{4.41}$$

The matrix T_m in (4.41) can be interpreted as an oblique projection of A onto $K^m(A, r_0)$.

Each step of the bi-orthogonal Lanczos algorithm requires a matrix-vector multiplication with both A and A^H. The algorithm breaks down if non-zero vectors v_i and w_i are generated for which $v_i^H\,w_i = 0$.

4.3.2 Computation of Iterands

Once the basis V_m of $K^m(A, r_0)$ has been constructed, the coefficients y_m in (4.16) can be computed by either a residual *projection* or a residual *norm minimization* approach.

Residual Projection Methods

If the basis V_m is orthonormal (as in Arnoldi's method), the coefficients y_m can be computed by the Ritz-Galerkin condition : $r_m \perp K^m(A, r_0)$, i.e.,

$$V_m^H\,r_m = 0\,. \tag{4.42}$$

By eliminating r_m using (4.17), one obtains

$$V_m^H\,A\,V_m\,y_m = V_m^H\,r_0\,, \tag{4.43}$$

which by (4.26) and (4.18) can be written as

$$H_{m,m}\,y_m = \beta_1\,e_1\,, \tag{4.44}$$

where $\beta_1 = r_0/\parallel r_0 \parallel$ and e_1 is the first unit vector of size m. One obtains a linear system of size equal to the dimension of $K^m(A, r_0)$ for y_m. Such a system has to be solved at each iteration. For Hermitian problems the matrix $H_{m,m}$ can be replaced by T_m defined in (4.27)

$$T_m\,y_m = \beta_1\,e_1\,. \tag{4.45}$$

If the basis V_m is bi-orthogonal to the basis W_m (as in the bi-orthogonal Lanczos method), the coefficients y_m can be computed by the Petrov-Galerkin condition: $r_m \perp L^m(A^H, r_0^*)$, i.e.,

$$W_m^H \, r_m = 0 \, . \tag{4.46}$$

By an analogous reasoning as above, one obtains the linear system (4.45) for y_m, with matrix T_m defined in (4.39).

Residual Norm Minimization Methods

In the minimal residual norm approach, the coefficients y_m are determined such that the Euclidean norm $\| r_m \|$ is minimal.

If the basis V_m is orthonormal, then by (4.17), (4.18) and (4.25) the residual norm minimization is equivalent to determining y_m such that

$$\begin{aligned} \| r_m \| &= \| V_{m+1} \, (\beta_1 \, e_1 - \widetilde{H}_{m+1,m} \, y_m) \| \\ &= min_{y \in \mathbb{C}^m} \, \| V_{m+1} \, (\beta_1 \, e_1 - \widetilde{H}_{m+1,m} \, y) \| \, , \end{aligned} \tag{4.47}$$

where the unit vector e_1 is of size $m+1$. As V_{m+1} is orthogonal, (4.47) is equivalent to

$$\| r_m \| = min_{y \in \mathbb{C}^n} \, \| \beta_1 \, e_1 - \widetilde{H}_{m+1,m} \, y \| \, . \tag{4.48}$$

The coefficients y_m are obtained by solving the $(m+1) \times m$ linear system

$$\widetilde{H}_{m+1,m} \, y_m = \beta_1 \, e_1 \tag{4.49}$$

in a least squares sense.

If the basis V_m is bi-orthogonal, we obtain by an analogous reasoning as above that the minimal residual criterion entails determining y_m such that

$$\| r_m \| = min_{y \in \mathbb{C}^m} \, \| V_{m+1} \, (\beta_1 \, e_1 - \widetilde{T}_{m+1} \, y) \| \, . \tag{4.50}$$

Solving the least-squares problem in the right-hand side of (4.50) results in an algorithm in which the memory requirements grow linearly with m due to the required storage of all basis vectors. The computational cost of solving (4.50) grows even faster than linearly with m. To avoid this, the true minimization can be replaced by the following *quasi*-minimization in which y_m is determined such that

$$\| r_m \| \approx min_{y \in \mathbb{C}^n} \, \| \beta_1 \, e_1 - \widetilde{T}_{m+1} \, y \| \, . \tag{4.51}$$

The coefficients y_m are obtained by solving the linear system

$$\widetilde{T}_{m+1} \, y_m = \beta_1 \, e_1 \, , \tag{4.52}$$

in a least squares sense. The tridiagonal structure of $\widetilde{T}_{m+1}$ allows to construct an algorithm in which the coefficients y_m in (4.52) are computed by a three terms recursion. As opposed to solving (4.50), the computational cost of each step of this algorithm is independent of m.

4.4 Overview of Krylov Subspace Methods

4.4.1 Conjugate Gradient Method

Historically the first Krylov subspace method to be developed was the Conjugate Gradient (CG) method [58] .

The CG method is suitable for solving linear systems with real, symmetric positive definite matrices A and combines the Lanczos method (4.27) to generate a basis for the Krylov subspace with the Ritz-Galerkin (4.42) condition to determine the coefficients y_m. Relation (4.26) with $H_{m,m}$ replaced by T_m implies that the matrix T_m is symmetric positive definite at every iteration step m if A is symmetric positive definite. This allows to solve the system (4.44) by LU decomposition without pivoting. This LU decomposition can be updated progressively and rewritten as a coupled two-terms recursion for the residual and the so-called search direction vectors. The resulting algorithm requires per iteration one matrix-vector product involving the matrix A, three vector updates and two vector inner products.

The residual vector r_m and the search direction p_m at iteration step m can be expressed as

$$r_m = \psi_m(A)\, r_0 \qquad \text{and} \qquad p_m = \pi_m(A)\, r_0 \tag{4.53}$$

where ψ_m and π_m are certain polynomials of degree m. The residual polynomial satisfies the additional constraint $\psi_m(0) = 1$.

The convergence rate of the CG algorithm increases during the iteration. The CG algorithm converges super linearly. Using Chebyshev polynomials of the first kind and results from approximation theory, the number of CG iterations required for a prescribed error-norm reduction can be bounded in terms of the spectral condition number $\mathrm{cond}_2(A)$ (3.88) of the matrix in the following way

$$\| x - x_m \|_A \leq 2 \left[\frac{\sqrt{\mathrm{cond}_2(A)} - 1}{\sqrt{\mathrm{cond}_2(A)} + 1} \right]^m \| x - x_0 \|_A \ . \tag{4.54}$$

This bound is a pessimistic one in cases where the spectrum of A consists of a cluster of eigenvalues surrounded by a few outliers. More detailed information on the convergence of the CG algorithm can be gained by investigating the convergence of the Ritz values towards the eigenvalues as

in [112]. Similar convergence behavior is seen for other Krylov subspace methods as well.

4.4.2 Minimal Residual Method

For real symmetric *indefinite* matrices the CG algorithm is no longer appropriate. On its way to converge to a negative eigenvalue, a Ritz value may become zero and the matrix T_m in (4.27) singular.

In the Minimal Residual (MINRES) algorithm [87] this problem is solved by replacing the residual projection by a residual norm minimization. By induction on m it can be shown that the matrix $\widetilde{T}_{m+1}$ in (4.27) has full column rank. The corresponding least-squares problem (4.49) (with $\widetilde{H}_{m+1,m}$ replaced by $\widetilde{T}_{m+1}$) is solved by rotating $\widetilde{T}_{m+1}$ to upper triangular form by a series of Givens rotations. In the resulting algorithm, the tridiagonal structure of $\widetilde{T}_{m+1}$ allows to update the residual from one iteration to the next by a three-terms recursion.

The linear systems that the CG and MINRES algorithms solve are the same up to the additional row in $\widetilde{T}_{m+1}$. This information can be used to compare the convergence of both methods. In the absence of roundoff errors, the number of iterations needed to reach convergence is about the same for both methods. In finite precision arithmetic however three-terms recursions are less robust than the mathematically equivalent coupled two-terms recurrences [52, 43]. Compared with the CG algorithm, the MINRES algorithm therefore is more easily subject to round-off error problems.

As an illustration we plotted in Figure 4.1 the convergence histories of the CG and the MINRES algorithm in solving the system on the initial mesh and in the first Newton step in example problem **EP**-1 described in Section 3.7.1. As the MINRES algorithm minimizes the residual in each iteration, its convergence history is more smooth. Both methods require the same number of iterations to reduce the residual norm up to about 10^{-11}. To reach lower tolerances, the MINRES algorithm requires more iterations.

4.4.3 Full Orthogonalization and Generalized Minimal Residual Methods

For solving large sparse non-Hermitian systems, the Arnoldi algorithm can be combined with either a Ritz-Galerkin or a minimal residual criterion. The first alternative yields the Full Orthogonalization Method (FOM) [97], while the second yields the Generalized Minimal Residual (GMRES) method [101]. In the GMRES method the least squares problem is solved by Givens rotations. As for CG and MINRES, the linear systems that FOM and GMRES solve in the m-th iteration to determine y_m differ in the $m+1$-st

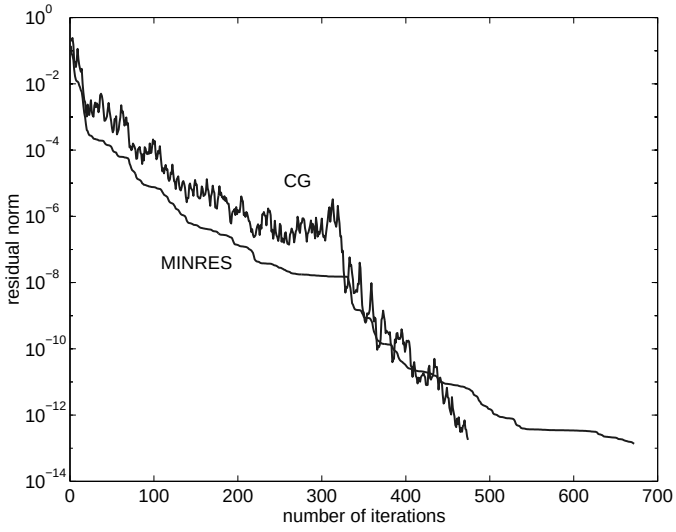


Figure 4.1: Convergence history of the CG and MINRES algorithms in solving the system on the initial mesh in the first Newton step in example problem **EP**-1.

row only. This fact can be exploited to prove that both algorithms require about the same number of iterations to reach convergence.

In order to avoid excessive memory requirements and floating point operations during the Arnoldi process, FOM and GMRES can be restarted after m iterations. At the $m+1$-st iteration the initial residual is equal to r_m and the iterative process is started anew. The resulting methods are denoted by FOM(m) and GMRES(m). Restarting FOM or GMRES causes a severe loss of speed of convergence. Other strategies to truncate the Krylov subspace have therefore been developed (see e.g. [29]).

4.4.4 BiConjugate Gradient Method

Another approach for solving non-Hermitian systems consists in combining the bi-Lanczos method with a Petrov-Galerkin condition. The resulting method is the BiConjugate Gradient (BiCG) method [38]. Its advantage is its use of short term recurrences.

The method breaks down however if the linear system determining the coefficient y_m becomes rank deficient. To distinguish this breakdown from the breakdown in the construction of the bi-orthogonal basis, the latter and former are called breakdown of the first kind and of the second kind

respectively. To overcome these breakdowns, a *look-ahead* strategy [42, 41, 20] needs to be incorporated into the algorithm.

The convergence of the BiCG algorithm can be very irregular in the sense that very large intermediate residuals can occur. This irregular convergence is related to the system determining the coefficient y_m being badly conditioned and to the near breakdown of the algorithm. The residual peaks adversely affect the convergence of the algorithm [103]. During the iteration true and updated residuals drift apart and the final computed solution may have very little significance.

The BiCG algorithm implicitly solves a second linear system involving the matrix A^H together with the linear system involving the matrix A. Unless one is interested in the solution of the former system, the operations involved in solving it are essentially wasted. The application of the BiCG algorithm is furthermore restricted to problems where the conjugate transpose of the matrix is readily available.

4.4.5 Conjugate Gradient Squared

In terms of the residual and search direction polynomials ψ_m and π_m introduced in (4.53), the coefficients (r_m, r_m^*) and $(A\,p_m, p_m^*)$ required in the BiCG algorithm can be written as

$$(r_m, r_m^*) = (\psi_m(A)r_0, \psi_m(A^H)r_0^*) \tag{4.55}$$

and

$$(A\,p_m, p_m^*) = (A\,\pi_m(A)r_0, \pi_m(A^H)r_0^*) \tag{4.56}$$

respectively. Redundant operations related to solving the A^H linear system can be recycled into more useful work by replacing these expressions with

$$(r_m, r_m^*) = (\psi_m^2(A)r_0, r_0^*) \tag{4.57}$$

and

$$(A\,p_m, p_m^*) = (A\,\pi_m^2(A)r_0, r_0^*)\,. \tag{4.58}$$

This gives rise to the Conjugate Gradient Squared (CGS) algorithm [106], an algorithm in which the CG polynomials defining the residual and the search directions are squared. It has been observed that in many practical computations the CGS algorithm converges twice as fast as the BiCG algorithm (see e.g. [8, 22]). The former does however suffer from the same irregular convergence and breakdown situations as the latter.

4.4.6 BiConjugate Gradient Stabilized

The BiCGSTAB algorithm [113] was developed with the aim of reducing the numerical instabilities of the CGS algorithm. The iteration

$$r_m = \psi_m^2(A) r_0 \,, \tag{4.59}$$

is replaced by

$$r_m = \phi_m(A)\psi_m(A) r_0 \,, \tag{4.60}$$

where $\psi_m(t)$ is the polynomial associated with the BiCG algorithm and where $\phi_m(t)$ is a recursively defined polynomial constructed with the aim to smooth the convergence history of the CGS algorithm. Due to its smoother convergence behavior, the BiCGSTAB algorithm produces more accurate solutions and in many cases converges faster than the CGS algorithm.

4.5 Methods for Complex Symmetric Systems

Linear systems with a complex symmetric coefficient matrix can be solved by Krylov subspace methods for non-Hermitian matrices. These methods do not exploit the symmetry of the problem. The symmetry can be exploited by building Krylov subspace methods based on the bi-orthogonal Lanczos method for complex symmetric matrices. A third alternative for complex symmetric systems consists in rewriting the complex system as an equivalent real one of double dimension. Complex arithmetic is then avoided.

4.5.1 The Complex Symmetric Bi-Orthogonal Lanczos Method

For complex symmetric matrices, a convenient choice for L^m in (4.28) is

$$L_m = \overline{K^m}(A, r_0) = \{\overline{v} \,|\, v \in K^m(A, r_0)\} \tag{4.61}$$

where $\overline{v}$ denotes the complex conjugate of v. With this choice, only one sequence of basis vectors needs to be computed as W_m can be set equal to

$$W_m = \overline{V}_m \,, \tag{4.62}$$

and only one matrix-vector multiplication per bi-Lanczos step is required. Indeed, given a basis vector w_m, the product

$$A^H w_m = \overline{A}\,\overline{v}_m \,, \tag{4.63}$$

is readily available. Given complex-valued vectors v and w, we denote by $<,>$ the true inner product

$$< v, w >= v^H w\,, \tag{4.64}$$

and by $<,>_T$ the bilinear form

$$< v, w >_T = v^T w\,. \tag{4.65}$$

For the latter so-called quasi-null vectors exist, i.e., vectors v for which

$$< v, v >_T = 0 \quad \text{even though} \quad v \neq 0\,. \tag{4.66}$$

Relations (4.30) and (4.62) imply that the vectors V_m are $<,>_T$ orthogonal

$$V_m^T V_m = D_m\,. \tag{4.67}$$

The recurrence relation for these vectors follows from (4.39) and is given by

$$A\,V_m = V_{m+1}\,\widetilde{T}_{m+1} \tag{4.68}$$

The resulting bi-orthogonal Lanczos algorithm for complex symmetric matrices coincides with the Lanczos algorithm for Hermitian matrices with the inner product (4.64) replaced by the bilinear form (4.65) [39]. It breaks down if a newly generated basis vector is a quasi-null vector.

4.5.2 Complex Orthogonal Conjugate Gradient and Symmetric Quasi Mimimal Residual Method

Krylov subspace solvers constructed by the bi-orthogonal Lanczos method for complex symmetric matrices can be combined with either the Petrov-Galerkin condition

$$V_m^T\,r_m = 0\,, \tag{4.69}$$

or the quasi-minimal residual condition

$$\| r_m \| \approx min_{y \in \mathbb{C}^m} \| \beta_1\, e_1 - \widetilde{T}_{m+1}\, y \|\,, \tag{4.70}$$

where is $\widetilde{T}_{m+1}$ the tridiagonal matrix defined in (4.68). The first alternative yields the algorithm called Conjugate Orthogonal CG (COCG) in [114] and BiCG in [39]. The second alternative yields the Symmetric Quasi Minimal Residual (SQMR) algorithm [39].

Both the COCG and the SQMR algorithms are short recurrence methods that require only one matrix vector multiplication per iteration. Per

iteration they are therefore cheaper than Krylov subspace methods for general non-Hermitian systems.

The COCG algorithm is straightforward to implement. All that needs to be done is to replace the inner product (4.64) by the bilinear form (4.65) in a given CG implementation. Like the BiCG algorithm, the COCG algorithm is susceptible to breakdown of the first and second kind and suffers from an irregular convergence behavior. The breakdown of the first kind is caused by the introduction of quasi-null basis vectors.

As in the MINRES algorithm it can be shown that the matrix $\widetilde{T}_{m+1}$ in (4.70) has full column rank. When the recursion for the residual vectors is implemented as a three terms recurrence, the SQMR algorithm is only susceptible to breakdown of the first kind. To ensure robustness however, it is preferable to implement the SQMR algorithm by coupled two terms recurrences. This can be done by LU-factoring $\widetilde{T}_{m+1}$ prior to the least squares solve [43]. By doing so, one again introduces the possibility of breakdown of the second kind.

As for the CG/MINRES and the FOM/GMRES pairs, the number of iterations that the COCG and SQMR algorithms require to converge is about the same. The converge history of SQMR is smoother than that of COCG.

4.5.3 Equivalent Real Formulations

Given a complex matrix or vector, we denote its real and imaginary components using the subscripts R and I respectively. The complex system

$$A\,x = b\,, \tag{4.71}$$

can be rewritten as a real system of double dimension for the unknowns $(x_R\, x_I)^T$. Two equivalent real formulations exist [39]: a non-symmetric one

$$A_* \begin{pmatrix} x_R \\ x_I \end{pmatrix} = \begin{pmatrix} b_R \\ b_I \end{pmatrix} \qquad \text{where} \qquad A_* = \begin{pmatrix} A_R & -A_I \\ A_I & A_R \end{pmatrix} \tag{4.72}$$

and a symmetric variant

$$A_{**} \begin{pmatrix} x_R \\ -x_I \end{pmatrix} = \begin{pmatrix} b_R \\ b_I \end{pmatrix} \qquad \text{where} \qquad A_{**} = \begin{pmatrix} A_R & A_I \\ A_I & -A_R \end{pmatrix}. \tag{4.73}$$

The systems (4.72) and (4.73) can be solved by appropriate Krylov subspace methods. Proposition 5.1 in [39] gives information about the spectra of A_* and A_{**}. Denoting the spectrum of A by $\mathrm{spec}(A)$, it is shown by relating the Jordan forms of A and A_* that

$$\mathrm{spec}(A_*) = \mathrm{spec}(A) \cup \overline{\mathrm{spec}(A)}\,. \tag{4.74}$$

As A_{**} is real symmetric, its spectrum is real, i.e.,

$$\mathrm{spec}(A_{**}) = \overline{\mathrm{spec}(A_{**})}\,. \tag{4.75}$$

And as

$$\begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix}^{-1} A_{**} \begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix} = -A_{**}\,, \tag{4.76}$$

the matrices A_{**} and $-A_{**}$ are similar, i.e.

$$-\lambda_i \in \mathrm{spec}(A_{**}) \quad \forall \lambda_i \in \mathrm{spec}(A_{**})\,. \tag{4.77}$$

The following result relating the spectra of A_{**} and $\overline{A}\,A$ is proven in e.g. [61, p. 214]

$$\mathrm{spec}(A_{**}) = \left\{\lambda \in \mathbb{C} \,|\, \lambda^2 \in \mathrm{spec}(\overline{A}\,A)\right\} \tag{4.78}$$

Consider as an example the matrix A resulting from a field-circuit coupled problem with spectrum shown in Figure 3.2. For this matrix the spectra of its counterparts A_* and A_{**} are shown in Figure 4.2 and Figure 4.3 respectively.

4.6 Preconditioning

In practical engineering applications, Krylov subspace methods should be used in combination with a preconditioner in order to obtain efficient algorithms. The convergence of Krylov subspace methods depends on the eigenvalue distribution of the coefficient matrix of the linear system. The idea behind preconditioning is to transform the given system into a system that has a spectrum more favorable to the convergence of the Krylov iteration.

Suppose the coefficient matrix of the linear system (4.1) is real symmetric positive definite. Preconditioning the CG algorithm for solving this system amounts to solving the transformed system

$$C^{-1}\,A\,C^{-T}(C^T x) = C^{-1} b\,. \tag{4.79}$$

Let the symmetric positive definite preconditioning matrix be defined as

$$M = CC^T\,. \tag{4.80}$$

Applying the CG algorithm to the system (4.79) is equivalent to applying a modified or preconditioned CG algorithm to the system (4.1). The unpreconditioned and preconditioned CG algorithm differ in one step. Given

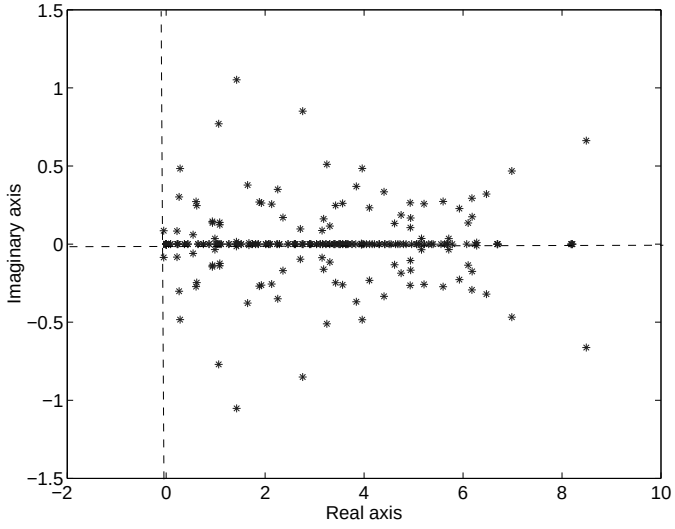


Figure 4.2: Spectrum of A_* for the matrix A whose spectrum is shown in Figure 3.2.

