Immanuel Anjam

Funktionaaliset a posteriori virhe-estimaatit Maxwellin yhtälöille

Tietotekniikan
(simulointi ja optimointi)
pro gradu -tutkielma
20. syyskuuta 2010

JYVÄSKYLÄN YLIOPISTO
TIETOTEKNIIKAN LAITOS

Jyväskylä

Tekijä: Immanuel Anjam
Yhteystiedot: immanuel.anjam@jyu.fi
Työn nimi: Funktionaaliset a posteriori virhe-estimaatit Maxwellin yhtälöille
Title: Functional type a posteriori error estimates for Maxwell's equations
Työ: Tietotekniikan (simulointi ja optimointi) pro gradu -tutkielma
Ohjaajat: Olli Mali, Pekka Neittaanmäki, Sergey Repin
Sivumäärä: 79

Tiivistelmä: Funktionaaliset a posteriori virhe-estimaatit ovat osoittautuneet luotettavaksi tavaksi arvioida osittaisdifferentiaaliyhtälöiden numeeristen ratkaisujen virhettä. Tässä tutkielmassa malliongelma on Maxwellin yhtälöistä johdettu toisen kertaluvun reuna-arvotehtävä. Tälle yhtälölle on johdettu jo aikaisemmin funktionaaliset a posteriori virhe-estimaatit, mutta niiden suorituskykyä ei ole vielä tutkittu kattavasti. Tutkielman alkuosa keskittyy malliongelman numeeriseen ratkaisemiseen: elementtimenetelmään, jossa käytetään Nédélecin elementtiä. Tutkielman jälkimmäisessä osassa johdetaan funktionaalinen ala- ja yläraja. Näiden estimaattien todetaan analyyttisesti olevan *tarkkoja*. Tämä ominaisuus vahvistetaan myös numeerisilla testeillä. Numeeriset testit osoittavat myös, että yläraja on *herkkä* malliongelman funktion κ suhteen. Ylärajasta johdetaan myös kaksi uutta virheindikaattoria, joiden todetaan toimivan hyvin adaptiivisessa kontekstissa.

Abstract: Functional a posteriori error estimates have proven to be a reliable way to approximate the error in numerical solutions of partial differential equations. In this thesis we use the second order Maxwell system as our model problem. Functional a posteriori error estimates have already been derived for this model problem, but they have not yet been extensively tested. The first part of the thesis focuses on solving the model problem numerically. We use the finite element method with the Nédélec's element. In the latter part of the thesis we derive functional lower and upper bounds. These bounds are proven to be sharp. Also numerical tests verify this. Numerical tests also show that the upper bound is sensitive with respect to the function κ. We also derive two new error indicators from the upper bound, and discover that they work well in adaptive context.

Avainsanat: funktionaalinen, a posteriori, virhe-estimaatti, indikaattori, alaraja, minorantti, yläraja, majorantti, Nédélec, FEM, ODY, Maxwell
Keywords: functional, a posteriori, error estimatimate, indicator, lower bound, minorant, upper bound, majorant, Nédélec, FEM, PDE, Maxwell

Sanasto

a posteriori	jälkeenpäin
a priori	ennalta, edeltäkäsin
alaraja	suure, joka on aina pienempi tai yhtäsuuri kuin tarkka virhe
FEM	elementtimenetelmä (engl. *finite element method*)
GA	keskimääräisen gradientin menetelmä (engl. *gradient averaging*)
herkkä	suure on herkkä, mikäli jonkin parametrin pienet muutokset aiheuttavat suureen arvon suuria muutoksia
majorantti	katso *yläraja*
minorantti	katso *alaraja*
ODY	osittaisdifferentiaaliyhtälö (engl. *PDE, partial differential equation*)
silottaminen	keskiarvoistaminen
tarkka	tarkka virhe-estimaatti antaa teoriassa mielivaltaisen tarkkoja arvioita numeerisen approksimaation virheelle, kun verkon elementtien määrä kasvaa äärettömään
virheindikaattori	virheen jakaumaa arvioiva funktio, joka yleensä ei ole alaraja eikä yläraja
yläraja	suure, joka on aina suurempi tai yhtäsuuri kuin tarkka virhe

Matemattiset merkinnät

$\lvert \cdot \rvert$	pinta-ala tai itseisarvo
$\lvert \cdot \rvert_{\mathrm{curl}}$	$H(\mathrm{curl})$-seminormi
$\lVert \cdot \rVert$	L^2-normi
$\lVert \cdot \rVert_{\mathrm{curl}}$	$H(\mathrm{curl})$-normi
$\lVert \cdot \rVert_{L^2}$	L^2-normi
$\lvert\!\lvert\!\lvert \cdot \rvert\!\rvert\!\rvert$	energianormi
∇	gradienttioperaattori
δ_{ij}	Kroneckerin delta
∂_i	osittaisderivaatta komponentin i suhteen
κ	sähkökentän permittiivisyys
μ	magneettinen permeabiliteetti
φ	testifunktio (H^1-FEM)
$\boldsymbol{\psi}_i$	testifunktioavaruuden V_h i:s virittävä kantafunktio
Ω	alue avaruudessa $\mathbb{R}^D$
$\overline{\Omega}$	alueen Ω sulkeuma
$\partial\Omega$	alueen Ω reuna
$a(\cdot,\cdot)$	bilineaarinen ja symmetrinen muoto variaatioyhtälössä
$a_M(\cdot,\cdot)$	muodon $a(\cdot,\cdot)$ osa, joka muodostaa massamatriisin $\mathbf{M}$
$a_S(\cdot,\cdot)$	muodon $a(\cdot,\cdot)$ osa, joka muodostaa jäykkyysmatriisin $\mathbf{S}$
$\tilde{a}(\cdot,\cdot)$	bilineaarinen ja symmetrinen muoto variaatioyhtälössä
$\hat{A}$	referenssielementillä $\hat{K}$ määritellyt vapausasteet
$\mathbf{b}$	voimavektori
$\mathbf{b}_K$	funktion $\boldsymbol{F}_K$ vektoriosa
$\mathbf{B}_K$	funktion $\boldsymbol{F}_K$ matriisiosa
$\mathbf{c}$	tuntemattomien kertoimien vektori
curl	skalaariarvoinen curl-operaattori vektoriarvoisille funktioille
$\underline{\mathrm{curl}}$	vektoriarvoinen curl-operaattori skalaariarvoisille funktioille
D	alueen Ω dimensio
div	divergenssioperaattori
det	determinanttioperaattori
$\hat{e}_i$	referenssielementin $\hat{K}$ i:s reuna
$\boldsymbol{f}$	voimafunktio
$\boldsymbol{F}_K$	affiinikuvaus referenssielementiltä $\hat{K}$ elementille K

h	elementtiverkon karakteristinen koko
I_d	majorantista $M_\oplus$ johdettu virheindikaattori
I_{eff}	tehokkuusindeksi (engl. *efficiency index*)
I_r	majorantista $M_\oplus$ johdettu virheindikaattori
J	energiafunktionaali
K	elementti alueen Ω elementtiverkosta
$\mathbf{K}$	jäykkyysmatriisin $\mathbf{S}$ ja massamatriisin $\mathbf{M}$ summa
$\hat{K}$	referenssielementti
$l(\cdot)$	lineaarinen muoto variaatioyhtälössä
$\tilde{l}(\cdot)$	lineaarinen muoto variaatioyhtälössä
$\mathbf{M}$	massamatriisi (engl. *mass matrix*)
$M_\ominus$	minorantti, alaraja (engl. *minorant, lower bound*)
$M_\oplus$	majorantti, yläraja (engl. *majorant, upper bound*)
$\boldsymbol{n}$	alueen Ω reunan ulkonormaali
$\hat{\boldsymbol{N}}_i$	referenssielementtiin $\hat{K}$ liittyvä i:s kantafunktio
P_i	alueen Ω elementtiverkon i:s solmupiste tai elementin K i:s solmupiste
$\hat{P}_i$	referenssielementin $\hat{K}$ i:s solmupiste
$\mathbb{P}_k$	k-asteiset polynomit
$\tilde{\mathbb{P}}_k$	k-asteiset homogeeniset polynomit
$\mathcal{P}_K$	elementtiin K liittyvä Piola-muunnos
$\mathbf{R}$	rotaatiomatriisi
$\hat{R}$	referenssielementillä $\hat{K}$ määritelty funktioavaruus
$\mathbb{R}^D$	D-ulotteinen euklidinen reaaliavaruus
$\mathbf{S}$	jäykkyysmatriisi (engl. *stiffness matrix*)
$\boldsymbol{t}$	alueen Ω reunan vastapäivään suunnistettu tangentti
$\hat{\boldsymbol{t}}_i$	referenssielementin $\hat{K}$ reunan $\hat{e}_i$ vastapäivään suunnistettu tangentti
$\boldsymbol{u}$	malliongelman tarkka ratkaisu
$\boldsymbol{v}$	malliongelman numeerinen ratkaisu
V	testifunktioavaruus, variaatioavaruus
V_h	testifunktioavaruuden V äärellisulotteinen approksimaatio
$\boldsymbol{w}$	testifunktio (Nédélec-FEM)
$\boldsymbol{x}$	vapaa vektori alueessa Ω
$\hat{\boldsymbol{x}}$	vapaa vektori referenssielementissä $\hat{K}$
y	majorantin vapaa parametri

Vektoreita merkitään kursiivilla lihavoidulla fontilla ja vektorin komponentteja merkitään samoin kuin vektoriakin, mutta ilman lihavointia. Alaindeksin numero ilmaisee monesko komponentti on kyseessä. Esimerkiksi avaruuden $\mathbb{R}^2$ sijaintivektoria ilmaistaan

$$\boldsymbol{x} = (x_1, x_2)^T = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} .$$

Kaksiulotteiselle vektoriarvoiselle funktiolle $\boldsymbol{w}$ ja skalaariarvoiselle funktiolle φ tässä tutkielmassa käytetyt osittaisdifferentiaalioperaattorit ovat

$$\begin{aligned} \operatorname{curl} \boldsymbol{w} &:= \partial_1 w_2 - \partial_2 w_1 , \\ \underline{\operatorname{curl}}\, \varphi &:= \begin{pmatrix} \partial_2 \varphi \\ -\partial_1 \varphi \end{pmatrix} , \\ \operatorname{div} \boldsymbol{w} &:= \partial_1 w_1 + \partial_2 w_2 , \\ \nabla \varphi &:= \begin{pmatrix} \partial_1 \varphi \\ \partial_2 \varphi \end{pmatrix} . \end{aligned}$$

Ns. hatulla $\hat{\cdot}$ merkitseminen tarkoittaa sitä, että kyseessä on referenssielementtiin liittyvä suure, funktio tai operaattori. Esimerkiksi referenssielementtiä itseään merkitään $\hat{K}$ ja alueen Ω verkon elementtiä merkitään K.

Esipuhe

Olen työskennellyt Prof. Sergey Repinin tutkimusprojektissa sen verran kauan että elementtimenetelmä (lyh. *FEM*) on tullut tutuksi ja ymmärrykseni virheen estimoinnista on kasvanut huomattavasti. Aluksi tutkin FEM-menetelmän perusperiaatteita ja toteutin CSC:n FEM-ohjelmisto Elmerillä [13] yksinkertaisen Poissonin yhtälön ratkaisijan. Yksinkertaisuudestaan huolimatta kaiken tämän oppiminen oli yllättävän haastava prosessi.

Tämän jälkeen sain tehtäväkseni tutkia miten Nédélecin elementit toimivat. Tämä oli erittäin haastava ja aikaavievä tehtävä, ja tein varmasti kaikki virheet mitä kykenin tekemään. Lopulta sain kuitenkin toteutettua FEM-ratkaisijan Nédélecin elementillä. Seuraavaksi tutustuin funktionaalisiin virhe-estimaatteihin. Ohjaajani Olli Malin suuren panostuksen ansiosta saimme lyhyessä ajassa toteutettua funktionaaliset virhe-estimaatit. Tässä vaiheessa totesimme että materiaalia on enemmän kuin tarpeeksi pro gradu -tutkielmaa varten.