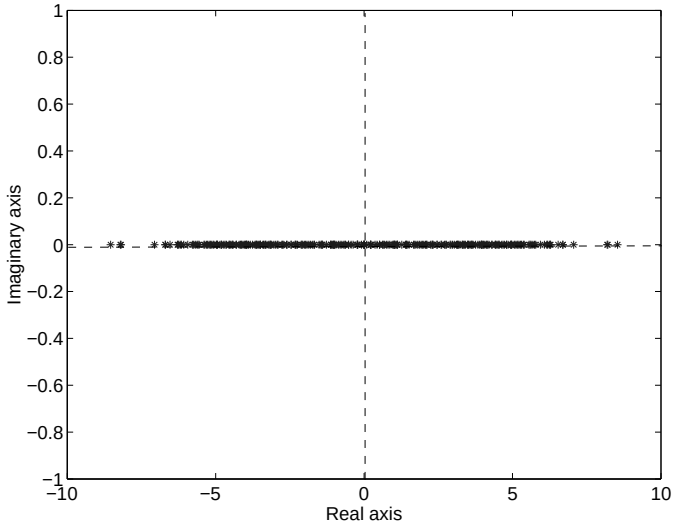


Figure 4.3: Spectrum of A_{**} for the matrix A whose spectrum is shown in Figure 3.2.

the residual vector r_m, this preconditioner step computes the vector z_m by solving the linear system

$$M z_m = r_m. \tag{4.81}$$

However, it is not necessary to split the preconditioner as in (4.80) in order to preserve symmetry. The preconditioned CG algorithm can be written as the unpreconditioned algorithm for the system $M^{-1}A = M^{-1}b$ with the standard inner product is replaced by the inner product defined by M.

Any stationary iterative scheme of the form (4.5) can be implemented as a preconditioner for the CG algorithm by setting the starting solution and the right-hand vector for the stationary iteration equal to

$$x_0 = 0 \quad \text{and} \quad b = r_m \tag{4.82}$$

respectively and performing one iteration in every CG iteration.

The speed of convergence of the preconditioned algorithm depends on the eigenvalues of the matrix $M^{-1}A$. The preconditioned iteration will converge fast if

$$M^{-1}A \approx I \tag{4.83}$$

in some sense. The relations (4.81) and (4.83) provide guidelines on how the matrix M should be constructed. There is a tradeoff to be made between the cost of solving the linear system (4.81) and quality of the approximation (4.83).

For more general linear systems, the preconditioned system has the form

$$M_1^{-1} A M_2^{-1}(M_2 x) = M_1^{-1} b\,. \tag{4.84}$$

where M_1 and M_2 are the so-called left and right preconditioners. Setting either M_1 or M_2 equal to the identity results in a right- and left-preconditioned system respectively.

In the above discussion preconditioning was introduced as a means to speed up the convergence of the Krylov iteration. In another perspective, the roles of the preconditioner and the Krylov iteration are reversed. The Krylov iteration is then viewed as an accelerator of the stationary scheme. We will give an example in Section 6.7.

4.7 Preconditioning Techniques

In this section we give examples of preconditioning strategies based on matrix splitting methods discussed in Section 4.2 and on incomplete factorization schemes.

Matrix splitting preconditioners

The matrices M_{JAC} and M_{SSOR} defined in (4.9) and (4.10) respectively are symmetric positive definite if A is symmetric positive definite and can therefore be used to precondition the CG algorithm. The performance of the Jacobi and the SSOR preconditioner is rather poor. In Chapter 5 we will see multilevel improvements of these preconditioners.

Incomplete matrix factorization preconditioners

In Incomplete LU (ILU) factorization preconditioners, a sparse approximate factorization of A is computed. Given the factors L and U, several variants are distinguished based on the constraints imposed on the residual matrix $R = LU - A$. In the simplest form, denoted by ILU(0), all elements that cause fill-in the Gaussian elimination process are set to zero. The resulting product LU is equal to A in the non-zero pattern of A only. Other variants compute more accurate factorizations by allowing some fill-in in the factors. The amount of fill-in is controlled either by fixing a non-zero pattern of LU, or by setting to zero all fill-in elements that are small in size. In the equivalent of these preconditioners for symmetric positive definite problems, the Gaussian elimination is replaced by a Choleski factorization. The resulting preconditioners are called Incomplete Choleski (IC) preconditioners.

Variable preconditioning

In some problems it can be useful to allow the preconditioner to vary during the iteration. We will give examples of such problems in Section 7.3. In that situation the Krylov iteration needs to be adapted to accommodate a variable preconditioner. Several such modifications have been proposed in the literature [4, 98, 115]. The Flexible GMRES (FGMRES) variant [98] in particular is derived from right preconditioned GMRES.

4.8 Numerical Examples

In this section we present some numerical results of solving the stationary and the time-harmonic example problems described in Section 3.7 by preconditioned Krylov subspace iterations. We postpone the discussion of similar results for field-circuit coupled problems until Section 7.2. In all tests the convergence criterion was taken to be a reduction of the residual norm by 10^{-12}, measured in the two-norm. The solvers we used were taken from the `PETSc` package [5] (see also Chapter 8).

4.8.1 Numerical Examples for Stationary Fields

In this subsection we give convergence results for the CG algorithm in solving the linear systems in example problem **EP**-1 described in Section 3.7.1. In Figure 4.4 on page 78 we plotted the average number of SSOR preconditioned CG iterations per Newton step versus the adaptive refinement step. The number of unknowns in each refinement step is plotted in Figure 3.6. The number of SSOR/CG iterations is high (relative to the problem size) for each of the refinement steps and grows with the problem size. This number is not sensitive to the choice of the relaxation parameter ω. In Chapter 6 we will discuss a family of preconditioners for which the number of iterations is smaller and independent of the problem size.

4.8.2 Numerical Examples for Time-Harmonic Fields

In this subsection we give convergence results for the COCG algorithm in solving the linear systems in example problem **EP**-3 described in Section 3.7.3. In Figure 4.5 we plotted the number of iterations in each adaptive refinement step. We compare SSOR with ILU(0) preconditioning. In the latter we apply reverse Cuthill-McKee reordering. Numerical experiments show that doing so reduces the number of iterations. Figure 4.5 shows that the ILU preconditioned algorithm requires less iterations than when SSOR preconditioning is used. As in the previous subsection however the number of iterations is overall rather high and growing with the problem size.

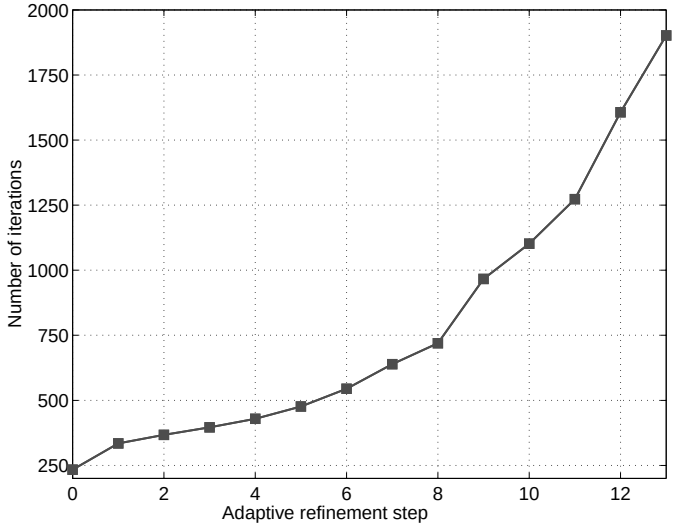


Figure 4.4: Average number of SSOR/CG steps per Newton step versus the number of adaptive refinement step in example problem **EP**-1.

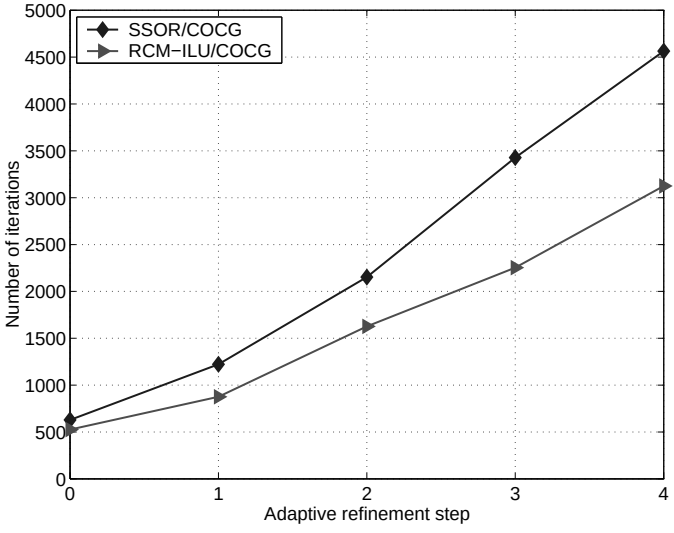


Figure 4.5: Number of SSOR/COCG and RCM-ILU/COCG iterations versus the adaptive refinement step for example problem **EP**-3.

Chapter 5

Basics of Geometric Multigrid

5.1 Introduction

In this chapter we introduce a family of stationary iterative techniques that employ a sequence of coarser level discretizations to speed up the solution process of the discretized PDE problem on the finest level. General references to these so-called *multigrid* or *multilevel* methods include [53, 73, 74, 75, 94, 85, 111]. In this chapter we focus on a class of multigrid techniques that explicitly rely on geometric information of the discretized problem to build the coarser level discretizations. The next chapter deals with a second class of multigrid methods that are able to build coarser level discretization using algebraic information about the finest grid problem only. In order to distinguish both classes, we will refer to the first and the second by *geometric* and *algebraic* multigrid respectively. The aim of this chapter is to explain the general multigrid principles that are common to both algebraic and geometric multigrid. For the ease of presentation, we will only consider model problems in this chapter.

In the following section we discuss the multigrid method for solving a model stationary magnetic field problem. This discussion is extended to a time-harmonic problem without circuit relations in the next section. The multigrid treatment of a general field-circuit coupled problem is postponed until Chapter 7.

5.2 Multigrid for the Stationary Problem

5.2.1 Classical Iterative Schemes as Smoothers

Consider the following discrete two-dimensional magnetostatic problem

$$A\,x = b\,. \tag{5.1}$$

The properties of A are listed in Section 3.5.2. The convergence history of the classical matrix splitting iterative methods presented in Section 4.2 applied to (5.1) show a fast decrease of the residual norm in the initial stage of the iteration. After this initial stage, the convergence stalls. This behavior can be explained by the following property. Classical matrix splitting iterative methods efficiently remove high frequency components of the error. The rate of convergence is poor when only low frequency error conponents are present. The error does not become small, but smooth. This is illustrated for the Gauss-Seidel method in Figure 5.1. In a multigrid context, classical matrix splitting schemes are therefore called *smoothers*. Successive errors are related by (cfr. (4.5))

$$e_{m+1} = S\,e_m \qquad \text{where} \qquad S = (I - M^{-1}\,A)\,, \tag{5.2}$$

where M characterizes the splitting scheme. E.g., it is the scaled diagonal of A in case of damped Jacobi smoothing, and the lower triangular part of A in case of Gauss-Seidel smoothing.

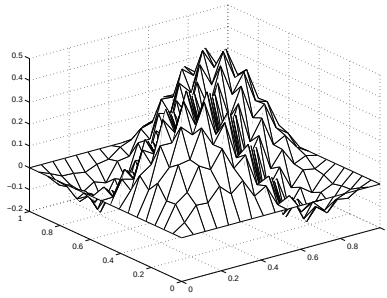
5.2.2 Coarse Grid Acceleration

Smooth errors appear as more oscillatory when represented on coarser discretizations. The basic idea in multigrid techniques is then to use the smoother on different discretizations levels in order to wipe out error components according to their frequency. Computations on a coarse grid can be viewed as a means to speed up convergence of the fine grid iteration.

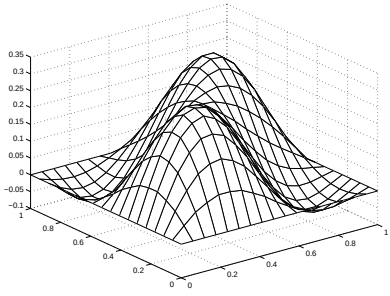
To explain a *two-grid method* for solving (5.1), we consider a fine and coarse grid discretization of the PDE

$$-\frac{\partial^2 A_z}{\partial x^2} - \frac{\partial^2 A_z}{\partial y^2} = \mu\,J_z \tag{5.3}$$

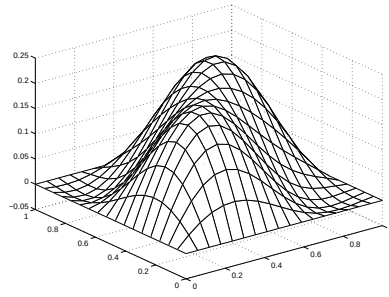
on the unit square. We assume this domain to be discretized uniformly in x and y direction. Let h and H denote the mesh width of a fine and coarse grid discretization of the unit square and let Ω^h and Ω^H denote the corresponding meshes. The mesh Ω^H can be constructed from Ω^h for example by deleting every other grid line in x and y direction. This process is referred to as *standard $h \to 2h$ coarsening*. Restriction and interpolation



(a) Initial error.



(b) Error after 5 iterations.



(c) Error after 10 iterations.

Figure 5.1: Illustration of the smoothing property of the Gauss-Seidel iteration. Figure 5.1(a) shows the initial error while Figure 5.1(b) and Figure 5.1(c) show the error after 5 and 10 iterations respectively.

operators I_h^H and I_H^h are mappings from the fine to the coarse grid solution spaces and vice versa. The discretization of PDE (5.3) on the fine and coarse grid results in the linear systems

$$A^h x^h = b^h \tag{5.4}$$

and

$$A^H x^H = b^H \tag{5.5}$$

respectively. In a two-grid method for solving (5.4), the action of the smoother (5.2) is combined with a *coarse grid correction*. One iteration

of the two-grid method consists of the followings steps. Given an approximate solution x_m^h to (5.4), the high frequency components of the error e_m^h are eliminated by applying a few (typically one or two) steps of the smoother, transforming x_m^h into $\overline{x}_m^h$. Low frequency error components are eliminated by restricting the residual after smoothing $r_m^h = b^h - A^h\,\overline{x}_m^h$ to the coarser grid, solving the defect equation

$$A^H\,e_m^H = I_h^H\,r_m^h\,, \tag{5.6}$$

on the coarser grid, interpolating the coarse grid correction to the fine grid and adding it to the previous approximation $\overline{x}_m^h$

$$\begin{aligned} \overline{x}_m^h \rightarrow \overline{\overline{x}}_m^h \quad \text{where} \quad \overline{\overline{x}}_m^h &= \overline{x}_m^h + I_H^h\,e_m^h \\ &= \overline{x}_m^h + I_H^h\,(A^H)^{-1}\,I_h^H\,r_m^h \\ &= \overline{x}_m^h + I_H^h\,(A^H)^{-1}\,I_h^H\,(b^h - A^h\,\overline{x}_m^h)\,. \end{aligned} \tag{5.7}$$

This implies that the coarse grid correction operator can be written as

$$e_{m+1} = K_{h,H}\,e_m \quad \text{where} \quad K_{h,H} = I^h - I_H^h\,(A^H)^{-1}\,I_h^H\,A^h\,. \tag{5.8}$$

As adding the correction term to the existing approximation may introduce high frequency error components, a few post-smoothing steps are usually performed. Combining the smoother and the coarse grid correction yields the two-grid iteration operator

$$M_{h,H}(\nu_1,\,\nu_2) = (S_2^h)^{\nu_2}\,K_{h,H}\,(S_1^h)^{\nu_1} \tag{5.9}$$

where the subindices 1 and 2 are used to distinguish between the pre- and post-smoother. The integers ν_1 and ν_2 denote the number of pre- and post-smoothing steps respectively. With this two-grid iteration operator corresponds a splitting of the matrix A^h such that

$$M_{h,H} = (I^h - (Q_{h,H})^{-1}A^h)\,. \tag{5.10}$$

Given this splitting, successive iterands produced by the two-grid algorithm can then be related through

$$x_{m+1}^h = x_m^h + (Q_{h,H})^{-1}(b^h - A^h\,x_m^h)\,. \tag{5.11}$$

In Figure 5.2 the two-grid cycle is represented schematically.

5.2.3 Multigrid Extension

In a multigrid algorithm on more than two grids, the two-grid algorithm is applied recursively in order to solve the coarse grid equation (5.6). Depending on the order in which the grids are visited during one multigrid iteration,

one distinguishes different *cycles* such as the V-, W- and F-cycle shown in Figure 5.3. In order to denote the number of pre- and post-smooting step in a cycle, we will use the notation V(ν_1, ν_2) for the V cycle for example.

Multigrid methods are known to be optimal in the sense that the iteration matrix (5.9) can be bounded in some norm by a constant smaller than one *independent* of the mesh size h. This implies that the multigrid convergence is mesh size independent.

5.2.4 Interpolation and Anti-Periodic Boundary Conditions

In this subsection we comment on the interpolation in problems with anti-periodic boundary conditions. In geometric multigrid an interpolation operator with locally *negative* weights is constructed. This will motivate the construction of an interpolation with the same property in algebraic multigrid is Section 6.9.2.

Consider the PDE (5.3) posed on the unit square with the anti-periodic boundary conditions imposed on the left and right boundary Γ_1 and Γ_2 and homogeneous Dirichlet boundary conditions on the top and bottom boundary, i.e.,

$$A_{z\,|\,\Gamma_2} = -A_{z\,|\,\Gamma_1}\,. \tag{5.12}$$

In coding this constraint, the nodes on Γ_2 are assumed to be fictitious points. The values of A_z on Γ_2 can be calculated from the values on Γ_1 whenever needed. Given are grids Ω^h and Ω^H uniform in the x and y directions where Ω^H is constructed Ω^h by standard $h \to 2h$ coarsening. Suppose that the problem is discretized by linear finite elements, and let the function u^h defined in (3.47) be the discrete approximation to A_z. For linear finite elements the coefficients c of the expansion of u^h in terms of the FE basis functions are given by the nodal values of u^h, i.e.,

$$c_i = u^h(\mathbf{a}_i)\,. \tag{5.13}$$

This implies that the conditions (5.12) is enforced in discrete setting by requiring that for the FE nodes $\mathbf{a}_j \in \Gamma_2$ and $\mathbf{a}_k \in \Gamma_1$ lying on the same vertical grid line the following conditions is satisfied

$$c_j = -c_k\,. \tag{5.14}$$

A seven point interpolation will be used to interpolate the variables c^h on the fine grid from their coarse grid equivalents c^H. Let $\mathbf{a}_i$ be a fine grid point having a right coarse grid neighbor $\mathbf{a}_j$ on Γ_2 and let $\mathbf{a}_k$ be the node on Γ_1 connected to $\mathbf{a}_j$ by the periodic boundary conditions. The variable

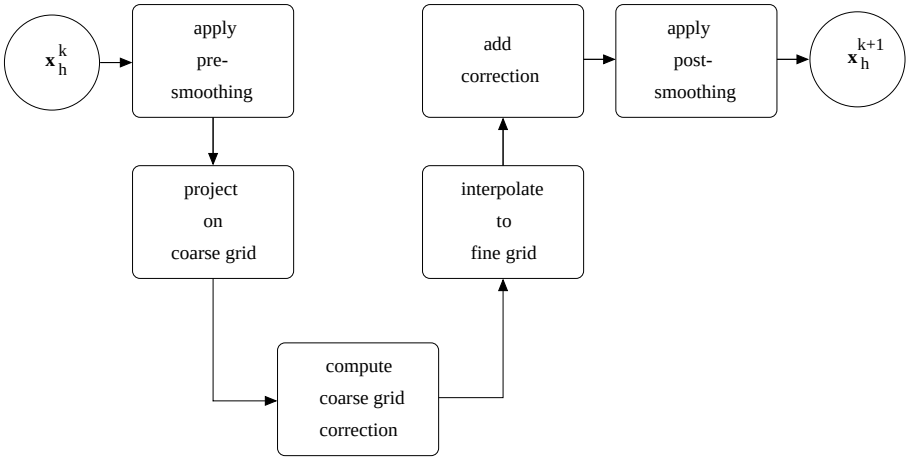


Figure 5.2: Scheme of a two-grid method.

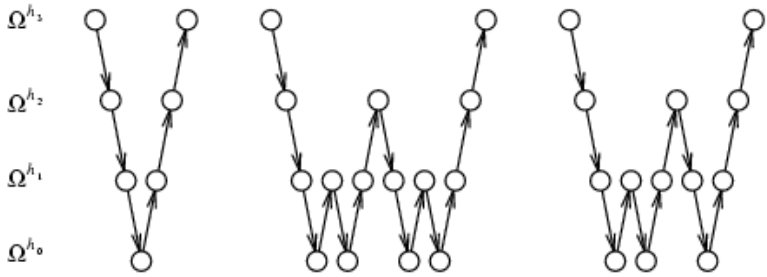


Figure 5.3: Standard multigrid cycles: V, W and F.

c_i^h is interpolated from c_j^H with a positive weight w_{ij}. The node $\mathbf{a}_j$ however lies on the fictious boundary, and has to be eliminated as interpolatory connection. Using the condition (5.14), one obtains that c_i^h is interpolated from c_k^H with weight $w_{ik} = -w_{ij}$.

A similar elimination is carried out in constructing the fine-level discretization matrix A^h. Prior to the elimination the entry A_{ij}^h is negative and the entry A_{ik}^h zero. After the elimination A_{ij}^h is zero and A_{ik}^h is positive: $\mathbf{a}_i$ is connected to $\mathbf{a}_k$ by a positive connection. The interpolation weight w_{ik} is negative if the corresponding matrix entry is positive.