Tutkielman kirjoitus kuitenkin keskeytyi tutkimuksen saatua lisää tuulta siipiensä alle. Hyvien tutkimustuloksien myötä kirjoitimme kollegoidemme kanssa aiheesta artikkelin [3]. Artikkelissa hyödynsimme myös tässä tutkielmassa tutkittavien virhe-estimaattien numeerisia tuloksia. Tämän tutkielman valmistuessa on artikkeli jo julkaistu.

Haluaisin kiittää Repiniä mahdollisuudesta tehdä pro gradu -tutkielmani tästä mielenkiintoisesta aiheesta. Kiitän ohjaajiani Prof. Pekka Neittaanmäkeä ja Olli Malia mainiosta ohjauksesta ja aidosta mielenkiinnosta tutkielmani aihetta kohtaan.

Lopuksi haluan kiittää perhettäni ja ystäviäni, joita ilman elämä olisi aivan liian tylsää ja merkityksetöntä.

Sisältö

1 Johdanto

Moderni elektromagnetismin teoria sai perustansa vuonna 1873, kun Maxwell[1] julkaisi artikkelinsa *Treatise on Electricity and Magnetism*. Artikkelissaan hän esitti joukon yhtälöitä, jotka on nimetty hänen mukaansa Maxwellin yhtälöiksi. Ajan saatossa Maxwellin käsite eetteristä on vaihtunut kenttien käsitteeseen ja yhtälöiden määrä on supistunut neljään.

Kuten lukuisat muut luontoa kuvaavat yhtälöt, myös Maxwellin yhtälöt ovat osittaisdifferentiaaliyhtälöitä (lyh. *ODY*). Huolimatta osittaisdifferentiaaliyhtälöiden teorian suurista harppauksista 1900-luvulla, ei näitä yhtälöitä kyetä ratkaisemaan tarkasti lähes milloinkaan. ODY:ja ratkaistaankin tämän takia numeerisesti eri menetelmillä. Elementtimenetelmä (lyh. *FEM*) on suosittu menetelmä ODY:jen ratkaisemiseen. Menetelmä kehitettiin 1900-luvun puolivälissä, ja sillä on vankka matemaattinen perusta. Menetelmä on laskennallisesti tehokas: laskentakapasiteettia voidaan ohjata sinne, missä ratkaisulle halutaan tarkempi esitys. Elementtimenetelmään perustuvia kaupallisia ja vapaan lähdekoodin ohjelmistoja on nykyään saatavana useita.

Viime aikoina numeeristen ratkaisujen luotettavuus on saanut paljon huomiota. On tärkeää tietää, kuinka hyviä ratkaisuja käytetty menetelmä tuottaa. Ilman minkäänlaista virhearviota ovat simulaatiot hataralla pohjalla. Jos numeerisen ratkaisun virhettä kyetään arvioimaan luotettavasti, on mahdollista paikallistaa ongelmakohdat, muuttaa ratkaistavaa ongelmaa ja tehdä simuloinnit uudestaan.

Tässä tutkielmassa tutkimme Maxwellin yhtälöistä johdettua toisen kertaluvun yhtälöä. Yhtälö esitetään toisessa luvussa. Ratkaisemme tämän yhtälön FEM-menetelmällä käyttäen Nédélecin elementtiä, jonka esittelemme kolmannessa luvussa. Tämän tutkielman pääaiheena on virheen estimointi, ja neljännessä luvussa erittelemme erilaisia virhelähteitä. Viidennessä luvussa pääsemme tutkielman varsinaiseen antiin eli virhe-estimaatteihin. Tässä tutkielmassa käytetylle yhtälölle on jo aikaisemmin johdettu funktionaaliset a posteriori virhe-estimaatit, mutta niitä ei ole vielä tutkittu kattavasti. Johdamme virheen ylärajasta myös kaksi uutta virheindikaattoria. Kuudennessa luvussa havainnoimme virhe-estimaattien ja -indikaattoreiden toimintaa.

Lukijalta oletetaan elementtimenetelmän ja osittaisdifferentiaaliyhtälöiden teorian perusteiden tuntemusta. Kummastakin aiheesta on runsaasti englanninkielistä materiaalia. Solinin kirja [39] tarjoaa ytimekkään, joskin teknisen lähestymisen osittaisdifferentiaaliyhtälöiden teoriaan. Elementtimenetelmään tutustuvalle voin suositella Hughesin kirjaa [19] sekä Krizekin ja Neittaanmäen kirjaa [22]. Suomen kielellä elementtimenetelmään voi tutustua CSC:n oppaalla [16], jossa on kattava osio elementtimenetelmästä.

Tämän tutkielman ymmärtäminen ei varsinaisesti vaadi elektromagnetismin tuntemista,

[1] James Clerk Maxwell, 1831–1879

vaikka käyttämämme yhtälö on johdettu Maxwellin yhtälöistä. Elektromagnetismin tunteminen on kuitenkin välttämätöntä yhtälön fysikaalisten ominaisuuksien ymmärtämiseksi. Pollacin ja Stumpin kirja [34] on kattava esitys elektromagnetismin teorialle. Elektromagnetismia tuntevalle voin suositella Monkin kirjaa [28], joka käsittelee pelkästään Maxwellin yhtälöitä ja niiden ratkaisemista elementtimenetelmällä.

2 Malliongelma

Malliongelmamme on toisen kertaluvun reuna-arvotehtävä, jolla on homogeeninen Dirichlet'n reunaehto. Alueen $\Omega \subset \mathbb{R}^2$ oletetaan olevan Lipschitz-alue. Tehtävänä on löytää vektoriarvoinen funktio $\boldsymbol{u}(\boldsymbol{x})$, $\boldsymbol{x} = (x_1, x_2)^T \in \overline{\Omega}$, joka toteuttaa reuna-arvotehtävän

$$\underline{\mathrm{curl}}(\mu^{-1} \operatorname{curl} \boldsymbol{u}) + \kappa\, \boldsymbol{u} = \boldsymbol{f} \qquad \text{kun } \boldsymbol{x} \in \Omega\,, \tag{2.1}$$

$$\boldsymbol{u} \times \boldsymbol{n} = 0 \qquad \text{kun } \boldsymbol{x} \in \partial\Omega\,, \tag{2.2}$$

missä magneettinen permeabiliteetti $\mu(\boldsymbol{x}) > 0$ ja sähkökentän permittiivisyys $\kappa(\boldsymbol{x}) > 0$ ovat positiivisia vakioita tai positiivisia rajoitettuja funktioita. Voimafunktio $\boldsymbol{f}(\boldsymbol{x})$ kuuluu avaruuteen $L^2(\Omega, \mathbb{R}^2)$ ja vektorilla $\boldsymbol{n}$ merkitään alueen Ω ulkonormaalia.

Osittaisdifferentiaaliyhtälö (2.1) voidaan johtaa sekä aikariippuvaisista, että aikaharmonisista Maxwellin yhtälöistä. Funktioiden κ ja $\boldsymbol{f}$ fyysinen merkitys vaihtelee sen mukaan, miten yhtälö on johdettu. Yhdessä reunaehdon (2.2) kanssa yhtälö mallintaa elektromagneettisen kentän jakautumista onkalossa, jota ympäröi täydellisesti johtava seinä. Tämä reuna-arvotehtävä on elliptinen. Tarkempi kuvaus malliongelmassa esiintyvistä funktioista löytyy liitteestä A, jossa malliongelma johdetaan.

Toisin kuin liitteessä A, yhtälössä (2.1) on kaksi eri curl-operaattoria. Koska malliongelmassamme alue Ω kuuluu kaksiulotteiseen avaruuteen, tarvitsemme skalaariarvoisen operaattorin vektoriarvoiselle funktiolle $\boldsymbol{w}$ ja vektoriarvoisen operaattorin skalaariarvoiselle funktiolle φ:

$$\operatorname{curl} \boldsymbol{w} := \partial_1 w_2 - \partial_2 w_1\,, \qquad \underline{\mathrm{curl}}\, \varphi := \begin{pmatrix} \partial_2 \varphi \\ -\partial_1 \varphi \end{pmatrix}.$$

Kolmiulotteisessa avaruudessa tarvitaan ainoastaan yksi curl-operaattori. Tässä tutkielmassa keskitymme kuitenkin kaksiulotteiseen tapaukseen, sillä malliyhtälön ratkaiseminen sekä virhe-estimaattien toteutus ovat kolmiulotteisessa tapauksessa huomattavasti vaikeampia tehtäviä.

Seuraavat kaksi huomautusta ovat hyödyllisiä havaintoja siitä miten kaksiulotteisessa avaruudessa voidaan käyttää vaihtoehtoisia tapoja curl-operaattorien ja reunaehdon (2.2) ilmaisemiseen.

Huomautus 2.1. *Käyttämällä rotaatiomatriisia*

$$\mathbf{R} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

voimme esittää curl*-operaattorin divergenssin avulla:*

$$\operatorname{curl} \boldsymbol{w} = \operatorname{div} \mathbf{R}\boldsymbol{w} \,.$$

Operaattorin $\underline{\text{curl}}$ *voimme esittää gradienttikentän rotaationa:*

$$\underline{\operatorname{curl}}\, \varphi = \mathbf{R}\nabla\varphi \,.$$

Alueen Ω *reunan vastapäivään suunnistettu tangentiaalivektori voidaan myös ilmaista rotaatiomatriisin avulla, sillä* $\boldsymbol{t} = \mathbf{R}^T\boldsymbol{n}$*.*

Huomautus 2.2. *Koska avaruudessa* $\mathbb{R}^2$ *pätee* $\boldsymbol{u} \times \boldsymbol{n} = \boldsymbol{u} \cdot \mathbf{R}^T\boldsymbol{n}$*, huomautuksen 2.1 mukaan reunaehto (2.2) on ekvivalentti yhtälön* $\boldsymbol{u} \cdot \boldsymbol{t} = 0$ *kanssa, missä* $\boldsymbol{t}$ *on reunan* $\partial\Omega$ *vastapäivään suunnistettu tangentti.*

Tämän luvun alussa esitetyssä ongelmassa tehtävänä on etsiä sellainen funktio $\boldsymbol{u}$, joka toteuttaa yhtälöt (2.1)-(2.2). On selvää, että $\boldsymbol{u}$ toteuttaa nämä yhtälöt vain jos se on kahdesti differentioituva, eli kuuluu avaruuteen $C^2(\Omega, \mathbb{R}^2)$. Kahdesti differentioituvuus on hyvin tiukka sileysvaatimus, jota ei yleensä kyetä täyttämään. Onkin tarpeellista kyetä heikentämään näitä sileysvaatimuksia. Tämä onnistuu määrittelemällä heikko differentioituvuus ja Sobolev-avaruudet. Näiden avulla malliyhtälölle voidaan määritellä ekvivalentti muoto, jonka ratkaisuilta vaaditaan differentioituvuusominaisuuksia vain heikossa mielessä.

2.1 Heikko differentioituvuus ja Sobolev-avaruudet

Tässä luvussa kertaamme L^2 avaruuden määritelmän ja määrittelemme heikon differentioituvuuden avulla Sobolev-avaruudet. Oletamme, että lukija tuntee mitta- ja integraaliteorian sekä funktionaalianalyysin alkeet. Emme siis määrittele mitallisuuden käsitteitä saati paneudu tarkemmin avaruuksien määrittelyihin. Suomen kielellä mitta- ja integraaliteoriaa löytyy Kilpeläisen luentomonisteesta [21]. Funktionaalianalyysistä on kirjoitettu lukuisia hyviä kirjoja, niistä voin suositella Kahanpään suomenkielistä luentomonistetta [20] sekä Friedmanin kirjaa [14]. Tässä luvussa alueen Ω oletetaan olevan Lipschitz-alue avaruudessa $\mathbb{R}^D$, missä $1 < D < \infty$.

Yhtälön (2.1) voimafunktio kuuluu tehtävän määrittelyn mukaan L^2-avaruuteen.