5.3 Multigrid for Time-Harmonic Problems

The classical multigrid algorithms are now generalised to solve model time-harmonic problems. We are interested in obtaining mesh width independent convergent algorithms and we will analyze the influence of the skin-depth on the convergence. A model problem on the square $\Omega = [0, L] \times [0, L]$ for the equation

$$-\frac{\partial^2 \hat{A}_z}{\partial x^2} - \frac{\partial^2 \hat{A}_z}{\partial y^2} + j\,\omega\,\sigma\,\mu\,\hat{A}_z = \mu\,\hat{J}_{s,z} \tag{5.15}$$

where ω, σ and $\mu = 1/\nu$ are the pulsation, the electric conductivity and the magnetic permeability respectively is considered. On the boundary inhomogeneous Dirichlet conditions are imposed such that for $\omega \neq 0$ the function

$$\hat{A}_z = \frac{\hat{J}_{s,z}}{j\,\omega\,\sigma}\left(1 - \frac{\cosh\left[\sqrt{j\,\omega\,\sigma\,\mu}(x - L/2)\right]}{\cosh\left[\sqrt{j\,\omega\,\sigma\,\mu}L/2\right]}\right) \tag{5.16}$$

is the exact solution of the problem. The problem is discretized on a regular mesh with mesh width $h = L/2^k$, $k = 2, 3, \ldots$ in both x and y direction using linear triangular finite elements. The matrix of the resulting linear system can be represented by the stencil

$$A \sim \frac{1}{h^2}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} + j\,\frac{\omega\,\sigma\,\mu}{12}\begin{bmatrix} 1 & 1 & 0 \\ 1 & 6 & 1 \\ 0 & 1 & 1 \end{bmatrix}. \tag{5.17}$$

In the multigrid algorithm for solving this linear system, the intergrid transfer operators are based on the diffusion part of (5.15), and are thus real-valued. The smoother is based on a splitting of the matrix (5.17), and is complex. These choices can be motivated by Fourier analysis. A Fourier analysis of this multigrid algorithm is carried out in [62].

In our numerical experiments we choose standard $h \to 2h$ coarsening, the W-cycle, seven-point restriction and interpolation and red-black Gauss-Seidel as smoother (see e.g. [53]). The various parameters have the following values: $L = 0.01$ m, $\sigma = 0.57 * 10^8\ \Omega^{-1}\mathrm{m}^{-1}$, $\mu = 4\pi * 10^{-7}\ \mathrm{VsA}^{-1}\mathrm{m}^{-1}$, $J_s = 10^6\ \mathrm{Am}^{-2}$, and $\omega = 2\pi f$, where f is the frequency. In Table 5.1 we listed the number of cycles required to reduce the initial residual by a fixed amount (namely by 10^{-12}) as a function of the mesh width for different frequencies. From Table (5.1), we conclude that for a given frequency, the required number of cycles reaches an upper limit, indicating that the convergence is mesh size independent. In Table 5.2, we listed the asymptotic

	Number of Cycles		
Mesh Width	f = 1e-5 Hz	f=50 Hz	f=5e4 Hz
4 × 4	11	11	9
8 × 8	12	12	12
16 × 16	13	12	13
32 × 32	12	12	18
64 × 64	12	12	29
128 × 128	12	12	14
256 × 256	12	12	12
512 × 512	12	12	12

Table 5.1: Number of multigrid cycles as a function of mesh width for different frequencies.

convergence factor ρ of the multigrid algorithm as a function of the frequency for various mesh widths. We observe that that the speed of convergence decreases with increasing frequency until the frequency reaches a threshold value. By increasing the frequency beyond this value, the multigrid algorithm speeds up again. This convergence behavior is in agreement with results from Fourier analysis.

	Mesh Width 32 × 32							
f [Hz]	1e-6	1	1e4	1e5	2e5	3e5	5e5	1e6
ρ	0.12	0.12	0.14	0.35	0.40	0.35	0.18	0.13

	Mesh Width 128 × 128							
f [Hz]	1e-6	1	1e4	2e5	1e6	2e6	4e6	5e6
ρ	0.12	0.12	0.125	0.13	0.35	0.37	0.35	0.30

Table 5.2: Asymptotic convergence factor as a function of the frequency for different mesh widths.

The fast convergence for frequencies that are high relative to the mesh width is irrelevant from a practical point of view. For such high frequencies a finer discretization is required to ensure the accuracy of the numerical solution. Furthermore, equation (5.15) is a model for the Maxwell equations assuming low frequency and thus neglecting the displacement current density.

Chapter 6

Algebraic Multigrid

6.1 Introduction

In this chapter we introduce a particular class of multigrid methods called *algebraic* multigrid methods. After having motivated their development, we explain them in detail for solving magnetostatic field problems. We describe how the applicability of the method can be extended to solve time-harmonic problems. By presenting numerical examples we give evidence of the efficiency and robustness of the method.

Multigrid techniques have been successfully applied to solve a broad range of discretized differential problems [57, 54, 34]. In some applications however it is cumbersome to construct efficient multigrid algorithms. For example, in unstructured grid problems on complicated geometries it is difficult to select appropriate coarse grids. Problems with discontinuous coefficients often result in very small convergence rates for standard geometric multigrid.

Motivated by these difficulties, *algebraic* multigrid (AMG) algorithms have been developed as an alternative to the classical geometric multigrid approaches. AMG algorithms construct the sequence of coarser grids, the intergrid transfer and coarse grid correction operators automatically from the finest grid matrix. Once this sequence has been constructed, AMG solvers proceed by solving the linear system by standard multigrid cycling. AMG solvers require only the linear system as input. No information about the geometry of the discretized PDE is used. This makes them attractive as plug-in solvers in application codes.

The development of AMG started in 1982 by a paper by Brandt e.a. [18] and was then popularized by Ruge and Stüben [95]. This work combines the concepts of matrix-dependent interpolation and Galerkin-based coarsening.

It resulted in the first fairly general AMG code called `AMG1R5`.

The development of AMG lay dormant until in the early '90s computations on unstructured grids demanded increasingly more effective iterative solvers. The original AMG idea attracted new interest [26, 50, 48, 65, 125]. Other AMG approaches were developed. An non-exhaustive list of these approaches includes: AMG based on aggregation by Braess [16] and by Vaněk e.a. [116, 117], based on approximate Schur complements by Reusken [93] and by Axelsson and Vassilevski [2, 3], based on multilevel ILU ideas by Botta e.a. [15], by Bank and Wagner [10], by Notay [83] and by Saad [99], based on energy minimizing interpolation by Wan e.a. [120] and on an element by element approach by Brezina e.a. [19].

In recent years, Stüben enhanced and extended `AMG1R5` in several ways [67, 108]. He provided entirely new coarsening and interpolation strategies and optimized `AMG1R5` in terms of memory management. He also provided functionality to solve linear systems resulting from the discretization of systems of coupled PDEs. The successor of `AMG1R5` is called `SAMG` .

The efficiency of multigrid algorithms hinges on the proper interaction of the smoother and coarse grid correction. Non-smooth errors should be damped by the smoother, while smooth errors should be treated by the coarse grid correction. In geometric multigrid the coarsening is fixed and the smoothing is adopted to the problem at hand. In algebraic multigrid, on the contrary, the smoother is fairly simple and work is invested towards adapting the coarse grid correction to the local properties of the smoother.

As we used `AMG1R5` and `SAMG` in our work, we will make these ideas more precise by describing the Brandt-Ruge-Stüben approach to AMG. This approach was originally developed for solving problems with coefficients that are real symmetric positive definite M-matrices. We will first consider solving discrete stationary magnetic systems. A separate discussion is devoted to problems with anti-periodic boundary conditions as such problems yield coefficient matrices that violate the M-matrix property. The generalization to solving discrete time-harmonic systems will be treated afterwards.

6.2 AMG Components

Let

$$A^h x^h = b^h \tag{6.1}$$

be a discrete magnetostatic system on a grid Ω^h with characteristic mesh size h. In solving (6.1) by AMG, one distinguishes two phases: a *setup* phase and a *cycling* phase.

In the setup phase AMG automatically constructs a hierarchy of coarser level discretizations using information contained in the system matrix A^h

only. In the cycling phase, AMG uses this hierarchy to solve the linear system (6.1) by standard multigrid cycling.

In constructing the first coarser level equivalent of (6.1), AMG selects a proper subset C^h of the fine grid points Ω^h. The subset C^h induces a partitioning of Ω^h

$$\Omega^h = C^h \cup F^h \,, \tag{6.2}$$

referred to as the C/F splitting of Ω^h. The set of coarser level points Ω^H is identified with C^h, and the coarser level system is denoted by

$$A^H x^H = b^H \,. \tag{6.3}$$

Interpolation and restriction operators I_H^h and I_h^H mapping from Ω^H to Ω^h and vice versa are constructed. In AMG, these operators depend on the matrix A^h. For symmetric problems the restriction operator is set equal to the transpose of the prolongation operator

$$I_h^H = (I_h^H)^T \,. \tag{6.4}$$

Having the interpolation and restriction operators I_H^h and I_h^H at ones disposal, AMG builds the coarser system matrix A^H by the Galerkin formula

$$A^H = I_h^H \, A^h \, I_H^h \,. \tag{6.5}$$

The above procedure is applied recursively to compute the next coarser level discretization using as input A^H. The recursion terminates if the number of points on the coarsest level drops below a prescribed number or if the fill-in in the coarsest level equivalent of the operator becomes too large.

Having fixed the restriction operator and the coarse level discretization by (6.4) and (6.5) respectively, only the construction of the C/F splitting (6.2) and the interpolation operator remain to be detailed in order to complete the description of the setup phase. This will be elaborated in Section 6.4 and Section 6.5.

In the AMG cycling phase, smoothing is performed by a simple point Gauss-Seidel smoother. To denote the operators used in the cycling phase, we will use the notation introduced in Section 5.2.2.

For symmetric positive definite matrices A^h, the coarse grid correction $K_{h,H}\, e$ (5.8) produced by Galerkin based coarsening (6.5) can be shown to be optimal in the sense that its A^h-norm is minimal among errors e varying in the range of the interpolation. From this, one derives that the convergence of AMG hinges on how well the range of the smoother and the interpolation match. For a fixed smoother, this implies that the C/F splitting and the interpolation have to be adapted to the local properties of that smoother. In

order to explain how AMG constructs a C/F splitting and the interpolation conform to these requirements, one needs a way to characterize smoothness purely algebraically, i.e., without reference to a grid. This is discussed in the next section.

6.3 Algebraic Smoothness

6.3.1 Algebraic Versus Geometric Smoothness

From now on, the indices h and H will be omitted if the meaning of the symbols is clear from the context.

An error e is said to be *algebraically smooth* if the smoother S defined in (5.2) is slow to converge on e, i.e., if

$$Se \approx e\,. \tag{6.6}$$

In order to illustrate the difference between algebraic smoothness and the more intuitive notion of geometric or visual smoothness, three model differential problems will be considered here. All three model problems are Poisson-type problems on the unit square discretized by standard central finite differences.

The first model problem is the x-anisotropic equation

$$\epsilon \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f\,, \tag{6.7}$$

where $\epsilon \ll 1$, supplied with Dirichlet boundary conditions. After applying a few point Gauss-Seidel steps to the discretized problem, the iteration error e satisfies (6.6) and is thus by definition algebraically smooth. The error is geometrically smooth in the y-direction only. In x-direction it is still oscillatory. This is illustrated in Figure 6.1.

The second model problem is the equation

$$\frac{\partial}{\partial x}\left(\nu \frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(\nu \frac{\partial u}{\partial y}\right) = f\,, \tag{6.8}$$

where the diffusion coefficient ν is discontinuous across the interface between two regions having different diffusion characteristics, supplied with Dirichlet boundary conditions. After applying a few point Gauss-Seidel steps, the error is again algebraically smooth. It is geometrically smooth over the interior of the two regions only. The gradient of the smoothed error is discontinuous across the interface. This is illustrated in Figure 6.2.

The third problem is the equation

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f\,, \tag{6.9}$$

where anti-periodic boundary conditions are imposed on the left and right edge of the unit square and Dirichlet conditions on the top and bottom edge. In this model the algebraically smooth error is geometrically smooth in the interior of the square. Like the solution of the continuous problem however, the smoothed error jumps across the boundary where anti-periodic boundary conditions are imposed.

In typical electromagnetic engineering problems, features of the first two model problems are present.

Let A be the symmetric positive definite coefficient matrix of the linear system (6.1) and let N be its dimension and D its diagonal. To capture algebraic smoothness in mathematical terms, the following norms are introduced on $\mathbb{R}^N$

$$\| v \|_1^2 = (A v, v) \qquad \text{and} \qquad \| v \|_2^2 = \left(D^{-1} A v, A v \right) . \tag{6.10}$$

The notation for these norms is motivated in [95]. Given two symmetric positive definite matrices B_1 and B_2, the following equivalence holds

$$(B_1 v, v) \leq c (B_2 v, v) \qquad \Leftrightarrow \qquad \rho(B_2^{-1} B_1) \leq c. \tag{6.11}$$

Using this equivalence, one can prove that the norms defined in (6.10) are related by

$$\| v \|_2^2 \leq \rho(D^{-1} A) \ \| v \|_1^2 . \tag{6.12}$$

For an eigenvector v of $D^{-1}A$ corresponding the eigenvalue λ the following equality holds

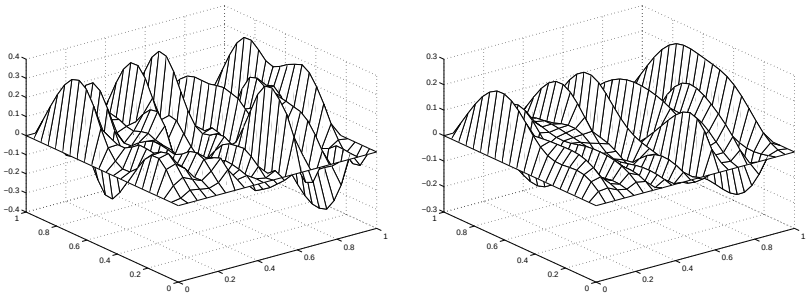
$$\| v \|_2^2 = \lambda \| v \|_1^2 . \tag{6.13}$$

Algebraically smooth errors in the case of Jacobi and Gauss-Seidel iterations are linear combination of eigenfunctions of $D^{-1} A$ corresponding to small eigenvalues. This is readily seen for damped Jacobi relaxation as small eigenvalues of $D^{-1} A$ correspond to eigenvalues of $S = (I - \omega D^{-1} A)$ close to one. The same holds for Gauss-Seidel relaxation. As a consequence of (6.13) we have that algebraically smooth errors e satisfy $\| e \|_2 \ll \| e \|_1$. In terms of the residual $r = A e$ this means

$$(D^{-1} r, r) \ll (e, r) , \tag{6.14}$$

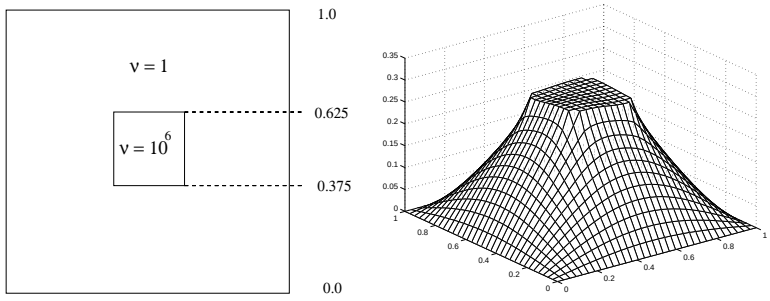
or, more explicitly,

$$\sum_i r_i^2 / a_{ii} \ll \sum_i r_i e_i . \tag{6.15}$$



(a) Error after 5 iterations. (b) Error after 10 iterations.

Figure 6.1: Illustration of the smoothing property of the Gauss-Seidel iteration in an x-anisotropic problem. After 10 iterations, the error is still oscillatory in x-direction.



(a) Values for ν. (b) Error after 10 iterations.

Figure 6.2: Illustration of the smoothing property of the Gauss-Seidel iteration in a problem with a discontinuous diffusion coefficient. After 10 iterations, the error is discontinuous across the interface defined by the discontinuity of the diffusion coefficient.

Therefore, on the average, we can expect for algebraic smooth errors e that

$$|r_i| \ll a_{ii}|e_i| \qquad \forall\, i. \tag{6.16}$$

Two grid points i and j in Ω^h are said to be *coupled* or *connected* if the corresponding matrix entry a_{ij} is non-zero. The *neighborhood* of a grid point i is defined by

$$N_i = \{j \neq i \,|\, a_{ij} \neq 0\}\,. \tag{6.17}$$

From (6.16), we obtain a quite good approximation of e_i as a function of its neighboring values e_j, $j \in N_i$, by evaluating

$$(r_i =)\; a_{ii}\, e_i + \sum_{j \in N_i} a_{ij}\, e_j = 0\,. \tag{6.18}$$

This is an important characteristic of algebraic mooth errors as it will provide the basic information for the construction of the interpolation in Section 6.5.

We now give an interpretation of algebraic smoothness in terms of relative magnitude and sign of the connections between grid points. In doing so, we distinguish discrete systems of the form (6.1) according to whether anti-periodic boundary conditions are present or not.

6.3.2 Algebraic Smoothness for M-Matrices

If no anti-periodic boundary conditions are present in the problem (6.1), then (3.95) states that coefficient matrix A is an M-matrix. This, in particular, means that all (non-zero) connections a_{ij} for $i \neq j$ are negative. In practical computations on unstructured meshes the matrix A might have a few positive off-diagonal entries due to roundoff errors.

The coupling between the points i and j is said to be *strong* if a_{ij} is negative and large in absolute value compared to the diagonal entry a_{ii}. Any positive connection is considered to be weak at this point.

The first two model problems at the beginning of this section yield M-matrices. In these two problems the error satisfies the relation (6.6) and is thus by definition algebraically smooth. It is geometrically (or visually) smooth in the directions of strong couplings only.

6.3.3 Algebraic Smoothness and Anti-Periodic Boundary Conditions

In discrete problems with anti-periodic boundary conditions, the coefficient matrix contains large off-diagonal entries a_{ij} in the rows i corresponding to

points lying on the anti-periodic boundary. These off-diagonal entries are of the same order of magnitude as the diagonal entry a_{ii}. This is the important difference compared with the M-matrix case, in which connections of large absolute value are always negative.

In problems with anti-periodic boundary conditions algebraically smooth errors are geometrically smooth in the directions of strong negative couplings. They have a jump discontinuity across the large positive connections.

6.4 Construction of the Interpolation

The concrete construction of the interpolation operator will be detailed in the next section. First some general requirements this operator has to satisfy are described.

Assume a C/F splitting of the points in Ω^h is given. For the interpolation, AMG associates to each fine grid point $i \in F^h$ a set of interpolatory connections $P_i \subset C^h$ and interpolation weights $\{w_{ij} \,|\, j \in P_i\}$. If interpolation is done along direct connections only, then $P_i \subset N_i \cap C^h$. In this case the interpolation is defined by

$$e_i^h = (I_H^h e^H)_i = \begin{cases} e_i^H & \text{if } i \in C^h \\ \sum_{j \in P_i} w_{ij}^h e_j^H & \text{if } i \in F^h \end{cases} . \tag{6.19}$$

Obtaining an efficient multigrid algorithm requires smooth error components to be accurately interpolated. For symmetric M-matrices, convergence of the two level cycle with one post-smoothing step can be proven if after interpolation (6.18) is satisfied $\forall i \in F^h$ [108]. This can be achieved if one sets

$$\forall i \in F^h \qquad P_i = N_i \qquad \text{and} \qquad w_{ij} = -a_{ij}/a_{ii} \,. \tag{6.20}$$

Setting $P_i = N_i$ however is too strong a requirement to be practical. It requires C^h to be large enough to contain all N_i for $i \in F^h$. It yields a slow coarsening of Ω^h and leads to a large fill-in in the Galerkin coarser grid matrices (6.5). As the smoothing operator on the coarser grid is based on a matrix splitting of the Galerkin operator, the computational complexity of the smoothing operation will grow. Since the computational cost of the multigrid cycle is largely determined by the smoothing operation, the choice (6.20) leads to computationally too expensive multigrid cycles.

The cost of the multigrid cycle can be reduced by allowing a faster coarsening, i.e., by limiting the size of the coarser grid C^h. By performing the coarsening too fast however, one looses the accuracy of the interpolation and therefore the (fast) convergence of the multigrid cycle. In truncating the set of interpolatory connections, one therefore seeks to balance the speed

of convergence and the computational work per cycle. The computation of the interpolation weights and the selection of the coarser grid points thus become closely coupled processes.

Convergence is guaranteed by requiring that each fine grid point is sufficiently connected to its interpolatory connections, i.e., by requiring $\sum_{j \in P_i} a_{ij}$ to be large enough. This condition is easily fulfilled by allowing a slow coarsening, i.e., by allowing many points to enter the coarse grid. To accelerate the coarsening, a maximal independent set condition is imposed on the coarse grid points. This condition prevents two coarse grid points to be strongly connected to each other. The above requirements on the coarsening and the interpolation counteract and are such that coarsening is done in the direction of strong couplings.

6.5 Practical Interpolation Algorithms

6.5.1 Set of Strong Connections

Practical interpolation algorithms are described as detailed in [108] and implemented in `SAMG`. These algorithms are an improvement and extension of those used in `AMG1R5` as will be shown in the numerical experiments in Section 6.9.

We introduce the following notations for matrices with both positive and negative off-diagonal entries

$$a_{ij}^{-} = \begin{cases} a_{ij} & (\text{if } a_{ij} < 0) \\ 0 & (\text{if } a_{ij} \geq 0) \end{cases} \qquad \text{and} \qquad a_{ij}^{+} = \begin{cases} 0 & (\text{if } a_{ij} \leq 0) \\ a_{ij} & (\text{if } a_{ij} > 0) \end{cases} . \tag{6.21}$$

A variable i is said to be *strongly coupled to* variable j if

$$-a_{ij} \geq \epsilon_{str} \max\{ |a_{ik}| \mid a_{ik} < 0, k \in N_i \} \tag{6.22}$$

with fixed $0 < \epsilon_{str} < 1$, and the set of strong connections of a variable i is defined as

$$S_i = \{ j \in N_i \,|\, i \text{ strongly coupled to } j \} . \tag{6.23}$$

In this definition, all positive connections are considered to be weak. Practical experience has shown that the precise value of ϵ_{str} is not critical in many applications. A reasonable default is 0.25. For a discussion of situations in which the value of ϵ_{str} is critical however, we refer to [19].

We assume a C/F splitting has been constructed by the algorithm presented in [108]. This allows one to define the sets

$$C_i^s = C \cap S_i \qquad \text{and} \qquad F_i^s = F \cap S_i . \tag{6.24}$$

We describe two interpolation techniques: *direct* and *standard* interpolation. By interpolating to a variable $i \in F$ by direct interpolation the strong couplings of i to other points in F are neglected, i.e. P_i is set to C_i^s. The standard interpolation improves the direct interpolation by extending the interpolation stencil to points in F_i^s. First interpolation for M-matrices and small perturbations thereof is described. Later interpolation for problems with anti-periodic boundary conditions will be treated.

6.5.2 Direct Interpolation

Assume A to be an M-matrix. The slow variation of an algebraically smooth error in the direction of strong negative couplings allows one to determine the value of a smooth error in a point i by a weighted average of the values at strong neighbors. Given a set $P_i \subseteq C \cap N_i$, this justifies the approximation

$$\frac{\sum_{j\in P_i} a_{ij}\, e_j}{\sum_{j\in P_i} a_{ij}} \approx \frac{\sum_{j\in N_i} a_{ij}\, e_j}{\sum_{j\in N_i} a_{ij}} \tag{6.25}$$

for smooth errors. The approximation (6.25) is better satisfied the more strong couplings of an F-variable i are contained in P_i. It allows one to approximate (6.18) by

$$a_{ii}\, e_i + \alpha_i \sum_{j\in P_i} a_{ij}\, e_j = 0 \quad \text{with} \quad \alpha_i = \frac{\sum_{j\in N_i} a_{ij}}{\sum_{j\in P_i} a_{ij}}\,, \tag{6.26}$$

which leads to an interpolation formula (6.19) with interpolatory connections

$$P_i = C_i^s\,, \tag{6.27}$$

and matrix-dependent, positive weights

$$w_{ij} = -\alpha_i\, a_{ij}/a_{ii} \qquad (i \in F,\, j \in P_i)\,. \tag{6.28}$$

If the matrix A has a limited number of positive entries relatively small in size, the interpolation is modified by adding the positive entries to the diagonal. For smooth errors (6.18) is then approximated by

$$\begin{aligned} \tilde{a}_{ii}\, e_i + \alpha_i \sum_{j\in P_i} a_{ij}^-\, e_j = 0 \quad &\text{with} \quad \tilde{a}_{ii} = a_{ii} + \sum_{j\in N_i} a_{ij}^+ \\ &\text{and} \quad \alpha_i = \frac{\sum_{j\in N_i} a_{ij}^-}{\sum_{j\in P_i} a_{ij}^-} \end{aligned} \tag{6.29}$$

instead of (6.26), yielding positive interpolation weights

$$w_{ij} = -\alpha_i \, a_{ij}/\tilde{a}_{ii} \qquad (i \in F,\, j \in P_i)\,. \tag{6.30}$$

This treatment of positive off-diagonal entries prevents α_i from becoming positive and obtaining negative interpolation weights. The approximation in (6.29) is only accurate enough if for each $i \in F$, an algebraically smooth error satisfies

$$\sum_j a_{ij}^+ \, e_j \approx e_i \, \sum_j a_{ij}^+\,, \tag{6.31}$$

which, for $j \neq i$, either requires $e_j \approx e_i$ or a_{ij}^+ to be small compared to a_{ii}.

6.5.3 Standard Interpolation

In the direct interpolation a fine grid point i is not interpolated from its strong connections on the fine grid. For a point $j \in F_i^s$, this interpolation can be improved by eliminating e_j from

$$a_{jj} \, e_j + \sum_{k \in N_j} a_{jk} \, e_k = 0\,, \tag{6.32}$$

to obtain

$$e_j = -\sum_{k \in N_j} a_{jk} \, e_k / a_{jj}\,. \tag{6.33}$$

The latter expressions are used to eliminate e_j, $j \in F_i^s$, from (6.18), resulting in a new equation for e_i

$$\hat{a}_{ii} \, e_i + \sum_{j \in \hat{N}_i} \hat{a}_{ij} \, e_j \text{ with } \hat{N}_i = \{j \neq i \,|\, \hat{a}_{ij} \neq 0\}\,. \tag{6.34}$$

By defining P_i as

$$P_i = C_i^s \cup (\cup_{j \in F_i^s} C_j^s)\,, \tag{6.35}$$

we now define the interpolation exactly as in (6.29)-(6.30) with all a's replaced by $\hat{a}$'s and N_i replaced by $\hat{N}_i$.

The interpolation described in this subsection is the interpolation used by default in `SAMG`, and is therefore called the *standard* interploation.

6.5.4 Anti-Periodic Boundary Conditions

Let Γ_1 and Γ_2 be part of the boundary connected by anti-periodic boundary conditions. Across these boundaries nodes are connected by large positive connections and the algebraically smooth error exhibits a jump as discussed in Subsection 6.3.3. Therefore condition (6.31) is strongly violated across these boundaries. The interpolation based on strong negative connections solely does not reflect the discontinuity of the error. The interpolation has to be modified in such a way that strong positive connections are taken into account.

In Subsection 5.2.4 we saw that in a geometric multigrid approach, the elimination of the boundary condition yielded an interpolation with negative weights. Note that, as AMG does not have any information about the geometry of the model, it has no immediate way of distinguishing interior and boundary points.

To describe the fix proposed in [108], assume a C/F splitting and the set of interpolatory connections P_i for each variable i in F have been constructed as described previously. The strong positive $F-F$ connections of a variable i in F are searched for, i.e. one verifies where there exist $j \neq i$ such that

$$a_{ij} \geq \epsilon^+_{str} \max_{k \neq i} |a_{ik}| \,, \tag{6.36}$$

where ϵ^+_{str} is some tolerance set to, for instance, $\epsilon^+_{str} = 0.5$. The set of strong connections of the variable i is increased to adding all j satisfying (6.36) to S_i. The variable j corresponding to the strongest positive coupling is added to P_i. The variable i is interpolated from variable j with *negative* weight

$$w_{ij} = -\beta_i \frac{a^+_{ij}}{\sum_{k \in N_i} a^+_{ik}} \quad \text{with} \quad \beta_i = \frac{\sum_{j \in N_i} a^+_{ij}}{\sum_{j \in P_i} a^+_{ij}} \,. \tag{6.37}$$

6.6 Computational Work Metrics

The computational work for cycling and the amount of required memory by AMG is characterized by two numbers [108, 26]. Denoting by N_k the number of grid points on level k, the *grid complexity* is defined by

$$c_G = \sum_k N_k / N_1 \,. \tag{6.38}$$

Given a number of levels constructed, the level $k = 1$ corresponds to the finest level. The grid complexity gives an indication of how fast the grids

are reduced in size. Denoting by r_k the average number of non-zeros in the system matrix on level k, the *algebraic complexity* is defined by

$$c_A = \sum_k r_k N_k / r_1 N_1 . \tag{6.39}$$

It is the ratio of the number of non-zeros on all levels to those on the finest grid. It gives an indication of the number of operations required for smoothing on all grids relative to that on the finest grid. It also gives an indication of the amount of memory required for storing the required matrices all grids relative to that of storing the fine grid matrix.

The geometric and algebraic complexities are influenced by the speed of coarsening. A faster coarsening will lead to smaller complexities. To obtain small complexities, an *aggressive* coarsening strategy was implemented in SAMG. Given a C/F splitting of Ω^h computed by the standard algorithm, aggressive coarsening calls the standard algorithm a second time on C^h. The loss in convergence speed caused by the fast coarsening is compensated by using AMG as a preconditioner for a Krylov subspace solver (see Section 6.7).

6.7 Krylov Acceleration

We discuss under which conditions the two-grid iteration operator $Q_{h,H}$ (5.10) defined by AMG can be used as preconditioner for the CG iteration and what the advantages of this approach are.

The interpolation operator I_H^h has full column rank by (6.19) and therefore A^H is positive definite if A^h is positive definite [108]. If the coarse grid solve is done exactly, if $\nu_1 = \nu_2$ and if

$$S_1^h = (S_2^h)^T , \tag{6.40}$$

then we have by Theorem 3.5 in [60] that

$$Q_{h,H} \text{ is symmetric positive definite} . \tag{6.41}$$

The matrix $Q_{h,H}$ is only defined implicitly. Condition (6.40) can be enforced by doing the post-smoothing in reversed order of the pre-smoothing. The coarse solve can be performed by a recursive application of the two-grid algorithm. By property (6.41), AMG can be used as a preconditioner for the CG algorithm.

Numerical experiments in Section 6.9 will show two advantages of using AMG as a preconditioner over using it as a solver. Firstly, the AMG preconditioned CG algorithm requires less iterations and less CPU time to converge. Secondly, the number of iterations of AMG as preconditioner

is less sensitive to the mesh width, and, in non-linear problems, to perturbations in the linearized system matrix. The outer CG iteration adds robustness to AMG.

The convergence of AMG as a solver is determined by the spectral radius of the matrix $I - (Q_{h,H})^{-1}A$. The reduction in the number of iterations by using AMG as a preconditioner can be explained by examining the convergence of the Ritz values towards the eigenvalues of the preconditioned operator during the CG iteration. Suppose that the spectrum of $I - (Q_{h,H})^{-1}A$ consists of a cluster of eigenvalues close to 0 surrounded by a few outliers. In this case only a few CG iterations are needed for the outermost Ritz values to become close approximations of the outliers in the spectrum of $I - (Q_{h,H})^{-1}A$. Once they are sufficiently close, the AMG solver converges as if those outliers were not present. The effective spectral radius is reduced and the speed of convergence increased. An analysis along these lines can be used to explain why a poorly converging AMG scheme can make an efficient, mesh width independently converging preconditioner. We will give an example of such an analysis is Section 6.9.2.

6.8 AMG for Time-Harmonic Systems

We extended the applicability of the Brandt-Ruge-Stüben AMG approach to solve discrete time-harmonic systems. We describe both the algorithm and its implementation.

6.8.1 Algorithm

An AMG algorithm for solving discrete time-harmonic systems should be such that it coincides with a geometric multigrid algorithm for model problems. In Section 5.3 we saw that geometric multigrid for model time-harmonic problems consists of the following components. In constructing the coarser levels and in performing intergrid transfer, geometric multigrid uses the same components as in stationary problems. Only the smoothing is performed in complex arithmetic. The same algorithm is obtained in AMG setting if we only pass the real part of the fine grid system matrix to the routines that compute the C/F splitting and compute the interpolation. Only the routines that compute the coarser grid matrix and carry out the multigrid cycling see both the real and imaginary part of the fine grid system matrix.

To discuss the algorithm in more detail, let

$$A^h x^h = b^h \tag{6.42}$$

be a discrete single frequency time-harmonic system on a grid Ω^h. The matrix A^h has real and imaginary parts

$$A^h = A^h_R + j\, A^h_I \,. \tag{6.43}$$

As described in Section 3.5.3, the matrix A^h_R has the properties of a discrete stationary matrix and therefore the AMG setup on A^h_R can be performed as described earlier.

A complex coarser grid equivalent A^H of the symmetric matrix A^h is computed as follows. First a C/F splitting of Ω^h and the corresponding interpolation operator I^H_h is computed based on the real part A^h_R. Given I^H_h and its transpose I^h_H, A^H is computed by the Galerkin product

$$A^H = I^H_h\, A^h\, I^h_H = I^H_h\, A^h_R\, I^h_H + j\, I^H_h\, A^h_I\, I^h_H \,. \tag{6.44}$$

Formula (6.44) implies that A^H inherits the complex symmetry from A^h. The real part of A^H is the Galerkin coarsening of the real part of A^h. The two grid algorithm can thus be applied recursively to compute the intergrid and coarser grid operators on the next levels.

After the setup phase, the multigrid cycling can be performed in the usual way. The Gauss-Seidel smoothing operator is based on the splitting of the matrix A^H and is thus complex.

6.8.2 Implementation

The implementation of the AMG setup for time-harmonic systems requires providing the construction of the C/F splitting and the interpolation operator with the real part of the coefficient matrix as input and providing the computation of the Galerkin product with the entire matrix as input. We found two ways to make this possible without having to drastically change the AMG code.

The first alternative consists in rewriting the complex time-harmonic system (6.42) into the equivalent real form (4.73) or (4.72). The latter can be viewed as a linear system resulting from the discretization of a system of two coupled PDEs, one for the real and one for the imaginary part of the vector potential. They can be solved using an AMG solver for systems of PDEs . By making appropriate choices for AMG parameters in the setup of the system AMG code, it is possible to exactly implement the setup algorithm as proposed in the previous section.

In our work we use the system AMG code developed by Stüben. This code requires the diagonal of the input matrix to be positive. Therefore only the equivalent real form (4.72) is considered from here on.

The system version of `SAMG` builds the coarser discretizations based solely on the diagonal blocks of the matrix (4.72). In our case both diagonal blocks

are equal to the real part of the matrix (6.43). In SAMG we choose options such that the C/F splitting and the interpolation operator are built from the first block of equations in (4.72) and forced upon the second. In this way we avoid computing the interpolation twice.

To give more details on how SAMG performs smoothing in real arithmetic, let x and $\overline{x}$ denote the complex iterand before and after smoothing. Omitting the index h, a point Gauss-Seidel step on the system (6.42) can be written as

$$\overline{x}_i = (A_{ii})^{-1} \left[b_i - \sum_{k<i} A_{ik}\, \overline{x}_k - \sum_{k>i} A_{ik}\, x_k \right]. \tag{6.45}$$

Denoting by $b_{R,\,i}$ and $b_{I,\,i}$ the real and imaginary part of the i-th component of the vector b, this smoothing step is in real arithmetic equivalent to

$$\begin{aligned} \begin{pmatrix} \overline{x}_{R,\,i} \\ \overline{x}_{I,\,i} \end{pmatrix} = & \begin{pmatrix} A_{R,\,ii} & -A_{I,\,ii} \\ A_{I,\,ii} & A_{R,\,ii} \end{pmatrix}^{-1} \left[\begin{pmatrix} b_{R,\,i} \\ b_{I,\,i} \end{pmatrix} \right. \\ & - \sum_{k<i} \begin{pmatrix} A_{R,\,ik} & -A_{I,\,ik} \\ A_{I,\,ik} & A_{R,\,ik} \end{pmatrix} \begin{pmatrix} \overline{x}_{R,\,k} \\ \overline{x}_{I,\,k} \end{pmatrix} \\ & \left. - \sum_{k>i} \begin{pmatrix} A_{R,\,ik} & -A_{I,\,ik} \\ A_{I,\,ik} & A_{R,\,ik} \end{pmatrix} \begin{pmatrix} x_{R,\,k} \\ x_{I,\,k} \end{pmatrix} \right]. \end{aligned} \tag{6.46}$$

The user can instruct SAMG to perform this smoother by specifying 2×2 block smoothers. Each block couples the PDE for the real and imaginary part of the vector potential in a finite element mesh point.

The equivalent real formulation used in combination with a system AMG code that is unaware of the time-harmonic nature of the PDE problem has several disadvantages. The matrix (4.72) requires twice the memory of the matrix in complex form (6.43). Not only the system matrix, but also the coarse grid hierarchy is stored twice by the system AMG code. This increase in memory can be a limiting factor in treating larger problems. The system AMG code furthermore computes Galerkin products in the form

$$\begin{pmatrix} I_h^H & 0 \\ 0 & I_h^H \end{pmatrix} \begin{pmatrix} A_R^h & -A_I^h \\ A_I^h & A_R^h \end{pmatrix} \begin{pmatrix} I_H^h & 0 \\ 0 & I_H^h \end{pmatrix} = \begin{pmatrix} A_R^H & -A_I^H \\ A_I^H & A_R^H \end{pmatrix} \tag{6.47}$$

which take twice the number of floating point operations of the computation of the Galerkin products in the form (6.44). In doing away with the original complex formulation completely and treating the problem as a general real one, one also looses the symmetry. A Krylov subspace method for general non-symmetric problems is therefore required if one wishes to accelerate the AMG solver.

An alternative implementation consists in providing a layer of routines on top of the scalar AMG code. This layer of routines passes the real part A_R^h

of the matrix A^h to those AMG routines that compute the C/F splitting and the (real) interpolation operator I_H^h. The interpolation operator is then provided as input to the top layer routines for the computation of the Galerkin product (6.44) in complex form. Having the complex coarser grid matrix A^H available, the process is repeated recursively until the coarsest grid is reached. After the setup, the multigrid cycling can be performed by the top layer routines.

In this implementation the only AMG routines being used are those that compute the C/F splitting and the interpolation on a given level. It has the advantages of maintaining the complex symmetry of the problem and of avoiding the duplication of the required memory and floating point operations in the setup phase in passing to an equivalent real formulation.

We realized some hybrid version between the two implementations described. In that hybrid implementation the setup is performed using the equivalent real formulation. After the setup, the coarse grid and interpolation operators on all levels are passed to top layer routines. The equivalent real coarser grid operators are transformed to complex *symmetric* ones and used by the calling routines to perform cycling in complex arithmetic. In realizing the hybrid implementation, we developed an interface between `PETSc` and `SAMG` described in Chapter 8.

6.9 Numerical Examples for Stationary Fields

In this section the performance of AMG for solving stationary magnetic field problems is reported. Two examples are considered. In the first one, the influence of the problem size on the convergence of AMG is investigated. In the second, a problem with anti-periodic boundary conditions is considered.

Comparable numerical results can be found in literature. The performance of `AMG1R5` as a solver on regular structured grid problems is described in [95], and on both structured and unstructured grid problems in [26]. It is compared with other multigrid approaches in [48]. The performance of `SAMG` both as a solver and as a preconditioner on model and engineering problems is reported in [67, 108].

6.9.1 Scalability Study

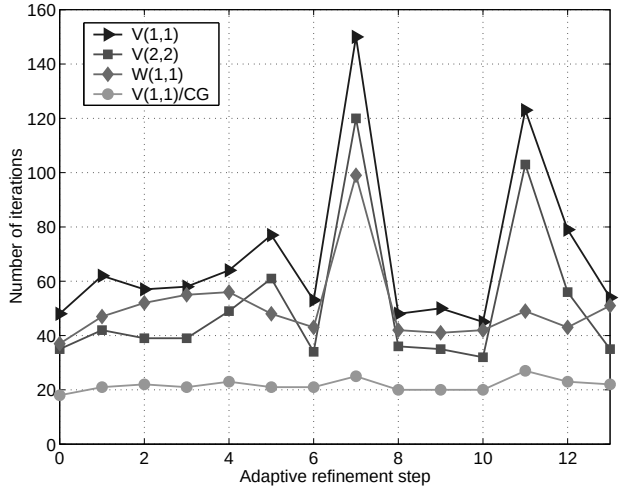
We evaluate the performance of both `AMG1R5` and `SAMG` for example problem **EP**-1 described in Section 3.7.1. A sequence of systems resulting from the Newton linearization has to be solved in each adaptive refinement step. We report on number of iterations, required CPU time and memory as functions of the number of grid points. We compare the CPU time required by AMG to converge with that of the SSOR preconditioned CG solver available in

PETSc. In using AMG as a solver, we consider the V(1,1), the V(2,2) as well as the W(1,1) cycle. In using AMG as a preconditioner for the CG algorithm, we only consider the V(1,1) cycle. In discussing the memory requirements of SAMG, we will also evaluate the aggressive coarsening strategy discussed in Section 6.6. As convergence criterion a reduction of the residual by a factor of 10^{-12} is taken.

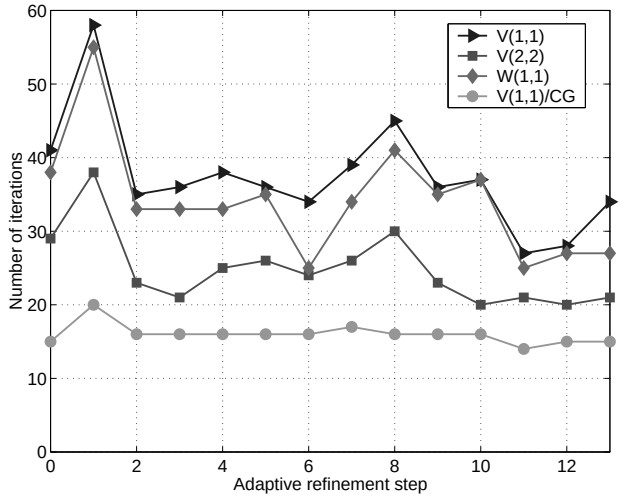
In Figure 6.3(a) and Figure 6.3(b) the number of iterations required by AMG1R5 and SAMG to reach convergence is plotted as a function of the adaptive refinement step respectively. The number of cycles for a particular mesh refinement step is the average of the number of cycles required in each Newton step.

From Figure 6.3(a), one can conclude that the number of iterations required by AMG1R5 as a solver using the V(1,1) cycle is independent of the number of grid points except for the peaks in the seventh and eleventh adaptive refinement step. This multigrid behavior is in sharp contrast the behavior of single level solver depicted in Figure 4.4. The peaks in Figure 6.3(a) can be explained by the fact that the C/F splitting and the interpolation are such that some frequency components of the error are not well captured. The number of V(1,1) cycles for example varies between 36 and 150. The average convergence factor for this cycle lies between 0.46 and 0.83. From the same figure one can also conclude that the gain in convergence speed by switching to computationally more expensive cycles like V(2,2) and the W(1,1) is limited. Using AMG1R5 as a preconditioner for the CG method however is beneficial. The number of iterations drops and becomes less sensitive to variations in the grid. Using AMG as a preconditioner requires per iteration one matrix-vector multiplication involving the fine grid matrix and a few inner products more than using AMG as a preconditioner. These operations constitute only a small fraction of the total computational cost of one AMG cycle. In discussing the CPU time measurements, we will see that the reduction in number of iterations outweighs the fact that each iteration is computationally more expensive.

From Figure 6.3(b), we can draw similar conclusions for the number of iterations of SAMG used as a solver and as a preconditioner. The number of V(1,1) cycles varies between 34 and 58, and the average convergence factor between 0.44 and 0.62. In comparing Figure 6.3(a) and Figure 6.3(b), one sees that the convergence of SAMG is less sensitive to variations in the grid than that of AMG. The reason for this is the higher quality of the interpolation in SAMG. The averaging of the number of iterations in a given refinement step hides the behavior of AMG in solving the individual Newton steps. We therefore plotted in Figure 6.4(a) the number of AMG1R5 iterations for every Newton step on the coarsest grid. In Figure 6.4(b) we did the same for SAMG. From these figures we conclude that the number of iterations for both AMG1R5 and SAMG remains bounded, and that the CG algorithm

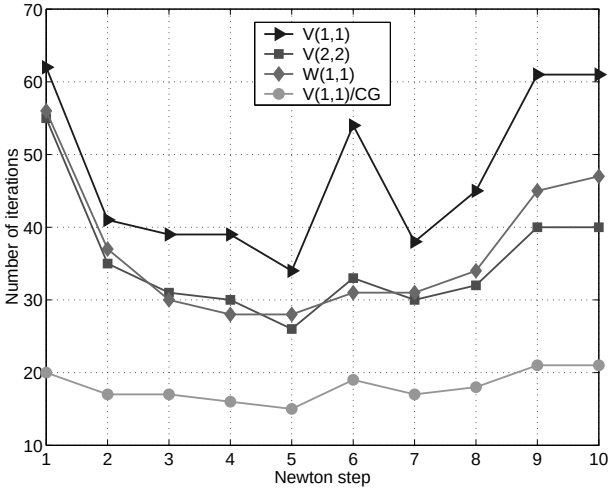


(a) Results for AMG1R5.