Määritelmä 2.1 (L^2-avaruus)**.** *Avaruus $L^2(\Omega)$ pitää sisällään kaikki ne funktiot, joiden neliöt ovat Lebesquen mielessä integroituvia alueessa Ω:*

$$L^2(\Omega) := \{f : \Omega \to \mathbb{R} : \|f\| < \infty\}.$$

Kuten usein on tapana, merkitsemme L^2-normia $\|\cdot\|_{L^2}$ lyhyesti $\|\cdot\|$:

$$\|f\| := \left(\int_\Omega |f|^2 \, d\boldsymbol{x}\right)^{(1/2)}.$$

Sisätulolla

$$(f, g) = \int_\Omega f g \, d\boldsymbol{x}$$

varustettuna $L^2(\Omega)$ on Hilbert-avaruus[1]*.*

Tämä L^2-avaruuden määrittely on skalaariarvoiselle funktiolle. Vastaavan avaruuden määrittely vektoriarvoiselle funktiolle on hyvin suoraviivaista. Jos funktion $\boldsymbol{f} = (f_1, f_2, \ldots, f_{\tilde{D}})$ jokainen komponenttifunktio $f_i \in L^2(\Omega)$, sanotaan että $\boldsymbol{f}$ kuuluu avaruuteen L^2. Tätä merkitään $\boldsymbol{f} \in L^2(\Omega, \mathbb{R}^{\tilde{D}})$.

Malliongelmamme ratkaisun osittaisderivaatat eivät välttämättä ole olemassa klassisen differentioituvuuden mielessä. Tämän vuoksi otamme käyttöön heikon derivaatan käsitteen, jonka avulla voimme määritellä Sobolev-avaruudet[2]. Skalaariarvoisen funktion heikko derivaatta määritellään seuraavasti.

Määritelmä 2.2 (Heikko derivaatta)**.** *Funktiolla $f \in L^2(\Omega)$ on kertaluvun α heikko derivaatta $g = \partial^\alpha f$, kun $g \in L^2(\Omega)$ ja*

$$(\varphi, g) = (-1)^{|\alpha|}(\partial^\alpha \varphi, f)$$

kaikilla $\varphi \in C^\infty(\Omega)$, jotka joukossa Ω poikkeavat nollasta vain kompaktissa joukossa.[3]

Skalaariarvoista funktiota sanotaan heikosti differentioituvaksi, jos sillä on heikko derivaatta. Huomaa, että jos funktiolla f on olemassa derivaatta klassisessa mielessä, se on identtinen heikon derivaatan kanssa. Heikko derivaatta onkin klassisen derivaatan yleistys. Lisäksi klassiset derivointisäännöt pätevät myös heikolle derivaatalle. Vektoriarvoinen funktio $\boldsymbol{f}$ on heikosti differentioituva, jos sen jokainen komponenttifunktio f_i on heikosti differentioituva.

[1] Hilbert-avaruudet ovat nimetty David Hilbertin (1862–1943) mukaan. Hilbert-avaruus on sisätuloavaruus, joka on Banach-avaruus sisätulon määräämällä normilla. Banach-avaruus on vektoriavaruus, jossa jokainen Cauchy-jono suppenee.

[2] Sobolev-avaruudet ovat nimetty Sergei Lvovich Sobolevin (1908–1989) mukaan.

[3] Tässä α on multi-indeksi: $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_D)$ ja $|\alpha| = \alpha_1 + \alpha_2 + \cdots + \alpha_D$ (missä D on sen avaruuden dimensio, missä joukko Ω sijaitsee).

Määritelmä 2.3 (Sobolev-avaruus H^m)**.** *Olkoon $m > 0$ kokonaisluku. Sobolev-avaruus $H^m(\Omega)$ on funktioiden $f \in L^2(\Omega)$ joukko, joilla on olemassa heikot derivaatat kaikilla $|\alpha| \leq m$. Sisätulolla*

$$(f, g)_m = \sum_{|\alpha| \leq m} \int_\Omega \partial^\alpha f \, \partial^\alpha g \, d\boldsymbol{x}$$

ja normilla

$$\|f\|_m = \sqrt{\sum_{|\alpha| \leq m} \|\partial^\alpha f\|^2}$$

Sobolev-avaruus on Hilbert-avaruus.

Edelleen, jos vektoriarvoisen funktion $\boldsymbol{f}$ jokainen komponenttifunktio $f_i \in H^m(\Omega)$, sanotaan että $\boldsymbol{f}$ kuuluu H^m-avaruuteen. Tätä merkitään $\boldsymbol{f} \in H^m(\Omega, \mathbb{R}^{\tilde{D}})$.

Tästä lähtien heikkoa differentioituvuutta ei tulla painottamaan erikseen, vaan kaikki differentiaalit käsitetään heikossa mielessä.

2.2 Malliongelman variaatiomuoto

Nyt voimme määritellä malliyhtälölle ekvivalentin muodon, jonka ratkaisuilta vaaditaan differentioituvuusominaisuuksia vain heikossa mielessä. Tätä muotoa kutsustaan *variaatiomuodoksi*. Kirjallisuudessa reuna-arvotehtävän variaatiomuotoa kutsutaan myös *yleistetyksi muodoksi* ja *heikoksi muodoksi*.

Ennen variaatiomuodon määrittelyä esitetään muutamia avaruuksia ja tuloksia joita tarvitsemme myöhemmin.

Määritelmä 2.4. *Olkoon $\Omega \subset \mathbb{R}^2$. Avaruus*

$$H(\mathrm{curl}, \Omega) := \{\boldsymbol{w} \in L^2(\Omega, \mathbb{R}^2) : \mathrm{curl}\, \boldsymbol{w} \in L^2(\Omega)\}.$$

on Sobolev-avaruus. Tämän avaruuden seminormi

$$|\boldsymbol{w}|_{\mathrm{curl}} := \|\mathrm{curl}\, \boldsymbol{w}\|$$

voidaan täydentää normiksi

$$\|\boldsymbol{w}\|^2_{\mathrm{curl}} := |\boldsymbol{w}|^2_{\mathrm{curl}} + \|\boldsymbol{w}\|^2 = \|\mathrm{curl}\, \boldsymbol{w}\|^2 + \|\boldsymbol{w}\|^2.$$