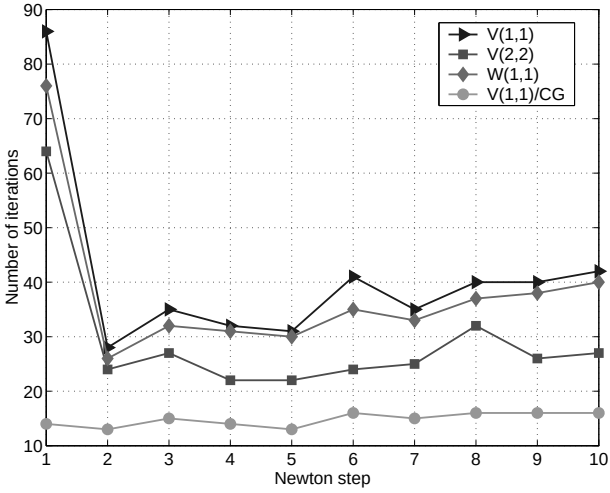


(b) Results for SAMG.

Figure 6.3: Number of iteration steps of AMG1R5 and SAMG versus the adaptive refinement step for example problem **EP**-1. The number of iterations shown for a particular refinement step is the average of the number of iterations in the individual Newton steps on a given mesh.



(a) Results for `AMG1R5`.



(b) Results for `SAMG`.

Figure 6.4: Number of iterations of `AMG1R5` and `SAMG` on initial mesh versus Newton step for example problem **EP**-1.

both stabilizes and accelerates convergence. Except for the first Newton step, SAMG requires less iterations than AMG1R5. Furthermore, SAMG is less sensitive to variations in the Jacobian than AMG1R5.

Next we compare the CPU time required by AMG1R5 and SAMG with the SSOR preconditioned CG algorithm. We will give CPU times of calculations on a 700 MHz Pentium 3 Linux workstation with 250 MB of RAM. The code was compiled using the GNU C++ and NAG Fortran90 compilers. In AMG we will consider the V(1,1) cycle only. In Figure 6.5 we plot the CPU time required by AMG1R5 as a solver and as a preconditioner and by the SSOR/CG solver versus the problem size on a logarithmic scale for both axes. From this figure the order of computational complexity of the algorithms, i.e. the exponent α in

$$\text{CPU time} \sim (\text{problem size})^{\alpha} \tag{6.48}$$

can be determined. For the AMG1R5/CG and SSOR/CG algorithms we have the complexity equals 1.1 and 1.4 respectively. Figure 6.6 illustrates the speedup delivered by AMG1R5. This speedup becomes more significant as the problem becomes larger. On the finest grid the former solver is 7.5 times faster than the latter. Using AMG1R5 as a preconditioner further reduces the CPU time. On the finest grid AMG1R5 as a preconditioner is twice as fast as AMG1R5 as a solver, i.e. 15 times faster that the SSOR-based solver. This SSOR-based solver was the solver implemented in the ESAT/ELEN code before the start of this thesis.

In Figure 6.7 the speed of AMG1R5 and SAMG both as a solver and as a preconditioner is compared. Using SAMG as a preconditioner delivers the fastest algorithm. On the finest grid SAMG used as a solver is faster than the SSOR/CG algorithm by a factor of 21.

To have an idea as of how the computer architecture and the compiler influence the CPU time measurements, we performed the same experiments on a Hewlett-Packard workstation with a PA-8500 processor at 400 MHz and 400 MB of RAM and on a SUN/Solaris workstation with an Ultra-Sparc 2 processor at 300 MHz and 400 MB of RAM. On these two systems, we use the vendor supplied compilers. There is virtually no difference in the results on the Hewlett-Packard and Linux systems. Calculations run slower on the SUN machine, but the amount of relative speedup delivered by AMG is about the same.

Finally we plotted in Figure 6.8 the memory requirements of AMG1R5 and SAMG with standard and with aggressive coarsening. This figure illustrates that SAMG with standard coarsening requires roughly half the memory of that of AMG1R5. With aggressive coarsening, memory requirements are halved once again. On the finest grid AMG1R5 builds a hierarchy of 15 coarser grids. SAMG with standard and with aggressive coarsening builds an hierarchy of 7

and 6 grids respectively. The price for the reduced memory consumption in aggressive coarsening is an increase in the number of iterations. Figure 6.9 illustrates that aggressive coarsening requires roughly double the number of iterations of standard coarsening. From Figure 6.10, we conclude that for the problem sizes considered, `SAMG` with standard or aggressive coarsening is equally fast to converge in CPU time. The increase in the number of iterations is compensated by each iteration being computationally less expensive.

6.9.2 Anti-Periodic Boundary Conditions

In the second numerical example we illustrate the effect of anti-periodic boundary conditions on the convergence of the classical Brandt-Ruge-Stüben algorithm. As an example we take example problem **EP**-2 described in Section 3.7.2. Results of computations done on a mesh obtained after 6 adaptive refinement steps with a total of 8084 elements and 4141 nodes will be analyzed in more detail. Table 6.1 however shows that the convergence characteristics we present in this example are mesh size independent.

The classical algorithm applied as a solver fails to converge as shown in Figure 6.11. In Table 6.1 we labelled the number of iterations by "div"if the solver failed to converge in less than 200 iterations. In the classical approach the positive connections across the anti-periodic boundary are erroneously treated as being weak, and the resulting interpolation is not properly constructed (see Section 6.5.4).

The classical algorithm can be forced to converge by using it as a preconditioner as shown in Table 6.1. Figure 6.11 shows that the accelerated classical algorithm requires a few iterations before convergence sets in.

To understand the behavior of AMG as a preconditioner in this problem, we investigated the convergence of the Ritz values of the iteration matrix $I - (Q_{h,H})^{-1}A$ (cfr. 5.10) towards the eigenvalues. To keep the computational cost of this analysis low, we performed computations on the initial mesh having 252 nodes and 453 elements. On this mesh the spectrum of the iteration matrix consist of a cluster of eigenvalues around 0 and two outliers at 0.80 and 0.59. The asymptotic convergence factor of AMG used as a solver is thus 0.80. The AMG/CG algorithm on the other hand requires 15 iterations to converge. In Figure 6.12 we plotted the Ritz values obtained in the successive CG iterations. These Ritz values could be obtained easily using the AMG-`PETSc` interface described in Chapter 8. As in [112], each sequence of Ritz values that seems to converge to a specific eigenvalue for increasing iteration count has been connected. One clearly sees that it only requires 4 and 5 CG iterations for the outermost and second outermost Ritz value to land close to 0.80 and 0.59 respectively. The outer CG iteration effectively removes the few outliers in the spectrum that prevents AMG from

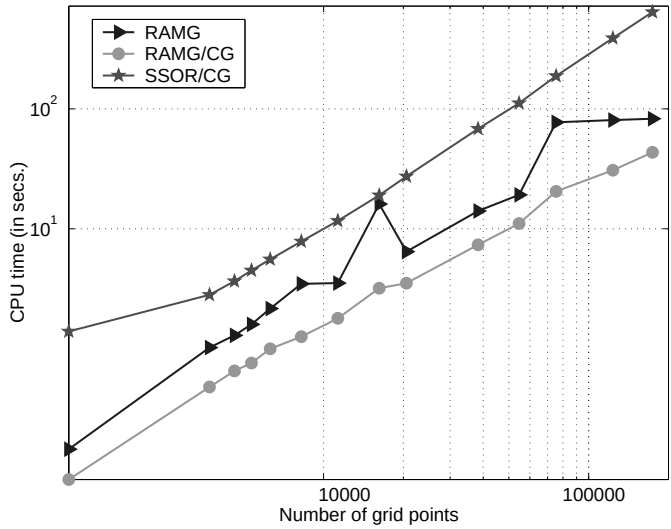


Figure 6.5: Logarithm of average CPU time required by `AMG1R5` as a solver and as a preconditioner and of SSOR/CG in each Newton step versus the logarithm of number of grid points for example problem **EP**-1.

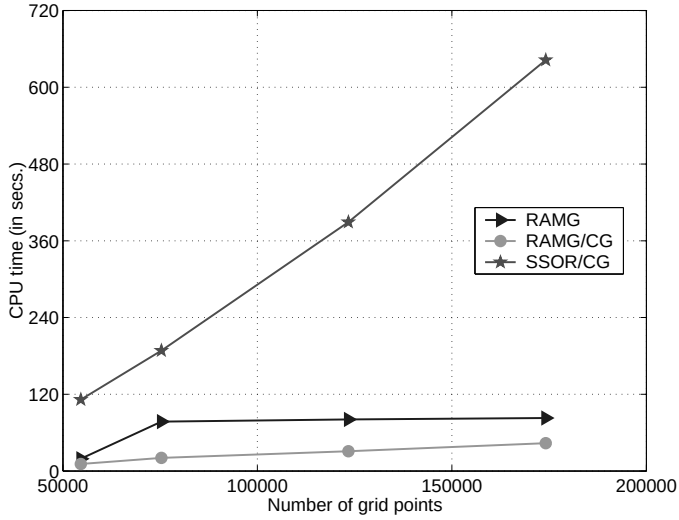


Figure 6.6: Average CPU time in each Newton step of `AMG1R5` as a solver and as a preconditioner and of SSOR/CG in last four adaptive refinement steps in example problem **EP**-1.

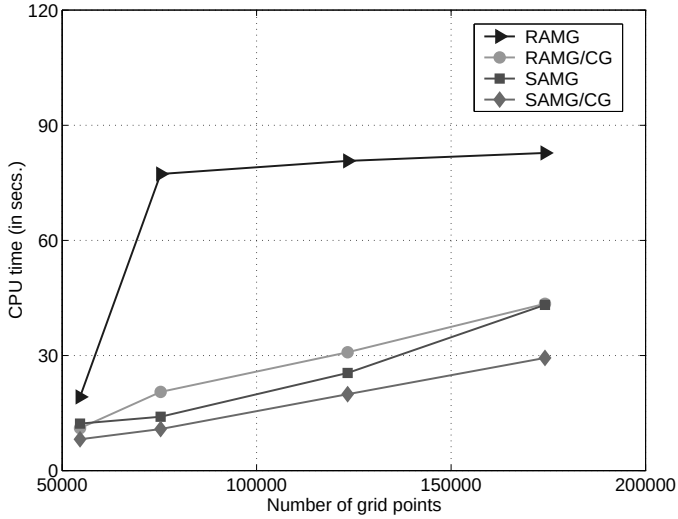


Figure 6.7: Average CPU time in each Newton step of `AMG1R5` and `SAMG` both as a solver and as a preconditioner versus number of grid points in last four adaptive refinement steps in example problem **EP**-1.

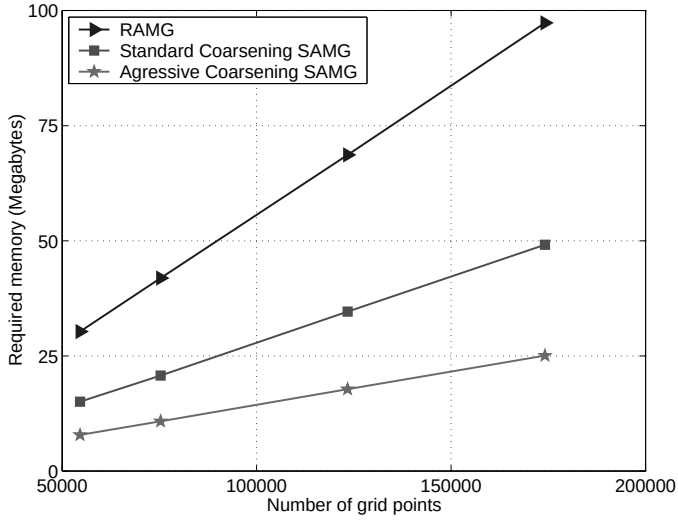


Figure 6.8: Average amount of memory in each Newton step required by `AMG1R5`, standard coarsening in `SAMG` and aggressive coarsening in `SAMG` versus the number of grid points in last four adaptive refinement steps in example problem **EP**-1.

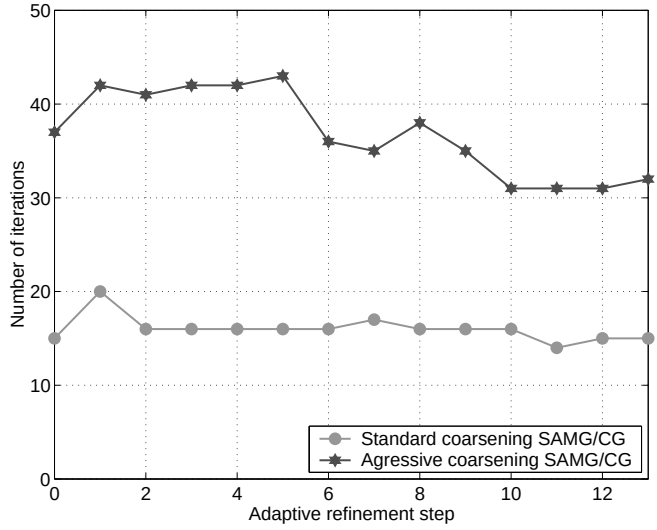


Figure 6.9: Average number of iterations in each Newton step of standard and aggressive coarsening in SAMG used as a preconditioner versus number of adaptive refinement step in example problem **EP**-1.

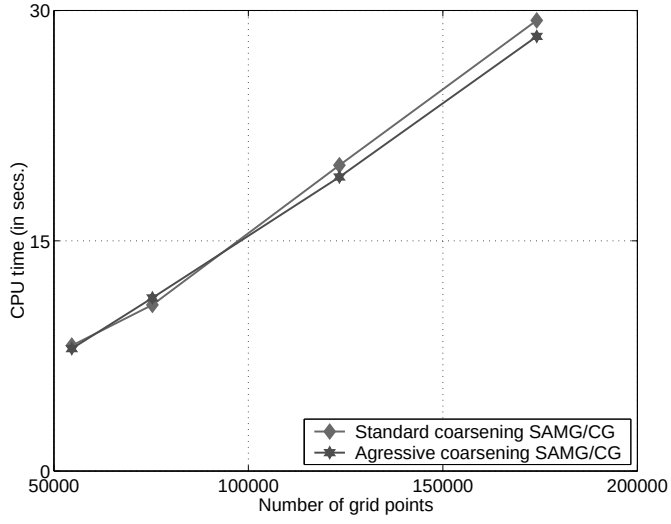


Figure 6.10: Average CPU time in each Newton step of standard and aggressive coarsening in SAMG used as a preconditioner versus number of grid points in last four adaptive refinement steps in example problem **EP**-1.

converging.

Another approach in making the classical algorithm converge is to take the positive connections properly into account as described in Section 6.5.4. This approach was implemented in SAMG, but is not a default part of the set-up algorithm. It is activated by a switch to be set by the user prior to calling the set-up phase. The two right-most columns in Table 6.1 give the number of iterations of SAMG as a solver and as a preconditioner with the switch for positive connections turned on. Figure 6.11 shows that both algorithms converge from the start of the iteration.

	Number of iterations			
Adaptive refinement step	classical AMG	classical AMG/CG	pos. connections AMG	pos. connections AMG/CG
0	148	15	76	14
1	div	16	27	14
2	div	17	43	15
3	div	17	28	14
4	div	18	47	16
5	div	17	36	15
6	div	17	27	14

Table 6.1: Number of iterations versus refinement step of classical AMG and AMG with provision for positive connections in example problem **EP**-2.

6.10 Numerical Examples for Time-Harmonic Fields

We now present numerical results obtained with the system version of SAMG in order to solve the time-harmonic example problem **EP**-3 described in Section 3.7.3. In this example we use the SAMG V(1,1) cycle as a preconditioner for the BiCGSTAB algorithm. The latter is available in the SAMG package. In Figure 6.13 we compare the CPU time required by SAMG used as a preconditioner with that required by the ILU preconditioned COCG solver in PETSc. In this experiment we use SAMG with standard coarsening. Figure 6.13 shows a computational speedup that increases with increasing mesh size. On the finest grid the use of SAMG results in six-fold reduction of CPU time. The number of iterations that SAMG with both standard and aggressive coarsening takes at each mesh refinement step in shown in Figure 6.14. This figure shows mesh width independent convergence for both coarsening strategies. In Figure 6.15 we report on the amount of memory required by both standard and aggressive coarsening in SAMG. Using aggressive coarsening roughly halves the memory requirements.

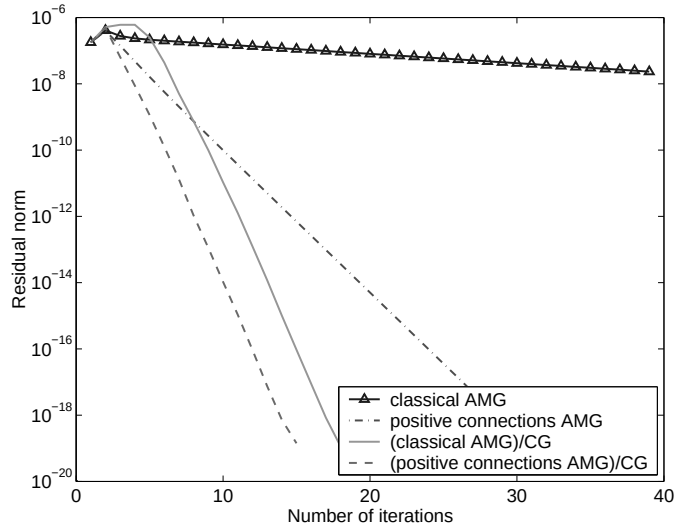


Figure 6.11: Convergence histories of classical AMG and AMG with provision for strong positive connections both as a solver and as a preconditioner on a mesh obtained after 6 refinement steps in example problem **EP**-2.

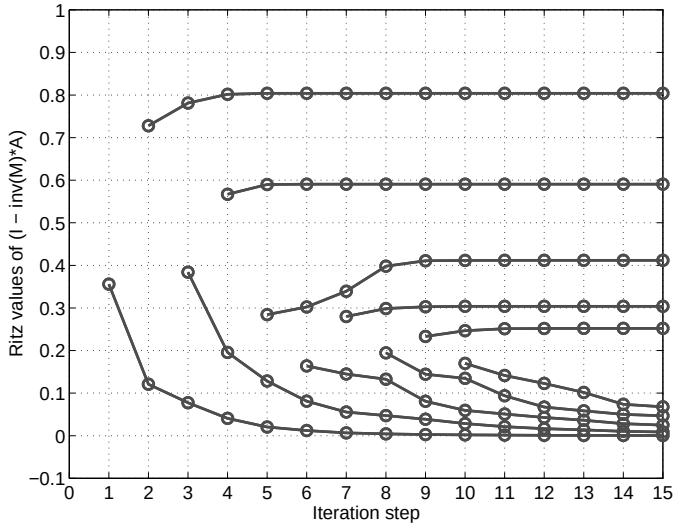


Figure 6.12: The location of the Ritz values of the operator $I - (Q_{h,H})^{-1}A$ in successive iteration steps of the CG algorithm on the initial mesh in example problem **EP**-2.

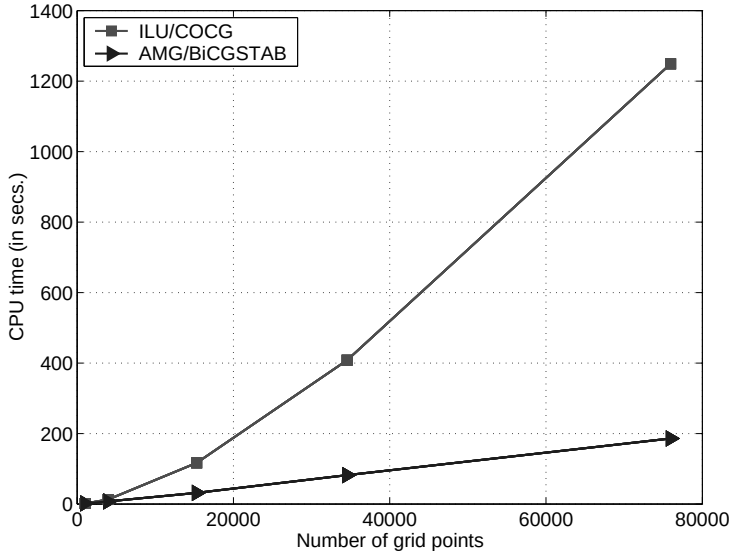


Figure 6.13: CPU time of SAMG preconditioned BiCGSTAB and ILU preconditioned COCG versus number of grid points in example problem **EP**-3. Timings in this figure were performed on a SUN/Solaris machine.

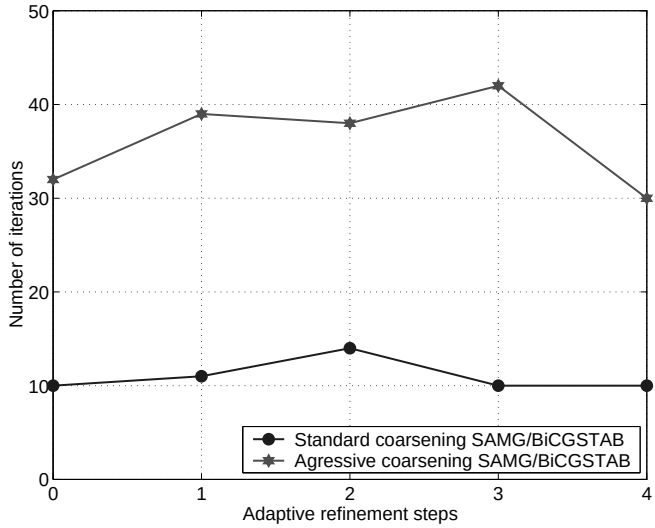


Figure 6.14: Number of iterations of SAMG with standard and aggressive coarsening versus mesh refinement step in example problem **EP-3**.

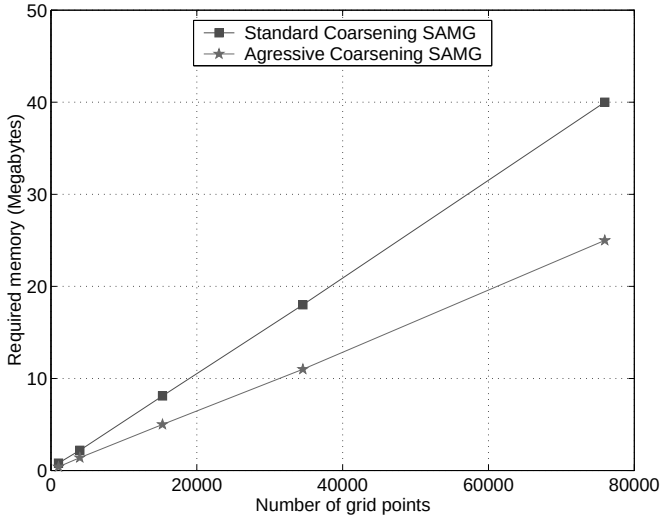


Figure 6.15: Memory requirements of SAMG with standard and aggressive coarsening versus mesh refinement step in example problem **EP-3**.

Chapter 7

Solving Field-Circuit Coupled Systems

7.1 Introduction

First some notations are introduced. As before, h denotesa typical measure for the finite element mesh width. In the right-hand side of (3.69) the electrical (mesh width independent) source terms χg will be denoted by g. In the matrix of (3.69) we set $-F$ and χS equal to B^h and C respectively. The discrete vector potential at the mesh points and the unknown currents and voltages in the circuit will be denoted by x_h and y respectively. The system resulting from the hybrid field-circuit coupled discretization discussed in Section 3.3.4 can then be written as

$$\mathcal{A}^h \begin{pmatrix} x_h \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ g \end{pmatrix} . \tag{7.1}$$

The matrix $\mathcal{A}^h$ is complex symmetric and has the following block structure

$$\mathcal{A}^h = \begin{pmatrix} A^h & B^h \\ (B^h)^T & C \end{pmatrix} . \tag{7.2}$$

The submatrices A^h, B^h and C represent the finite element discretization of the Helmholtz operator, the field-circuit coupling terms and the electrical circuit respectively.

The system (7.1)-(7.2) has the following properties:

- the matrix C remains unchanged in both size and value as the mesh is refined. Its size is small relative to that of A^h. In typical application the size of A^h and C ranges between 100 and 10^6 and between 1 and 500 respectively.

- the coupling matrix B^h involves integrals of the vector potential over cross-sections of solid and stranded conductors. Its sparsity structure is very different from that of A^h. Depending on the problem, the matrix B^h is sparse with dense blocks or dense.
- the field-circuit coupling is performed in such a way that no fill-in occurs in A^h. The latter diagonal block therefore corresponds exactly to the matrices considered in the previous chapter.

The presence of the blocks B^h and C destroys the M-matrix structure of the real part of the system matrix and thus hampers the immediate application of AMG in solving the system (7.1)-(7.2).

In this chapter we first investigate how the presence of the circuit relations influences the convergence of preconditioned Krylov subspace solvers. In developing better performing iterative schemes, we first tried to reuse the `SAMG` solver introduced in the previous chapter without altering the black box nature of the code. We did so by incorporating this code in block preconditioning and Schur complement schemes.

In an attempt to improve on this approach, we developed a generalized AMG scheme. In this generalized scheme the sequence of coarser grids is built based on the real part of the submatrix A^h. The matrices B^h and C are taken into account on the coarsest grid and in the cycling phase. This scheme is a generalization of a multigrid scheme for a PDE augmented by a scalar equation. It only requires the fine level system as input, and is therefore algebraic. The scheme was incorporated in the ESAT/ELEN code and proved to be efficient and robust in practical computations.

For its implementation we extended the AMG-`PETSc` interface. This allowed us to make extensive use of the profiling tools in `PETSc` in order to optimize the cycling phase. Details will be given in Chapter 8.

7.2 Circuit Relations and Convergence of Krylov Methods

Before developing iterative solvers for the coupled problem, the influence of the circuit relations on the convergence of preconditioned Krylov subspace solvers applied to the system (7.1)-(7.2) is studied. This is done by comparing the number of iterations required to solve a given finite element model coupled to different electrical circuit models. Given an electromagnetical device, different electrical circuit models can be constructed by, for example, interchanging the current and voltage sources, or by modeling the electrical circuit with different degrees of accuracy.

Numerical results show that it is hard to quantify the effect of the presence of the circuit relations on the convergence of the Krylov subspace sol-

vers. The circuit relations do *not* necessarily adversely affect convergence however. Examples have been found for which convergence for a model with circuit relations was faster than the same finite element model without circuit relations. In order to illustrate these points, we plotted in Figure 7.1 the number of SSOR preconditioned SQMR iterations versus the number of nodes for a given finite element model of an induction machine coupled with different electrical circuits. In this figure the different electrical circuits are labelled by $[c, \ell]$, where c and ℓ denote the number of cutset and loop equations in the circuit respectively (see Section 2.8). Note that the number of nodes inserted at each stage of the adaptive mesh refinement step is different for the different circuit topologies. The figure shows that on the finest mesh the model with no cutset and two loop equations requires 714 iterations to converge for example, while the model with no circuit equation requires 1920 iterations.

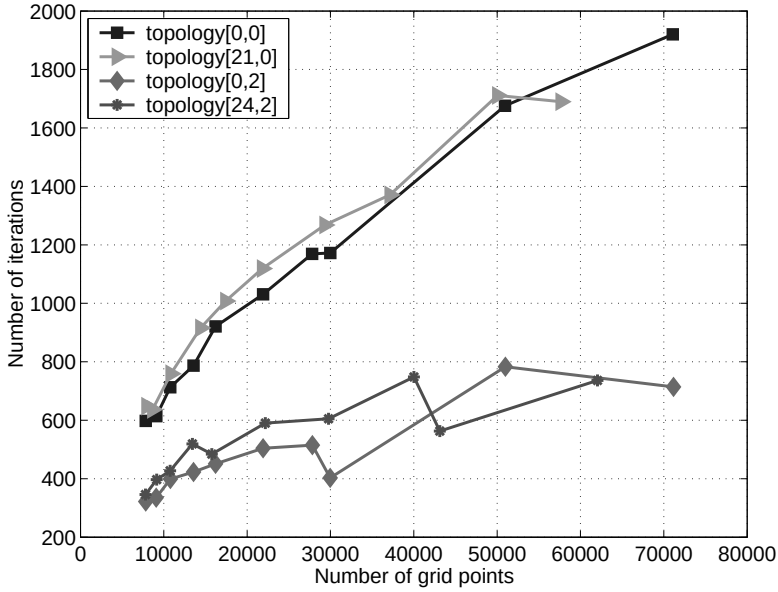


Figure 7.1: Number of SSOR preconditioned SQMR iterations versus the number of nodes for a given finite element model of an induction machine coupled with different electrical circuits.

7.3 Block Preconditioning Approaches

A first approach in reusing SAMG for solving (7.1)-(7.2) is the use of *block* preconditioning techniques. These preconditioners have been considered in the context of solving saddle point problems arising in different contexts [96, 121, 102, 24, 66]. They also serve as a starting point in developing overlapping domain decomposition methods [23, 104, 91].

The block Jacobi scheme

$$\begin{pmatrix} A^h & 0 \\ 0 & C \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix} \tag{7.3}$$

can be used as a preconditioner for the COCG or SQMR algorithms. The matrix B^h can be taken more directly into account by switching to the non-symmetric Gauss-Seidel preconditioner

$$\begin{pmatrix} A^h & 0 \\ (B^h)^T & C \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix} . \tag{7.4}$$

for Krylov subspace methods for non-symmetric problems such as GMRES or BiCGSTAB. The Gauss-Seidel preconditioning step is more expensive than that of Jacobi by the multiplication of the matrix B^h only. The application of the Jacobi and Gauss-Seidel schemes requires solving a linear system with coefficient matrices A^h and C at every step of the Krylov subspace method. One thus obtains an inner-outer iterative scheme. The inner scheme is the solving of the A^h and C linear systems and the outer scheme is the Krylov subspace iteration. As the size of C is much smaller than that of A^h, the cost of solving the C linear system is negligible compared to solving the A^h linear system. In solving the A^h system, AMG can be reused. The system can be solved either exactly or approximately by applying a few (but fixed) number of cycles. Another alternative consists in making the accuracy of the inner AMG solve function of the residual norm of the outer Krylov iteration. The resulting variable preconditioner can be accelerated by FGMRES (see Section 4.7) for instance.

Numerical results for the block Jacobi scheme will be presented in Section 7.6.

7.4 Schur Complement Approaches

Another approach in reusing SAMG in solving (7.1)-(7.2) is through the use of *Schur complements*. Schur complement methods are related to the Uzawa scheme for solving saddle point problems [119, 9, 37, 17] and to a class of domain decomposition methods called iterative substructuring methods [23, 104].

Using block elimination, the system (7.1) can be written equivalently as

$$S_C^h \, x_h = -B^h \, C^{-1} \, g \tag{7.5}$$

$$y = C^{-1} \left[g - (B^h)^T \, x_h \right] , \tag{7.6}$$

with the matrix S_C^h given by

$$S_C^h = A^h - B^h \, C^{-1} \, (B^h)^T . \tag{7.7}$$

This matrix is called the Schur complement of C in $\mathcal{A}^h$ [46]. Once system (7.5) has been solved for x^h, the dimension of C allows to solve (7.6) for y by a direct solver. We can therefore focus on solving the Schur complement system (7.6) from here on.

The complex symmetry of S_C^h allows to solve this system using the COCG or SQMR algorithms as outer Krylov iteration. The matrix-vector product with matrix S_C^h in each step of these algorithms requires solving a system with C as coefficient matrix. The LU factorization of C can be computed prior to the iteration. Each matrix-vector product with the matrix S_C^h then requires two triangular solves of dimension equal to that of C. The difficulty is that no preconditioner is immediately available for S_C^h. The approach we therefore propose consists in neglecting the $B^h \, C^{-1} \, (B^h)^T$ term in (7.7) and to precondition S_C^h with an approximation to A^h. This in particular allows to solve the preconditioning step in the COCG or SQMR algorithms by applying `SAMG` based on A^h.

By an argument similar to the derivation of (7.5)-(7.6), the system (7.1) can also be written as

$$S_A^h \, y = g \tag{7.8}$$

$$x_h = -(A^h)^{-1} \, B^h \, y \, , \tag{7.9}$$

where the Schur complement of A^h in $\mathcal{A}^h$ is given by

$$S_A^h = C - (B^h)^T \, (A^h)^{-1} \, B^h \, . \tag{7.10}$$

Solving the linear system (7.8) by Krylov subspace methods is complicated by the fact that the matrix-vector product with matrix S_A^h requires solving a linear system with A^h as coefficient matrix.

The Schur complement appproach has not been persued in this thesis.

7.5 Multigrid Scheme

Our multigrid technique for solving field-circuit coupled problems is a generalization of the method for solving an elliptic problem augmented by an algebraic equation, see e.g. [53, Section 11.4].

Let the linear system (7.1) with coefficient matrix $\mathcal{A}_h$ defined in (7.2) be a fine grid discretization of the coupled problem. Let Ω^h denote the set of fine grid degrees of freedom. The set Ω^h embraces both the magnetic and electric variables, i.e. the variables x_h and y. Denoting the former and the latter by Ω_M^h and Ω_E^h respectively, yields

$$\Omega^h = \Omega_M^h \cup \Omega_E^h \,. \tag{7.11}$$

Each variable in Ω_M^h corresponds to a finite element mesh point. The number of elements of Ω_E^h equals the number of loop and cutset equations.

Next we describe a generalized AMG algorithm for solving the the given linear system. We will first give details of the two-grid variant of our algorithm. In doing so, we discuss the set-up and the solve phase separately.

In the set-up phase, the set Ω^h is coarsened in such a way that the electric degrees of freedom Ω_E^h are in the coarse grid, i.e.,

$$\Omega_E^h \subset \Omega^H \,. \tag{7.12}$$

The magnetic degrees of freedom Ω_M^h are split into coarse and fine grid ones denoted by C_M^h and F_M^h respectively. The magnetic coarse grid Ω_M^H is identified with C_M^h. AMG constructs this splitting and the magnetic interpolation I_H^h mapping from Ω_M^H to Ω_M^h using information contained in the real part of the first diagonal block of $\mathcal{A}_h$, i.e. in the real part of the discrete differential operator A_h. The next coarser grid Ω^H and the interpolation operator $\mathcal{I}_H^h$ for the coupled problem can now be introduced. The set Ω^H is defined as follows

$$\Omega^H = \Omega_M^H \cup \Omega_E^h \,. \tag{7.13}$$

This definition implies that the interpolation $\mathcal{I}_H^h$ has the following block diagonal form

$$\mathcal{I}_H^h = \begin{pmatrix} I_H^h & 0 \\ 0 & I \end{pmatrix} , \tag{7.14}$$

where the second diagonal block denotes the identity on Ω_E^h. The symmetry of $\mathcal{A}_h$ motivates again to define the restriction $\mathcal{I}_h^H$ as the transpose of the interpolation. The coarse grid equivalent of $\mathcal{A}_h$ is computed by a Galerkin product, resulting in

$$\mathcal{A}_H = \mathcal{I}_h^H \, \mathcal{A}_h \, \mathcal{I}_H^h = \begin{pmatrix} A_H & B^H \\ (B^H)^T & C \end{pmatrix} , \tag{7.15}$$

with $A_H = I_h^H \, A_h \, I_H^h$ and

$$B^H = I_h^H \, B^h \,. \tag{7.16}$$

In the solve phase, we perform smoothing on the magnetic variables only. Smoothing leaves the electric variables unchanged. Given a right-hand side vector $(f^h,\, g)$ and a starting solution $(x_0^h,\, y_0)$ for the linear system (7.1), smoothing consists of computing a modified magnetic right-hand side vector

$$\overline{f}^h = f^h - B^h\, y_0\,, \tag{7.17}$$

and applying Gauss-Seidel smoothing to the system

$$A^h\, x^h = \overline{f}^h\,. \tag{7.18}$$

Let $\overline{x}^h$ denote the iterand in the field variables after smoothing. After smoothing, the fine grid residual is computed according to

$$\begin{array}{l} r_1^h = \overline{f}^h - A^h\, x^h \\ r_2^h = g - (B^h)^T\, \overline{x}^h - C\, y_0 \end{array}\quad, \tag{7.19}$$

and restricted to the coarser grid. The coarse grid correction is computed by solving the linear system with matrix (7.15) by a direct solver.

If the two-grid scheme is applied recursively to solve this coarse grid system, a multi-grid scheme is obtained. This multigrid scheme can be applied as a preconditioner for the COCG and SQMR algorithms.

7.6 Numerical Example

In this section we present some numerical results for the proposed schemes for field-circuit coupled problems. As a test case, we use example problem **EP**-4 presented in Section 3.7.4. We will compare the block Jacobi and generalized AMG schemes with ILU(0) preconditioning applied to the global matrix $\mathcal{A}^h$. These three schemes are accelerated by an outer COCG iteration. In the first two schemes `SAMG` with standard coarsening is used in the set-up phase. In the block Jacobi algorithm the A^h linear system is solved approximately by applying just *one* $V(1,1)$ cycle. In the generalized AMG algorithm' $V(1,1)$ cycling on the coupled system is performed.

Details on the implementation of the block Jacobi and generalized AMG schemes are given in Chapter 8. The block Jacobi scheme was implemented using the level 1 `SAMG-PETSc` interface described in Section 8.2. The multigrid cycling is carried out in real arithmetic by `SAMG`. The matrix C is LU factored by `PETSc` prior to the iteration. The generalized AMG scheme was implemented as an extension of the level 2 `SAMG-PETSc` interface described in Section 8.3. This in particular implies that multigrid cycling in the generalized AMG scheme is performed in complex arithmetic by `PETSc`. The

matrix $\mathcal{A}^H$ on the coarsest level is assembled and LU factored prior to the iteration.

The block Jacobi and generalized AMG schemes differ in computational complexity by the matrix-vector multiplication involving the matrix B^H on all coarser levels required in (7.17) and (7.19). Numerical results in example problem **EP**-4 have shown that the operation (7.16) introduces fill-in the matrix B^H is such a way that the matrices B^H become dense quickly on the coarser levels. In an efficient implementation of the generalized AMG scheme, it is very important to that this fact into account. In declaring the matrices B^H as *dense* in `PETSc` one avoids the time-consuming repeated calls to memory allocation routines in constructing the matrix. In defining the matrix as dense, `PETSc` furthermore calls BLAS routines for the matrix-vector multiplication.

In Table 7.1 we listed the number of block Jacobi, generalized AMG and ILU(0) preconditioned COCG steps as a function of the number of the adaptive refinement step. For the first two algorithms the number of iterations is mesh width independent. For the block Jacobi scheme this can be explained by analogy with the Schwarz type domain decomposition algorithms (see e.g. [104, Section 1.4]). Table 7.1 also shows that the generalized AMG scheme is superior to the block Jacobi scheme in terms of number of iterations by a factor of about 2.

Adaptive refinement step	0	1	2	3
Generalized AMG / COCG	102	91	99	94
Block Jacobi / COCG	180	189	200	199
ILU(0) / COCG	4455	4361	5117	5858

Table 7.1: Number of iterations of the block Jacobi, generalized algebraic multigrid and ILU(0) preconditioned COCG schemes as a function of the adaptive refinement step.

In Figure 7.2 the CPU time of the block Jacobi and ILU(0) preconditioned COCG solvers are compared. Calculations were carried out on a SUN/Solaris workstation having an Ultra 2 processor at 300 MHz and 400 MB of RAM. This figure shows that the block Jacobi converges faster than the ILU(0)/COCG solver and the speedup becomes more pronounced with increasing mesh size. On the finest grid considered, block Jacobi speeds up calculations by a factor of 8. In Figure 7.3 the CPU time of the block Jacobi and generalized AMG preconditioned COCG solvers are compared. This figure shows that the generalized AMG scheme is even faster to converge than block Jacobi. On the finest grid the former is faster than the latter by a factor of 3. Compared with the ILU(0) preconditioner, the use of AMG results in a speedup on the finest grid by a factor of 24!

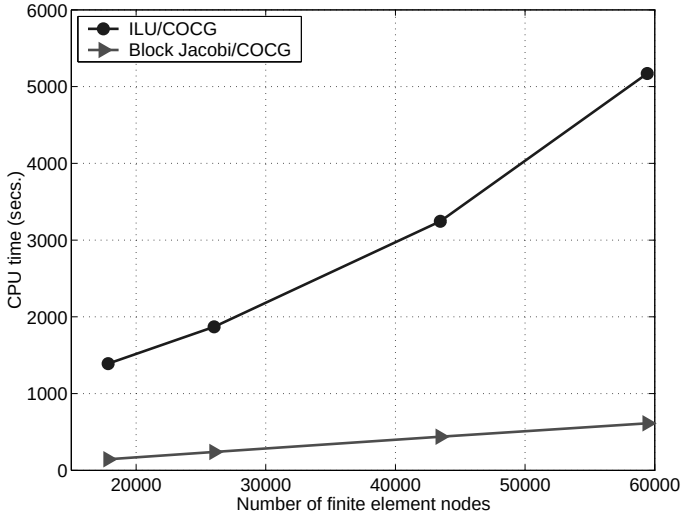


Figure 7.2: CPU time of block Jacobi and ILU preconditioned COCG method for complex symmetric systems versus the number of finite element grid points in example problem **EP**-4.

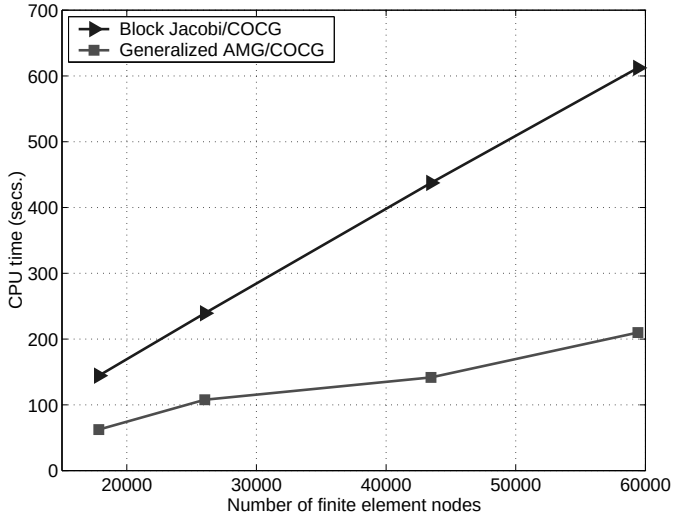


Figure 7.3: CPU time of block Jacobi and generalized AMG preconditioned COCG method for complex symmetric systems versus the number of finite element grid points in example problem **EP**-4.

Chapter 8

The AMG-PETSc interface

8.1 An Overview of PETSc

The Portable, Extensible Toolkit for Scientific Computations (PETSc) is a public domain software library for solving discretized PDEs. It is object oriented in design and implemented in C. We give a short overview of the package. More detailed information can be found in the manual [7] and in the online documentation [6].

The package's most basic objects are vectors (Vec) and matrices (Mat). Different formats for e.g. sparse, block-structured and dense matrices are provided. An example of an elementary operation on these objects is VecSetValues and MatSetValues. These routines assign a set of values to specific locations in the corresponding object. For efficiency, more advanced operations use BLAS and LAPACK routines whenever possible. Vectors and matrices can be either real or complex valued. They can be assigned to a single processor or distributed across several processors. To support parallel computing, objects in PETSc are built on top of the Message Passage Interface (MPI) (see e.g. [51]).

Using these objects as building blocks, the system of linear equations solver object (SLES) is implemented. Both direct as well as preconditioned Krylov subspace methods are available. The Krylov subspace object (KSP) accommodates a large suite of Krylov subspace solvers, including all those discussed in Chapter 4 except the SQMR algorithm. Preconditioning schemes accessible from within the preconditioning object (PC) include basic matrix splitting, incomplete factorization, multigrid and Schwarz type domain decomposition algorithms. All linear solvers are accessible through a uniform interface, facilitating the experimentation with different algorithms.

The system of non-linear equations solver (SNES) and the time stepping

(TS) object are built on top of the SLES object. PETSc provides support for the automatic profiling of floating point and memory usage and for intensive error checking.

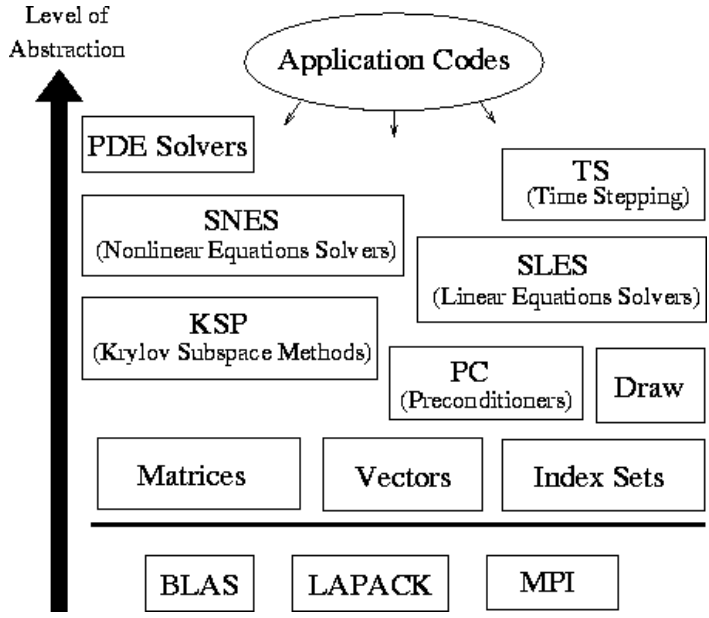


Figure 8.1: An overview of the PETSc library.