Lause 2.1 (Gaussin divergenssilause)**.** *Olkoon* $\Omega \subset \mathbb{R}^2$ *ja* $\boldsymbol{w} \in H(\mathrm{curl}, \Omega)$*. Silloin*

$$\int_\Omega \mathrm{div}\, \boldsymbol{w}\, d\boldsymbol{x} = \int_{\partial\Omega} \boldsymbol{w} \cdot \boldsymbol{n}\, ds\,,$$

missä $\boldsymbol{n}$ *on alueen* Ω *ulkonormaali.*

Todistus. Katso [37, sivu 288, lause 10.51] □

Lause 2.2 (Greenin lause)**.** *Olkoon* $\Omega \subset \mathbb{R}^2$ *ja* $\boldsymbol{w} \in H(\mathrm{curl}, \Omega)$ *ja* $\varphi \in H^1(\Omega)$*. Silloin*

$$\int_\Omega \mathrm{curl}\, \boldsymbol{w}\, \varphi\, d\boldsymbol{x} = \int_\Omega \boldsymbol{w} \cdot \underline{\mathrm{curl}}\, \varphi\, d\boldsymbol{x} + \int_{\partial\Omega} (\boldsymbol{w} \cdot \boldsymbol{t}) \varphi\, ds\,,$$

missä $\boldsymbol{t}$ *on alueen* Ω *reunan vastapäivään suunnistettu tangentti.*

Todistus. Olkoon $\boldsymbol{w} \in H(\mathrm{curl}, \Omega)$ ja $\varphi \in H^1(\Omega)$. Huomautuksen 2.1 mukaan

$$\mathrm{curl}\, \boldsymbol{w}\, \varphi = \mathrm{div}\, \mathbf{R}\boldsymbol{w}\, \varphi\,.$$

Osittaisderivointisääntöjen[4] avulla voimme kirjoittaa

$$\mathrm{curl}\, \boldsymbol{w}\, \varphi = -\mathbf{R}\boldsymbol{w} \cdot \nabla\varphi + \mathrm{div}(\mathbf{R}\boldsymbol{w}\, \varphi)\,.$$

Integroimalla tämä yhtälö puolittain alueen Ω yli saadaan

$$\int_\Omega \mathrm{curl}\, \boldsymbol{w}\, \varphi\, d\boldsymbol{x} = -\int_\Omega \mathbf{R}\boldsymbol{w} \cdot \nabla\varphi\, d\boldsymbol{x} + \int_\Omega \mathrm{div}(\mathbf{R}\boldsymbol{w}\, \varphi)\, d\boldsymbol{x}\,.$$

Soveltamalla oikean puolen jälkimmäiseen termiin Gaussin divergenssilausetta 2.1 saamme

$$\int_\Omega \mathrm{curl}\, \boldsymbol{w}\, \varphi\, d\boldsymbol{x} = -\int_\Omega \mathbf{R}\boldsymbol{w} \cdot \nabla\varphi\, d\boldsymbol{x} + \int_{\partial\Omega} (\mathbf{R}\boldsymbol{w} \cdot \boldsymbol{n}) \varphi\, ds\,.$$

Käyttämällä hyväksi sisätulon ominaisuuksia voimme kirjoittaa

$$\int_\Omega \mathrm{curl}\, \boldsymbol{w}\, \varphi\, d\boldsymbol{x} = -\int_\Omega \boldsymbol{w} \cdot \mathbf{R}^T \nabla\varphi\, d\boldsymbol{x} + \int_{\partial\Omega} (\boldsymbol{w} \cdot \mathbf{R}^T \boldsymbol{n}) \varphi\, ds\,.$$

Koska $\mathbf{R}^T = -\mathbf{R}$ ja huomautuksen 2.1 mukaan $\underline{\mathrm{curl}}\, \varphi = \mathbf{R}\nabla\varphi$ ja $\boldsymbol{t} = \mathbf{R}^T\boldsymbol{n}$, saamme Greenin lauseen. □

[4] Osittaisderivointisääntö: $\mathrm{div}(\boldsymbol{w}\, \varphi) = \mathrm{div}\, \boldsymbol{w}\, \varphi + \boldsymbol{w} \cdot \nabla\varphi$, missä $\boldsymbol{w} \in H(\mathrm{curl}, \Omega)$ ja $\varphi \in H^1(\Omega)$.

Variaatiomuotoa varten tarvitsemme sopivan testifunktioavaruuden. Malliongelmallemme sopiva testifunktioavaruus on avaruuden $H(\mathrm{curl}, \Omega)$ funktiot, jotka toteuttavat reunaehdon (2.2):

$$V := H_0(\mathrm{curl}, \Omega) := \{\boldsymbol{w} \in H(\mathrm{curl}, \Omega) : \boldsymbol{w} \times \boldsymbol{n} = 0\}\,.$$

Huomautus 2.3. *Olkoon $\boldsymbol{w} \in V$ ja $\varphi \in H^1(\Omega)$. Tällöin*

$$\int_\Omega \mathrm{curl}\,\boldsymbol{w}\,\varphi\,d\boldsymbol{x} = \int_\Omega \boldsymbol{w} \cdot \underline{\mathrm{curl}}\,\varphi\,d\boldsymbol{x}\,.$$

Tämä seuraa suoraan Greenin lauseesta 2.2 ja siitä, että huomautuksen 2.2 mukaan funktion $\boldsymbol{w}$ sisätulo tangentiaalikomponentin $\boldsymbol{t}$ kanssa on nolla.

Nyt olemme valmiit johtamaan variaatiomuodon. Ensin yhtälö (2.1) kerrotaan testifunktiolla $\boldsymbol{w} \in V$, jonka jälkeen integroidaan puolittain alueen Ω yli. Yhtälön ratkaisun $\boldsymbol{u} \in V$ tulee tällöin toteuttaa

$$\int_\Omega \left(\underline{\mathrm{curl}}(\mu^{-1}\,\mathrm{curl}\,\boldsymbol{u}) \cdot \boldsymbol{w} + \kappa\,\boldsymbol{u} \cdot \boldsymbol{w}\right) d\boldsymbol{x} = \int_\Omega \boldsymbol{f} \cdot \boldsymbol{w}\,d\boldsymbol{x} \tag{2.3}$$

kaikilla testifunktioilla $\boldsymbol{w} \in V$. Käyttämällä huomautusta 2.3 yhtälöön (2.3) saamme malliyhtälön variaatiomuodon: etsi funktio $\boldsymbol{u} \in V$, joka toteuttaa yhtälön

$$\int_\Omega \left(\mu^{-1}\,\mathrm{curl}\,\boldsymbol{u}\,\mathrm{curl}\,\boldsymbol{w} + \kappa\,\boldsymbol{u} \cdot \boldsymbol{w}\right) d\boldsymbol{x} = \int_\Omega \boldsymbol{f} \cdot \boldsymbol{w}\,d\boldsymbol{x}\,, \quad \forall \boldsymbol{w} \in V\,. \tag{2.4}$$

Tarkempi kuvaus variaatiomuodon johtamisesta elliptisille ongelmille löytyy Solinin kirjasta [39, sivu 14].