8.2 Level 1 AMG-PETSc Interface

The extensibility of PETSc allows users to easily add new objects to the package. For example, new preconditioners can be defined. In developing an interface between AMG and PETSc, we started by making AMG available as a preconditioner from within PETSc. We will call this interface the *level 1* AMG-PETSc interface. We implemented this interface for AMG1R5 and SAMG, and in the case of SAMG for both stationary and time-harmonic problems.

The definition of a preconditioner in PETSc is broken up into four stages. In PCCreate required data structures are created. These data structures are destroyed in PCDestroy when no longer required. In PCSetUp, operators defining the precondioner are computed before the Krylov iteration starts. In each Krylov iteration step, the preconditioner is applied by calling PCApply.

In the level 1 interface, we implemented these four steps for AMG. In

PCSetUp the setup phase of AMG is called. In PCApply, AMG cycling is performed.

The level 1 interface makes it possible to accelerate AMG by the Krylov subspace solvers available in PETSc. In accelerating AMG by the CG algorithm, the Ritz values can easily be extracted. The level 1 interface allowed us to implement the block-preconditioning strategies described in Section 7.3.

8.3 Level 2 AMG-PETSc Interface

In the level 1 interface, the multigrid cycling is performed by AMG. In order to obtain a more flexible environment in which we could adapt the cycling to our needs, we extended the level 1 to the *level 2* interface. In the latter cycling is performed by objects in PETSc. This interface was only implemented for SAMG for both stationary and time-harmonic problems.

The multigrid objects in PETSc provide an abstract algebraic framework for multigrid cycling. Their implementation is independent of any knowledge about the geometry or the discretization of the differential problem. They require as input a sequence of coarser grid matrices, restriction and interpolation matrices along with the necessary vectors. This information can be provided by the set-up phase of AMG. In this perspective it is natural to interface AMG with the multigrid objects in PETSc.

In the level 2 interface, PETSc passes the fine grid matrix to the AMG setup phase by calling PCSetUp. AMG computes the C/F splitting, the interpolation and coarse grid matrices on all levels. These operators are fetched from AMG common blocks and stored into PETSc matrices. These matrices are passed to the multigrid objects to perform the cycling.

The level 2 interface has all the functionality of the level 1 interface. The level 2 interface offers the additional advantage of having the coarser grid and intergrid transfer matrices available as PETSc matrices. The non-zero structure of these matrices for example can therefore easily be visualized.

The code for the level 1 interface with RAMG1R5 is available to other PETSc users as it was incorporated into the package. The code for the level 1 and level 2 interface with SAMG will be made available in the future.

8.4 Extension of the Level 2 AMG-PETSc Interface for Field-Circuit Coupled Problems

The level 2 interface provides a framework to implement the generalized multigrid algorithm for field-circuit coupled problems described in Section 7.5. We will give some details of the implementation.

The fine grid matrix $\mathcal{A}^h$ defined in (7.2) is stored in terms of the blocks A^h, B^h and C. Passing only A^h to the AMG setup routine by calling `PCSetUp` can therefore easily be done. After the setup, the coarse grid and interpolation operators A^H and I_h^H are made available as `PETSc` matrices. The coupling matrices B^H on the coarser levels are computed by a matrix-matrix multiplication involving the transpose of I_h^H and B^h. The electrical circuit matrix and the discretized PDE and coupling terms on all coarser levels are then passed to the solve phase for multigrid cycling.

In the `PETSc` multigrid framework we added a routine for the cycling on the coupled problem. This routine is an extension of the routine for standard cycling and is given below. This routine calls three routines that are particular for field-circuit coupled problems. The routine `MGFCUpdateRhs` updates the right-hand side vector for the field variables according to (7.17). The routine `MGFCCalculateResidual` computes the residual in the field variables and the electric variables according to (7.19). The routine `MGFCCoarsestGridSolve` does the direct linear solve involving the matrix (7.15) on the coarsest level. It is assembled from its blocks A^H, B^H and C and LU factored prior to the multigrid iteration.

```
int MGFieldCircuitCycle_Private(MG *mglevels)
  {
    MG     mg = *mglevels;
    int    levels = mg->levels;
    int    cycles = mg->cycles,ierr,its;
    double rnorm;
    Scalar zero = 0.0, done = 1.0, dminone = -1.0;
    Vec    r;

    PetscFunctionBegin;
    if (!mg->level) {
      /*..do solve on coarsest grid..*/
      ierr = MGFCCoarsestGridSolve(mg); CHKERRQ(ierr);
    } else {
      while (cycles--) {
        /*..update rhs vector in x-variables..*/
        ierr = MGFCUpdateRhs(mg); CHKERRQ(ierr);
        /*..do pre-smoothing..*/
        ierr = SLESSolve(mg->smoothd,mg->upd_b,mg->x,&its);
               CHKERRQ(ierr);
        /*..compute residual in both x and y variables..*/
        ierr = MGFCCalculateResidual(mg);
        /*..restrict x-residual and store restricted residual into
          rhs of next coarser grid....*/
        ierr = MatRestrict(mg->restrct,mg->r,mgc->b);CHKERRQ(ierr);
        /*..set start solution for error correction eqns. in x..*/
        ierr = VecSet(&zero,mgc->x);CHKERRQ(ierr);
```

```
        /*..copy fine grid y-residual into rhs of next coarser grid..*/
        ierr = VecCopy(mg->r_y, mgc->b_y); CHKERRQ(ierr);
        /*..set start solution for error correction eqns. in y..*/
        ierr = VecSet(&zero,mgc->y);CHKERRQ(ierr);
        /*..recursive call to next coarser grid..*/
        ierr = MGFieldCircuitCycle_Private(mglevels-1);CHKERRQ(ierr);
        /*..interpolate and add correction in x..*/
        ierr = MatInterpolateAdd(mg->interpolate,mgc->x,mg->x,mg->x);
               CHKERRQ(ierr);
        /*..add correction in y, i.e. overwrite y with y + e_y..*/
        ierr = VecAXPY(&done, mgc->y, mg->y); CHKERRQ(ierr);
        /*..update rhs vector in x-variables..*/
        ierr = MGFCUpdateRhs(mg); CHKERRQ(ierr);
        /*..do post-smooting..*/
        ierr = SLESSolve(mg->smoothu,mg->upd_b,mg->x,&its);
               CHKERRQ(ierr);
      }
    }
    PetscFunctionReturn(0);
  }
```

Chapter 9

Conclusions and Suggestions for Future Research

> Beato l'uomo che fa del proprio limite la propria forza. Beato l'uomo che non si ferma, che non si vuole arrendere. Beato l'uomo che non ha paura di se stesso, della propria fisicità. Beato l'uomo che sfida con amore il timore della terrena provvisorietà caricata di ogni tipo di diversità. Beato l'uomo che riesce a sorridere communque. Beato l'uomo totale. Beato l'uomo. [1]

9.1 Conclusions

The finite element discretization of time-harmonic models for electromagnetic energy transducers yields large systems of algebraic equations. This thesis showed that AMG techniques used in combination with Krylov subspace solvers are efficient in solving these systems, outperforming by far the more commonly used one-level iterative solvers. It also showed that AMG is not a particular code, but instead a general algorithm that can be adapted to the discretized problem at hand.

In this thesis solving discrete stationary magnetic field models by AMG was considered first. Afterwards the applicability of AMG was extended to solving quasi-stationary time-harmonic models without electrical circuit relations. Both the aspect of the multigrid iteration as well as the Krylov

[1] Mina in La Stampa, a Turin based daily

acceleration were studied in detail. Finally the AMG algorithm was generalized to solve field-circuit coupled problems. The generalisation of AMG for time-harmonic problems was partly developed during a one-month visit to the German National Center for Information Technologies in 1998. The code for the coupled problems was succesfully optimized in terms of CPU-time usage during a one-month visit to Argonne National Laboratory, USA, in 2001. The solvers developed in this thesis were incorporated in a simulation package in such a way as to allow their use in computing models of industrial relevance.

In solving *discrete stationary magnetic field problems without anti-periodic boundary conditions*, `RAMG` and its successor `SAMG` can be used as a black-box solver. For both codes the multigrid property that the number of iterations required for convergence remains bounded as the problem size increases was verified. Compared with an SSOR preconditioned CG algorithm, the use of the AMG codes as a solver results in a reduction of required CPU-time. The amount of gain in CPU-time grows with the problem size. Using the AMG codes as a preconditioner for the CG algorithm further enhances their robustness and speed. In the computation of a permanent magnet machine for example, the use of `RAMG` as solver and as preconditioner for a problem of about 175.000 unknowns, results in a speedup of a factor of 7,5 and 15 respectively. `SAMG` is even faster and requires less memory than `RAMG`. Compared with the SSOR precondioned CG algorithm, `SAMG` used as a solver and as a preconditioner delivers a speedup of a factor of 15 and 21 respectively.

The discretization of problems *with anti-periodic boundary conditions* yields coefficient matrices with large positive off-diagonal entries. `RAMG` fails to take these positive connections properly into account, causing the convergence of `RAMG` used as a solver to be hampered by a few outliers in the spectrum of the iteration matrix. These outliers can easily be captured by an outer CG iteration, and the convergence of `RAMG` used as a preconditioner is mesh-width independent as before. In `SAMG` the positive connections are properly taken into account in the construction of the interpolation.

The application of AMG was extended to solving *discrete time-harmonic problems* by providing only the real part of the fine level system matrix as input for the construction of the coarser grid and the interpolation. Given real-valued intergrid operators, the Galerkin product yields complex coarser grid matrices with similar structure and properties as the fine grid matrix. In the cycling phase, smoothing is performed based upon a splitting of complex-valued coefficient matrices. This algorithm was implemented in a system version of `SAMG` and accelerated by a Krylov subspace method for non-hermitian matrices. In the computation of an induction machine, the `SAMG` code used as a preconditioner on a mesh of almost 80.000 nodes was 6 times faster than an ILU preconditioned CG solver for complex symmetric matrices.

The generalization of AMG for solving *time-harmonic field-circuit coupled problems* relied on the 2×2 block structure of the coefficient matrix. In the generalized algorithm, the setup phase operates on the discretized PDE block of the matrix only. The electrical circuit and the coupling terms are taken into account in the cycling phase. In the computation of an induction machine, the use of the generalized AMG algorithm on a problem with about 60.000 field equations and 250 circuit equations resulted in a speedup of a factor of 24 compared with an ILU preconditioned CG solver.

For the implementation of the generalized AMG algorithm for field-circuit coupled systems, an interface was developed that allows to call the AMG codes from within the widely used software package `PETSc`. `PETSc`'s multigrid components were extended. The AMG-`PETSc` interface is such that `PETSc` calls the AMG setup phase and that `PETSc` performs multigrid cycling on the hierarchy of coarser levels constructed by AMG. The multigrid cycling can be accelerated by the Krylov subspace solvers available in `PETSc`. This interface was incorporated in `Olympos`. The coupling with `Olympos` generalizes previously developed interfaces between the AMG codes and `Olympos`, and between `PETSc` and `Olympos`.

9.2 Suggestions for Future Research

Optimization of the Non-Linear Solvers

In this thesis, the individual linearization steps in non-linear problems were solved to full accuracy. One can save in computational cost in the individual linear steps by imposing less stringent stopping criteria. The possible expense of this approach is a possible slowdown of the outer non-linear iteration. In Example Problem **EP**-1 for example more Newton steps are required if one solves each linear step approximately by applying only one AMG cycle. In inexact Newton methods [32, 33] one tries to balance the computational cost of the inner linear solves with the number of outer non-linear iterations. One strategy is to relate the accuracy of the linear solve with the norm of the non-linear residual. Another one is to avoid having to run the AMG setup phase in each linear solve by using the same multigrid hierarchy in several consecutive linear solves. The non-linear iteration can possibly be accelerated further by considering more advanced techniques to reach the region in which the non-linear iteration is guaranteed to converge.

As mentioned in Section 3.4.4, non-linear time-harmonic problems were solved by a Successive Substitution iteration in this thesis. This iteration converges linearly for initial guesses sufficiently close to the solution. This iteration can be accelerated by replacing it with Newton's method that converges quadratically locally. To obtain a globally convergent method,

suitable damping techniques need to be incorporated. Replacing the Successive Substitution iteration with Newton's method requires developing fast solvers for the real non-symmetric systems derived in [68].

Optimization of the Time-Harmonic Solver

The drawbacks of solving time-harmonic systems using an AMG code that is unaware of the complex symmetric nature of the problem were mentioned in Section 6.8.2. These drawbacks can be resolved by developing a code that constructs the C/F splitting and the interpolation operator using a scalar AMG code and that provides a routine for the computation of the Galerkin product involving real intergrid operators and a complex fine grid operator. For these future developments, the level 2 AMG-`PETSc` interface can be used as a starting point. This interface needs to be extended with an efficient `PETSc` routine for the matrix-matrix multiplication. Such a routine is hard to develop in the general case that the individual factors are scattered over several processes. One can therefore restrict oneself to the sequential case first.

Optimization of the Field-Circuit Coupled Solver

A profiling of the generalized AMG code for field-circuit coupled problems has shown that in the numerical example of Section 7.6 more than 70 % of the CPU-time required in the set-up phase is spent in computing the coupling terms B^H on the coarser grids. This points to the need of having a more efficient matrix-matrix multiply available than the one implemented in this work. We did not investigate how using `SAMG` with aggressive coarsening affects the memory requirements and the speed of convergence of the algorithm. The generalized AMG algorithm developed in Section 7.5 does not include any smoothing on the circuit variables. Smoothing on these variables can be done for example by solving the circuit equations with a direct solver after each smoothing step on the field variables. Further research in necessary to investigate how the inclusion of this smoother affects overall convergence speed.

Transient Problems

The discretization of transient field-circuit problems leads to real symmetric matrices that have the same structure as in the time-harmonic case. For these systems, the generalized AMG algorithm can be applied with little implementation effort. One should investigate the interaction of the individual linear solves with the outer time-stepping loop. It might be possible to save CPU time by solving the first time-steps inaccurately or by freezing the AMG setup for several steps as suggested for non-linear outer iterations.

In [76] it was claimed that the transient computation of induction machines is not practically feasible due to the large amount of CPU time spent in one-level iterative linear system solvers. It would be interesting to see how the results of [76] would change by using the generalized AMG solver for real-valued field-circuit coupled systems.

Multi-Harmonic Problems

Multiharmonic formulations are typically discretized by the Harmonic Balance Finite Element Method [122, 123]. Krylov subspace methods for solving the resulting linear systems in complex arithmetic are proposed in [27]. One could investigate whether it is possible to construct multilevel preconditioners for these systems by a generalization of the approach for single frequency systems.

Multi-Physics Problems

In other extensions of this thesis, one could look into iterative solvers for the multi-physics problems in `Olympos`. Coupled thermal-magnetic and coupled structural mechanics-magnetic problems are considered in [35] and [31] respectively. For such problems one could study the reuse of AMG by incorporating it in block-preconditioning techniques or by considering system variants.

Redesign of `Olympos` using `PETSc`

In this thesis `Olympos` and `PETSc` are interfaced at the level of the linear solve only. After the construction of the mesh, the linear system is assembled and stored in `Olympos` data structures. For the linear solve using `PETSc`'s SLES object, these data structures are copied into `PETSc` data structures. This copying entails a doubling of the required memory. The linearization of non-linear problems and the time-stepping of transient problems is handled by `Olympos`. In the current interface, `PETSc` never sees a non-linear or time-dependent problem.

A redesign in which the package uses `PETSc` as building blocks is strongly recommended. In this design, `Olympos` becomes an application code in the top layer of abstraction in Figure 8.1. `PETSc` matrices and vectors are the only data structures used to store the linear system, avoiding the afore mentioned increase in memory requirements. All preconditioners, Krylov subspace methods and interfaces to external packages are immediately available to `Olympos`. From this, Schwarz type domain decomposition algorithms can be implemented on subdomains generated by `METIS` [63] for example. The Newton linearization can be performed by `PETSc`'s SNES object, providing

a framework to experiment with Newton's method for time-harmonic problems and with inexact Newton methods. Time-stepping can be performed by using the TS object.

Extension of the AMG-PETSc Interface

In further the development of the AMG-PETSc interface, it would be interesting to add the possibility to visualize the C/F splitting constructed by SAMG. This requires fetching even more information from SAMG's data structures than currently implemented. More advanced generalizations include the development of the interface for systems of coupled PDEs and for parallel computations.

Extensions to 3D

The extension of Olympos for solving three-dimensional models requires a redesign of the whole package, including the tools for pre- and post processing, the mesh generation and the assembly and solve of the linear systems. For the discretization of the double curl equations derived in Section 2.4 and Section 2.5 so-called *edge* finite element discretizations are appropriate. These finite elements were first introduced in an abstract setting in [81, 82] and were later found to be attractive in computational electromagnetism (see e.g. [12, 11] and references therein)'. In edge elements the degrees of freedom are associated to the edges or facets of the elements in the triangulation instead of to nodes. One of the virtues of edge elements is that they assure appropriate continuity conditions of the approximated vector field accros the interface between two elements. As in 3D the size of the models is larger than in 2D, the need for fast solvers in 3D is even more urgent than in 2D. Multigrid algorithms for solving discrete edge element linear systems were developed in [1, 59]. These methods treat the nullspace of the curl operator and its complement seperately. They rely on edge elements to build a discrete representation of the nullspace. An algebraic variant of these methods was developed in [92]. Domain decomposition algorithms for edge element discretizations were studied in e.g. [110].

9.3 Scientific Output of this Work

This work has led to 5 papers in international journals, 2 publications in conference proceedinds, 3 posters presented during conferences and 13 talks. For each of these catagories, this scientific output is detailed in this section.

Papers in International Journals

- R. Mertens, H. De Gersem, K. Hameyer, R. Belmans, D. Lahaye, S. Vandewalle, D. Roose, *An Algebraic Multigrid Method for Solving Very Large Electromagnetic Systems*, IEEE Trans. on Magn. Vol.34 **5** pp. 3327-3330 (1998).

- H. De Gersem, D. Lahaye, S. Vandewalle and K. Hameyer, *Comparison of Quasi Minimal Residual and Bi-Conjugate Gradient Iterative Methods for Solving the Complex Symmetric Systems of Time Harmonic Magnetic Problems* Compel Vol.18 **3** pp. 298-310 (1999).

- D. Lahaye, H. De Gersem, S. Vandewalle and K. Hameyer, *Algebraic Multigrid for Complex Symmetric Systems*, IEEE Trans. on Magn. Vol.36 **4** pp. 1535-1538 (2000).

- H. De Gersem, D. Lahaye, R. Mertens, S. Vandewalle and K. Hameyer, *Solution Strategies for Transient, Field-Circuit Coupled Systems*, IEEE Trans. on Magn. Vol.36 **4** pp. 1531-1534 (2000).

- D. Lahaye, S. Vandewalle and K. Hameyer, *An Algebraic Multilevel Preconditioner for Field-Circuit Coupled Problems*, to appear in IEEE Trans. on Magn. in March 2002.

Publications in Conference Proceedings

- H. De Gersem, D. Lahaye, S. Vandewalle and K. Hameyer, *Comparison of Quasi Minimal Residual and Bi-Conjugate Gradient Iterative Methods for Solving the Complex Symmetric Systems of Time Harmonic Magnetic Problems* Eight International IGTE Symposium on Numerical Field Calculation in Electrical Engineering, September 21-23 1999, Graz, Austria, pp. 33.

- H. De Gersem, D. Lahaye, R. Mertens, S. Vandewalle and K. Hameyer, *Solution Strategies for the Hybrid Systems Arising from Field-Circuit Coupled Electromagnetic Models*, International Conference on Preconditioning Techniques for Large Sparse Matrix Problems in Industrial Applications, June 10-12, 1999, Minneapolis, USA, pp 55-61.

Posters Presented during Conferences

- *An Algebraic Multigrid Method for Solving Very Large Electromagnetic Systems*, Eleventh Conference on the Computation of Electromagnetic Fields, November 2-6, 1997, Rio de Janeiro, Brazil.

- *Algebraic Multigrid for Complex Symmetric Systems*, Twelfth Conference on the Computation of Electromagnetic Fields, October 25-28, 1999, Sapporo, Japan.
- *An Algebraic Multilevel Preconditioner for Field-Circuit Coupled Problems*, Thirteenth Conference on the Computation of Electromagnetic Fields, July 2-5, 2001, Evian, France.

Talks

- *Coupling the Finite and Spectral Element Method in Wave Propagation Problems*, September 24, 1996, Institute of Computational Mathematics, Johannes Kepler University Linz, Austria.
- *Coupling the Finite and Spectral Element Method in Wave Propagation Problems*, September 27, 1996, Post Graduate Programme Mathematics in Industry, Eindhoven University of Technology, Eindhoven, The Netherlands.
- *On the Use of Algebraic Multigrid in an Electromagnetic Systems Simulation Package*, Ninth Copper Mountain Conference on Multigrid Methods, April 11-16, 1999, Copper Mountain, Colorado, USA.
- *On the Use of Algebraic Multigrid in an Electromagnetic Systems Simulation Package*, Scientific Seminar Series of CRS4, April 6, 1999, CRS4, Cagliari, Italy.
- *On the Use of Algebraic Multigrid in an Electromagnetic Systems Simulation Package*, Sixth European Multigrid Conference, September 27-30, 1999, Ghent, Belgium.
- *On the Use of Algebraic Multigrid in an Electromagnetic Simulation Package*, February 17, 2001, Department of Computer Science, K.U. Leuven.
- *Solving Magnetic Field - Electrical Circuit Coupled systems*, Sixth Copper Mountain Conference on Iterative Methods, April 3-7, 2000, Copper Mountain, Colorado, USA.
- *Solving Magnetic Field - Electrical Circuit Coupled systems*, Fifth International Conference on Electric and Magnetic Fields, Ghent, Belgium, May 17-19, 2000.
- *Using PETSc: A Personal View*, March 15, 2001, Graduate School on Computational Mechanics, Department of Computer Science, K.U. Leuven.
- *A Multilevel Preconditioner for Field-Circuit Coupled Problems*, Tenth Copper Mountain Conference on Multigrid Methods, April 1-6, 2001, Copper Mountain, Colorado, USA.