Luettavuuden helpottamiseksi otamme käyttöön tiiviimmän merkintätavan. Merkitsemme yhtälön vasenta puolta symmetrisellä bilineaarimuodolla $a : V \times V \to \mathbb{R}$ ja yhtälön oikeaa puolta lineaarisella funktionaalilla $l : V \to \mathbb{R}$. Yhtälö (2.4) saadaan siten muotoon

$$a(\boldsymbol{u}, \boldsymbol{w}) = l(\boldsymbol{w})\,, \quad \forall \boldsymbol{w} \in V\,. \tag{2.5}$$

Variaatio-ongelma (2.5) voidaan esittää myös minimointiongelmana, sillä se on ekvivalentti energiafunktionaalin J minimoinnin kanssa:

$$\begin{aligned} J(\boldsymbol{w}) &:= \frac{1}{2}a(\boldsymbol{w}, \boldsymbol{w}) - l(\boldsymbol{w}) \\ &= \int_\Omega \left(\frac{\mu^{-1}}{2}|\mathrm{curl}\,\boldsymbol{w}|^2 + \frac{\kappa}{2}\,|\boldsymbol{w}|^2 - \boldsymbol{f} \cdot \boldsymbol{w}\right) d\boldsymbol{x}\,, \end{aligned} \tag{2.6}$$

missä minimoitava muuttuja on $\boldsymbol{w} \in V$.

Variaatio-ongelman (2.5) ratkaisemiseen tarvitaan menetelmä, joka tuottaa approksimaatioita $H(\mathrm{curl}, \Omega)$-avaruudesta. Kuuluuhan variaatio-ongelman tarkka ratkaisu tähän avaruuteen.

2.3 Galerkin-menetelmä

Galerkin-menetelmän ideana on approksimoida variaatiomuodon avaruutta V äärellisulotteisella aliavaruudella $V_h \subset V$, jonka virittävät äärellinen määrä kantafunktioita:

$$V_h = \langle \boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_m \rangle,$$

missä $m \in \mathbb{N}$. Malliongelman numeerinen ratkaisu $\boldsymbol{v} \in V_h$ voidaan siis esittää muodossa

$$\boldsymbol{v} = \sum_{i=1}^{m} c_i \boldsymbol{\psi}_i \,, \tag{2.7}$$

missä vakiot c_i ovat tuntemattomia. On selvää, että ratkaisun $\boldsymbol{v}$ laatu riippuu hyvin paljon siitä kuinka hyvin kantafunktiot $\boldsymbol{\psi}_i$ kykenevät esittämään tarkan ratkaisun u.

Variaatiomuoto (2.5) voidaan nyt kirjoittaa muotoon

$$a(\boldsymbol{v}, \boldsymbol{\psi}_j) = l(\boldsymbol{\psi}_j) \,, \quad j = 1, \ldots, m \,. \tag{2.8}$$

Sijoittamalla (2.7) yhtälöön (2.8) saamme

$$a\left(\sum_{i=1}^{m} c_i \boldsymbol{\psi}_i, \boldsymbol{\psi}_j \right) = l(\boldsymbol{\psi}_j) \,, \quad j = 1, \ldots, m \,.$$

Koska $a(\cdot, \cdot)$ on bilineaarinen, voidaan summa ja vakiot c_i ottaa sen ulkopuolelle:

$$\sum_{i=1}^{m} c_i a(\boldsymbol{\psi}_i, \boldsymbol{\psi}_j) = l(\boldsymbol{\psi}_j) \,, \quad j = 1, \ldots, m \,.$$

Tehtävä on näin saatu muutettua lineaariseksi yhtälöryhmäksi, jossa on m yhtälöä ja m tuntematonta. Yhtälöryhmä voidaan esittää matriisin $\mathbf{K} = \{k_{ij}\}_{i,j=1}^{m}$ ja vektorin $\mathbf{b} = \{b_j\}_{j=1}^{m}$ avulla kun määrittelemme niiden alkiot $k_{ij} = a(\boldsymbol{\psi}_i, \boldsymbol{\psi}_j)$ ja $b_j = l(\boldsymbol{\psi}_j)$:

$$\mathbf{K}\mathbf{c} = \mathbf{b} \,, \tag{2.9}$$

missä $\mathbf{c} = \{c_i\}_{i=1}^{m}$ on tuntemattomien kertoimien vektori. Matriisi K on selvästi symmetrinen. On helppo osoittaa, että matriisi $\mathbf{K}$ on myös positiivisesti definiitti, eli yhtälöryhmällä on yksikäsitteinen ratkaisu. Symmetrisille ja positiivisesti definiiteille matriiseille on kehitetty lukuisia tehokkaita ratkaisumenetelmiä (katso esim. [15]).