- *A Multilevel Preconditioner for Field-Circuit Coupled Problems*, April 10, 2001, Mathemetics and Computer Science Division, Argonne National Laboratory, Illinois, USA.
- *Algebraic Multigrid as a Solver and as a Preconditioner in Magnetic Field Computations*, June 11, 2001, Fachgebiet Theorie Elektromagnetischer Felder, Technical University Darmstadt, Germany.
- *A Multilevel Preconditioner for Field-Circuit Coupled Problems*, October 29, 2001, Center for Advanced Studies, Research and Development in Sardegna (CRS4), Cagliari, Italy.

Bibliography

[1] D.N. Arnold, R.S. Falk, and R. Winther. Multigrid in $H(div)$ and $H(curl)$. *Numer. Math.*, 85:197–218, 2000.

[2] O. Axelsson and P. Vassilevski. Algebraic multilevel preconditioning methods I. *Numer. Math.*, 56, 157–177 1989.

[3] O. Axelsson and P. Vassilevski. Algebraic multilevel preconditioning methods II. *SIAM J. Numer. Anal.*, 27(6), 1569–1590 1989.

[4] O. Axelsson and P. Vassilevski. A black box generalized conjugate gradient solver with inner iterations and variable-step preconditioning. *SIAM J. Matrix Anal. Applics.*, 4(12):625–644, 1991.

[5] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.

[6] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith. PETSc home page. http://www.mcs.anl.gov/petsc, 1998.

[7] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith. PETSc 2.0 users manual. Technical Report ANL-95/11 - Revision 2.0.29, Argonne National Laboratory, 2000.

[8] R. Bank, W. Coughran, M. Driscoll, W. Fichtner, and R. Smith. Iterative methods in semiconductor device simulations. *Comput. Phys. Comm.*, (53):201–212, 1989.

[9] R. Bank, B. Welfert, and H. Yserentant. A class of iterative methods for solving saddle point problems. *Numer. Math.*, (56):645–666, 1990.

[10] R.E. Bank and C. Wagner. Multilevel ILU decompositions. *Numer. Math.*, 82:543–576, 1999.

[11] R. Beck, P. Deuflhard, R. Hiptmair, R. W. Hoppe, and B. Wohlmuth. Adaptive multilevel methods for edge element discretizations of Maxwell's equations. *Surv. Math. Ind.*, 8:271–312, 1999.

[12] R. Beck and R. Hiptmair. Multilevel solution of the time-harmonic Maxwell's equations based on edge elements. *Int. J. Numer. Methods. Eng.*, 45:901–920, 1999.

[13] G. Bedrosian. A new method for coupling finite element field solutions with external circuits and kinematics. *IEEE Trans. Magn.*, 2(29):1664–1667, March 1993.

[14] A. Berman and R. Plemmons. *Nonnegative Matrices in the Mathematical Sciences*. Computer Science and Applied Mathematics. Academic Press, 1979.

[15] E.F.F. Botta, A. van der Ploeg, and F.W. Wubs. Nested grids ILU-decomposition (NGILU). *Journal of Computational and Applied Mathematics*, (66):515–526, 1996.

[16] D. Braess. Towards algebraic multigrid for elliptic problems of second order. *Computing*, (55):379–393, 1995.

[17] J. Bramble, J. Pasciak, and A. Vassilev. Analysis of the inexact Uzawa algorithm for saddle point problems. *SIAM J. Numer. Anal.*, 34(3):1072–1092, June 1997.

[18] A. Brandt, S. F. McCormick, and J. Ruge. Algebraic multigrid (AMG) for automatic multigrid solution with application to geodetic computations. Technical report, Institute for Computational Studies, POB 1852, Fort Collins, Colorado, 1982.

[19] M. Brezina, A.J. Cleary, R.D. Falgout, V.E. Henson, J.E. Jones, T.A. Manteuffel, S.F. McCormick, and J.W. Ruge. Algebraic multigrid based on element interpolation (AMGe). *SIAM J. Sci. Comput.*, 22(5):1570–1592, 2000.

[20] C. Brezinksi, M. R. Zaglia, and H. Sadok. New look-ahead Lanczos-type algorithms for linear systems. *Numer. Math.*, (83):53–85, 1999. to appear.

[21] M. A. Bruaset. *A Survey of Preconditioned Iterative Methods*. Pitman Research Notes in Mathematics Series. Longman Scientific & Technical, Essex, UK, 1995.

[22] G. Brussino and V. Sonnad. A comparison of direct and preconditioned iterative techniques for sparse unsymmetric systems of linear equations. *Internat. J. Numer. Methods Engrg.*, (28):801–815, 1989.

[23] T. Chan and T. Mathew. Domain decomposition algorithms. In *Acta Numerica*, pages 61–143. Cambridge University Press, 1994.

[24] E. Chow and Y. Saad. Approximate inverse techniques for block-partioned matrices. *SIAM J. Sci. Comput.*, 18(6):1657–1675, November 1997.

[25] L.O. Chua and P. M. Lin. *Computer Aided Analysis of Electronic Circuits - Algorithms and Computational Techniques.* Prentice-Hall, New Jersey, 1975.

[26] A. Cleary, R. Falgout, V. Henson, J. Jones, T. Manteuffel, S. McCormick, J. Miranda, and J. Ruge. Robustness and scalability of algebraic multigrid. *SIAM J. Sci. Comput.*, 21(5):1886–1908, 2000.

[27] H. De Gersem. *Simulation of Field-Circuit Coupled Motional Eddy Current Problems by Krylov Subspace Methods and Multilevel Techniques.* PhD thesis, KU Leuven, February 2001.

[28] H. De Gersem, R. Mertens, U. Pahner, R. Belmans, and K. Hameyer. A topological method used for field-circuit coupling. *IEEE Trans. Magn.*, 34(5):3190–3193, 1998.

[29] E. de Sturler. Truncation strategies for optimal Krylov subspace methods. *SIAM J. Numer. Anal.*, 36(3):864–889, 1999.

[30] R. De Weerdt. *Eindige elementen modellering van kooianker inductiemotoren.* PhD thesis, KU Leuven, June 1995.

[31] K. Delaere. *Computational and Experimental Analysis of Electric Machine Vibrations caused by Magnetic Forces and Magnetostriction.* PhD thesis, KU Leuven, March 2002.

[32] R.S. Dembo, S.C. Eisenstat, and T. Steihaug. Inexact Newton methods. *SIAM J. Numer. Anal.*, 19(2):400–408, 1982.

[33] J. E. Dennis Jr. and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, volume 16 of *Classics in Applied Mathematics.* SIAM, Philadelphia, PA, 1996.

[34] E. Dick, K. Riemslagh, and J. Vierendeels, editors. *Multigrid Methods VI*, volume 14 of *Lecture Notes in Computational Science and Engineering.* Springer, Berlin, 2000.

[35] J. Driesen. *Coupled Electromagnetic-Thermal Problems in Electrical Energy Transducers.* PhD thesis, KU Leuven, May 2000.

[36] P. Dular, C. Geuzaine, M.V. Ferreira, Sadowski, and J.P.A. Bastos. Connection boundary conditions with different types of finite elements applied to periodicity conditions and to the moving band. *COMPEL*, 20(1):109–119, 2001.

[37] H. Elman and G. Golub. Inexact and preconditioned Uzawa algorithms for saddle point problems. *SIAM J. Numer. Anal.*, 31(6):1645–1661, December 1994.

[38] R. Fletcher. Conjugate gradient methods for indefinite systems. In G. A. Watson, editor, *Proceedings of the Dundee Biennal Conference of Numerical Analysis*, pages 73–89, New York, 1975. Springer-Verlag.

[39] R. W. Freund. Conjugate gradient-type methods for linear systems with complex symmetric coefficient matrices. *SIAM J. Sci. Statist. Comp.*, 13(1):425–448, 1992.

[40] R. W. Freund, G. H. Golub, and N. M. Nachtigal. Iterative solution of linear systems. In *Acta Numerica*, volume 1, pages 229–269. Cambridge University Press, 1997.

[41] R. W. Freund, M. H. Gutknecht, and N. M. Nachtigal. An implementation of the look-ahead Lanczos algorithm for non-hermitian matrices. *SIAM J. Sci. Comput.*, 14:137–158, 1993.

[42] R. W. Freund and N. M. Nachtigal. An implementation of the look-ahead Lanczos algorithm, part ii. Technical Report RIACS Technical Report 90.46, Research Institute for Advanced Computer Science, NASA Ames Research Center, Moffett Field, California, December 1990.

[43] R. W. Freund and N. M. Nachtigal. An implementation of the QMR method based on coupled two-terms recurrences. *SIAM J. Sci. Comput.*, 15(2):313–337, 1994.

[44] P. J. Frey and P. L. George. *Mesh Generation.* Hermes Science Publishing, Oxford, 2000.

[45] G. H. Golub and H. A. Van der Vorst. Closer to the solution: iterative linear solvers. In I.S. Duff and G.A. Watson, editors, *The State of the Art in Numerical Analysis*, pages 63–92. Claredon Press, 1997.

[46] G. H. Golub and C. F. Van Loan. *Matrix Computations.* The John Hopkins University Press, Baltimore, Maryland, second edition, 1992.

[47] S. Goossens. *Krylov Convergence Acceleration and Domain Decomposition Methods for Nonmatching Grids.* PhD thesis, KU Leuven, June 2000.

[48] T. Grauschopf, M. Griebel, and H. Regler. Additive multilevel-preconditioners based on bilinear interpolation, matrix-dependent geometric coarsening and algebraic-multigrid coarsening for second order elliptic PDEs. *Applied Numerical Mathematics*, 23(1):63–97, 1997.

[49] A. Greenbaum. *Iterative Methods for Solving Linear Systems*, volume 17 of *Frontiers in Applied Mathematics*. SIAM, Philadelphia, PA, 1997.

[50] M. Griebel, T. Neunhoeffer, and H. Regler. Algebraic multigrid methods for the solution of the Navier-Stokes equations in complicated geometries. *Int. J. Numer. Methods for Heat and Fluid Flow*, 26:281–301, 1998.

[51] William Gropp, Ewing Lusk, and Anthony Sklellum. *Using MPI: Portable Parallel Programming with the Message Passing Interface*. MIT Press, Cambridge, MA, second edition, 1999.

[52] M. H. Gutknecht and Z. Strakoš. Accuracy of three-term and two-term recurrences for Krylov space solvers. Technical report, ETH Zürich, 1997.

[53] W. Hackbush. *Multi-Grid Methods and Applications*, volume 4 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 1985.

[54] W. Hackbush and G. Wittum, editors. *Multigrid Methods V*, volume 3 of *Lecture Notes in Computational Science and Engineering*. Springer, Berlin, 1998.

[55] K. Hameyer and R. Belmans. Design of very small electromagnetic and electrostatic micro motors. *IEEE Transactions on Energy Conversion*, 14(4):1241–1246, December 1999.

[56] K. Hameyer, R. Belmans, R. De Weerdt, and E. Tuinman. Finite element analysis of steady state behavior of squirrel cage induction motors compared with measurements, part II. *IEEE Trans. Magn.*, 33(2):2093–2096, March 1997.

[57] P.W. Hember and P. Wesseling, editors. *Multigrid Methods IV*, volume 116 of *International Series of Numerical Mathematics*. Birkhäuser Verlag, Basel, Switzerland, 1994.

[58] M.R. Hestenes and E.L. Stiefel. Methods of conjugate gradient for solving linear systems. *J. Res. Nat. Bur. Stand.*, 49:406–436, 1952.

[59] R. Hiptmair. Multigrid method for Maxwell's equations. *SIAM J. Numer. Anal.*, 36(1):204–225, 1998.

[60] M. Holst and S. Vandewalle. Schwarz methods: to symmetrize or not to symmetrize. *SIAM J. Numer. Anal.*, 34(2):699–722, April 1997.

[61] R. A. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge University Press, Cambridge, UK, 1985.

[62] J. Janssen and S. Vandewalle. Multigrid waveform relaxation on spatial finite element meshes: the discrete-time case. *SIAM J. Sci. Comput.*, 17(1):133–155, Jan 1996.

[63] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20(1):359–392, 1998.

[64] C. T. Kelley. *Iteratieve Methods for Linear and Nonlinear Equations*, volume 16 of *Frontiers in Applied Mathematics.* SIAM, Philadelphia, PA, 1995.

[65] F. Kickinger. Algebraic multigrid for discrete elliptic second-order problems. Technical Report 513, Departement of Mathematics, Johannes Kepler University Linz, Altenbergerstrasse 69, A-4040 Linz, Austria, March 1995.

[66] A. Klawonn. An optimal preconditioner for a class of saddle point problems with a penalty term. *SIAM J. Sci. Comput.*, 19(2):540–552, March 1998.

[67] A. Krechel, K. Stüben, and F. Brakhagen. Algebraic multigrid: Improved operator dependent interpolation. Technical Report 1042, German National Research Center for Information Technology (GMD), Schloss Birlinhoven, D-53754 Sankt-Augustin, Germany, January 1997.

[68] D. Lederer, H. Igarashi, and A. Kost. The Newton-Raphson method for complex symmetric systems. *ACES Journal*, 12:113–116, July 1997.

[69] J. L. Lions and E. Magenes. *Nonhomogeneous Boundary Value Problems and Applications.* Springer-Verlag, Berlin, 1972.

[70] P. Lombard and G. Meunier. A general purpose method for electric and magnetic combined problems for 2d, axisymmetric and transient systems. *IEEE Trans. Magn.*, 29(2):1737–1740, March 1993.

[71] B. Maerten, D. Roose, A. Basermann, J. Clinckemaillie, T. Coupez, J. Fingberg, H. Digonnet, R. Ducloux, J.-M. Gratien, U. Hartmann, G. Lonsdale, and C. Walshaw. Dynamic load-balancing of finite element applications with the drama library. *Applied Mathematical Modelling*, 25:83–98, 2000.

[72] J. C. Maxwell. *A Treatise on Electricity and Magnetism.* Reprint in Clarendon Press, 1891.

[73] S. McCormick, editor. *Multigrid Methods*, volume 3 of *Frontiers in Applied Mathematics.* SIAM, Philadelphia, PA, 1987.

[74] S. McCormick, editor. *Multilevel Adaptive Methods for Partial Differential Equations*, volume 6 of *Frontiers in Applied Mathematics.* SIAM, Philadelphia, PA, 1989.

[75] S. McCormick, editor. *Multilevel Projection Methods for Partial Differential Equations*, volume 62 of *CBMS-NSF Regional Conference Series in Applied Mathematics.* SIAM, Philadelphia, PA, 1992.

[76] R. Mertens. *Dynamische eindige elementen simulatie van een inductiemotor in een aandrijfsysteem.* PhD thesis, KU Leuven, September 1999.

[77] R. Mertens, S. Henneberger, K. Hameyer, and R. Belmans. Analysis of magnetic rollers with the finite element method. *IEEE Trans. Magn.*, 32(3):722–724, May 1996.

[78] R. Mertens, U. Pahner, R. Belmans, and K. Hameyer. Selected projection methods to improve the convergence of non-linear problems. *COMPEL*, 18(4):638–646, 1999.

[79] R. Mertens, U. Pahner, H. De Gersem, R. Belmans, and K. Hameyer. Improving the overall solver speed: A fast, reliable and simple adaptive mesh refinement scheme. In *International Workshop on Electric and Magnetic Fields*, pages 385–390, 1998. May 12-15, 1998, Marseille, France.

[80] N. Nachtigal, S.C. Reddy, and L.N. Trefethen. How fast are nonsymmetric matrix iterations? *SIAM J. Matrix Anal. Applics.*, 13(3):778–795, 1992.

[81] J.C. Nedelec. Mixed finite elements in R^3. *Numer. Math.*, 35:315–341, 1980.

[82] J.C. Nedelec. A new family of finite elements in R^3. *Numer. Math.*, 50:57–81, 1986.

[83] Y. Notay. A multilevel block incomplete factorization preconditioning. *Appl. Numer. Math*, 31:209–225, 1999.

[84] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Problems in Several Variables.* Academic Press, New-York, 1970.

[85] P. Oswald. *Multilevel Finite Element Approximation: Theory and Applications.* Teubner Skripte zur Numerik. B. G. Teubner, Stuttgart, 1994.

[86] U. Pahner. *A General Design Tool for the Numerical Optimization of Electromagnetic Energy Transducers.* PhD thesis, KU Leuven, May 1998.

[87] C.C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12(4):617–629, 1975.

[88] A. Plaks, I. Tsukerman, S. Painchaud, and L. Tabarovsky. Multigrid methods for open boundary problems in geophysics. *IEEE Trans. Magn.*, 36(4):633–638, 2000.

[89] M. Putti and C. Cordes. Finite element approximations of the diffusion operator on tetrahedra. *SIAM J. Sci. Comput.*, 19(4):1154–1168, 1998.

[90] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations.* Springer-Verlag, Berlin, 1994.

[91] Q. Quarteroni and A. Valli. *Domain Decomposition Methods for Partial Differential Equations.* Oxford Science Publications, 1999.

[92] S. Reitzinger. *Algebraic Multigrid Methods for Large Scale Finite Element Equations.* PhD thesis, Johannes-Kepler-Universität Linz, 2001.

[93] A. Reusken. On the approximate cyclic reduction preconditioner. *SIAM J. Sci. Comput.*, 21(2), 1999.

[94] U. Rüde. *Mathematical and Computational Techniques for Multilevel Adaptive Methods*, volume 13 of *Frontiers in Applied Mathematics.* SIAM, Philadelphia, PA, 1993.

[95] J. Ruge and K. Stüben. Algebraic multigrid. In S. McCormick, editor, *Multigrid Methods*, volume 3 of *Frontiers in Applied Mathematics*, pages 73–130, SIAM, Philadelphia, PA, 1987.

[96] T. Rusten and R. Winther. A preconditioned iterative method for saddlepoint problems. *SIAM J. Matrix Anal. Applics.*, 13(3):887–904, July 1992.

[97] Y. Saad. Krylov subspace methods for solving large unsymmetric linear systems. *Mathematics of Computation*, 37:105–126, 1981.

[98] Y. Saad. A flexible inner-outer preconditioned GMRES algorithm. *SIAM J. Sci. Comput.*, 14(2):461–469, 1993.

[99] Y. Saad. ILUM: a multi-elimination ILU preconditioner for general sparse matrices. *SIAM J. Sci. Comput.*, 17(4):830–847, 1996.

[100] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS Publishing Company, Boston, MA, 1996.

[101] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comp.*, 7(3):856–869, 1986.

[102] D. Silvester and D. Wathen. Fast iterative solution of stabilized Stokes systems part II: Using general block preconditioners. *SIAM J. Numer. Anal.*, 31(5):1352–1367, October 1994.

[103] G. Sleijpen and H. Van der Vorst. Reliable updated residuals in hybrid Bi-CG methods. *Computing*, 56:141–163, 1996.

[104] B. F. Smith, P. O. Bjørstad, and W. D. Gropp. *Domain Decompostion: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press, 1996.

[105] A. Sommerfeld. *Electrodynamics.* Academic Press, New York, 1952.

[106] P. Sonneveld. CGS, a fast Lanczos-type solver for nonsymmetric systems. *SIAM J. Sci. Statist. Comp.*, 10(1):36–52, 1989.

[107] G. Strang and G. J. Fix. *An Analysis of the Finite Element Method.* Prentice-Hall, 1973.

[108] K. Stüben. Algebraic multigrid: An introduction for positive definite problems with applications. Technical Report 53, German National Research Center for Information Technology (GMD), Schloss Birlinhoven, D-53754 Sankt-Augustin, Germany, March 1999.

[109] K. Stüben. An introduction to algebraic multigrid. In [111], pages 413–532. Academic Press, 2001.

[110] A. Toselli, B.I. Wohlmuth, and O.B. Widlund. An iterative substructuring method for Maxwell's equations in two dimensions. *Mathematics of Computation*, 70(235):935–949, 2001.

[111] U. Trottenberg, C. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, San Diego, 2001.

[112] H. Van der Vorst. The convergence behaviour of preconditioned CG and CG-S in the presence of rounding errors. *Lect. Notes in Math.*, (1457):126–136, 1990.

[113] H. A. Van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Statist. Comp.*, 13(2):631–644, 1992.

[114] H. A. Van der Vorst and J. B. M. Melissen. A Petrov-Galerkin method for solving $Ax = b$, where A is complex symmetric. *IEEE Trans. Magn.*, 26(2):706–708, 1990.

[115] H. A. Van der Vorst and C. Vuik. GMRESR: a family of nested GMRES methods. *Numerical Linear Algebra with Applications*, 1(4):369–386, 1994.

[116] P. Vaněk, J. Mandel, and M. Brezina. Algebraic multigrid on unstructured meshes. Technical Report 34, Center for Computational Mathematics, University of Colorado at Denver, UCD CCM, December 1994.

[117] P. Vaněk, J. Mandel, and M. Brezina. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*, (56):179–196, 1996.

[118] R. S. Varga. *Matrix Iterative Analysis*. Prentice-Hall, Boston, MA, 1962.

[119] R. Verfürth. A combined conjugate gradient-multigrid algorithm for the numerical solution of the Stokes problem. *IMA Journal of Numerical Analysis*, (4):441–445, 1984.

[120] W.L. Wan, T.F Chan, and B. Smith. An energy minimizing interpolation for robust multigrid methods. Technical Report 98-6, Departement of Mathematics, University of California, Los Angeles, UCLA CAM, February 1998.

[121] A. Wathen and D. Silvester. Fast iterative solution of stabilized stokes systems part I: Using simple diagonal preconditioners. *SIAM J. Numer. Anal.*, 30(3):630–649, June 1993.

[122] S. Yamada, K. Bessho, and J. Lu. Harmonic balance finite element method applied to nonlinear AC magnetic analysis. *IEEE Trans. Magn.*, 25(4):2971–2973, July 1989.

[123] S. Yamada, P.P. Biringer, and K. Bessho. Calculation of nonlinear eddy-current problems by the harmonic balance finite element method. *IEEE Trans. Magn.*, 27(5):4122–4125, September 1991.

[124] D.M. Young. *Iterative Solution of Large Linear Systems.* Academic Press, New York, 1971.

[125] L. Zaslavsky. An adaptive algebraic multigrid for reactor critical calculations. *SIAM J. Sci. Comput.*, 16:840–847, 1995.

Index