Bilineaarimuoto $a(\cdot,\cdot)$ koostuu kahdesta osasta: $a(\boldsymbol{u},\boldsymbol{w}) = a_S(\boldsymbol{u},\boldsymbol{w}) + a_M(\boldsymbol{u},\boldsymbol{w})$, missä

$$a_S(\boldsymbol{u},\boldsymbol{w}) = \int_\Omega \mu^{-1} \operatorname{curl}\boldsymbol{u} \operatorname{curl}\boldsymbol{w}\, d\boldsymbol{x}$$

ja

$$a_M(\boldsymbol{u},\boldsymbol{w}) = \int_\Omega \kappa\, \boldsymbol{u}\cdot\boldsymbol{w}\, d\boldsymbol{x}\,.$$

Molemmat muodot ovat muodon $a(\cdot,\cdot)$ tapaan symmetrisiä ja bilineaarisia. Matriisi $\mathbf{K}$ voidaan nyt ilmoittaa *jäykkyysmatriisin* $\mathbf{S} = \{s_{ij}\}_{i,j=1}^m$ ja *massamatriisin* $\mathbf{M} = \{m_{ij}\}_{i,j=1}^m$ summana, missä $s_{ij} = a_S(\boldsymbol{\psi}_i,\boldsymbol{\psi}_j)$ ja $m_{ij} = a_M(\boldsymbol{\psi}_i,\boldsymbol{\psi}_j)$:

$$(\mathbf{S}+\mathbf{M})\mathbf{c} = \mathbf{b}\,. \tag{2.10}$$

Kuten matriisi $\mathbf{K}$, myös $\mathbf{S}$ ja $\mathbf{M}$ ovat symmetrisiä ja positiivisesti definiittejä.

Lineaarisen yhtälöryhmän oikean puolen vektoria $\mathbf{b}$ kutsutaan yleensä *voimavektoriksi*. Matriisien $\mathbf{S}$ ja $\mathbf{M}$ sekä vektorin $\mathbf{b}$ nimet voivat muuttua sovelluskohteen mukaan. Matriisille $\mathbf{S}$ käytetään usein myös nimitystä *johtavuusmatriisi*. Yleensä kuitenkin käytetään nimiä jäykkyys, massa ja voima. Nämä nimitykset juontavat juurensa lujuuslaskentaan, joka oli Galerkin-menetelmän ensimmäisiä sovelluskohteita.

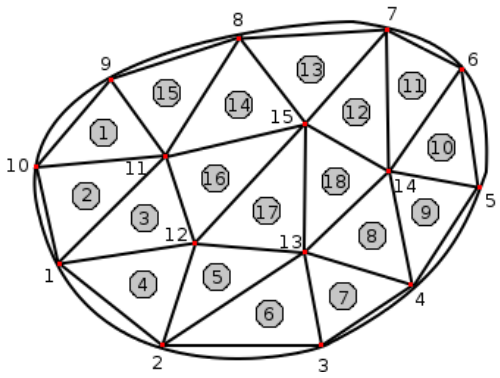
3 Nédélecin elementtimenetelmä

Elementtimenetelmä [16] on eräs Galerkin-approksimaatio, jossa kantafunktiot on valittu laskentataakkaa silmälläpitäen. Lineaarisen yhtälöryhmän (2.9) muodostamisessa suurin osa työstä tehdään matriisin $\mathbf{K}$ alkioiden $k_{ij} = a(\boldsymbol{\psi}_i, \boldsymbol{\psi}_j)$ laskemisessa. Elementtimenetelmässä kantafunktiot $\boldsymbol{\psi}_i$ valitaan siten, että suurin osa niiden tuloista ja niiden derivaattojen tuloista ovat nollia. Näitä alkioita ei tarvitse siten laskea lainkaan. Tämä nopeuttaa sekä matriisin $\mathbf{K}$ muodostamista, että yhtälöryhmän (2.9) ratkaisemista.

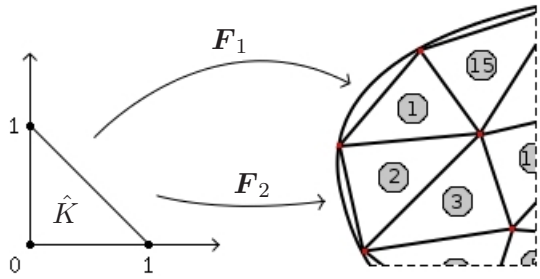
Elementtimenetelmässä tarkasteltava alue Ω jaetaan pieniin osiin, joita kutsutaan elementeiksi. Tätä elementtien kokoelmaa kutsutaan elementtiverkoksi. Kuvassa 3.1 on erään kaksiulotteisen alueen elementtiverkko. Kuvan verkko on jaettu kolmioihin, mutta elementit voivat olla mitä tahansa monikulmioita. Elementtiverkko voi myös sisältää erilaisia monikulmioita. Täten monimutkaisetkin alueet ovat mahdollisia.

Elementtien joukkoa nimitetään affiiniksi elementtiperheeksi, jos sen jokainen elementti voidaan muodostaa affiinikuvauksella ns. *referenssielementiltä*. Esimerkiksi kolmiot saadaan aikaiseksi affiinilla kuvauksella referenssikolmiosta, kuten kuvan 3.2 tilanteessa. Alueen Ω elementtiverkon kolmioelementtiin K liittyvä affiinikuvaus saadaan helposti selville koska tiedämme elementin K solmupisteiden koordinaatit. Koska affiinikuvaus voidaan muodostaa jokaiselle elementille, voidaan matriisin $\mathbf{K}$ alkioiden integrointi suorittaa muuttujanvaihdon jälkeen referenssielementillä. Tästä on laskennallista hyötyä, sillä numeerinen integrointikvadratuuri tarvitsee laskea vain kerran referenssielementillä. Muuten kvadratuuri pitäisi laskea erikseen jokaiselle elementtiverkon elementille.

Galerkin-approksimaation laatu riippuu paljolti siitä, miten ja millaiset kantafunktiot valitaan. Elementtimenetelmässä avaruuden V_h virittävät kantafunktiot $\boldsymbol{\psi}_i$ ovat yleensä palapolynomeja, mutta ne voivat olla myös esimerkiksi splinejä. Kantafunktiot valitaan siten, että



Kuva 3.1: Elementtiverkko. Elementtien numerot ovat ympyröity, ja kolmioiden kärkipisteet ovat solmuja.



Kuva 3.2: Elementteihin 1 ja 2 liittyvät affiinikuvaukset $\boldsymbol{F}_1$ ja $\boldsymbol{F}_2$.

ne liittyvät järjestelmällisesti vain tiettyihin elementteihin. Lineaarisessa H^1-FEM:issä jokainen kantafunktio liittyy vain yhteen solmuun. Jokainen kantafunktio eroaa nollasta vain tässä tietyssä solmussa ja tähän solmuun liittyvissä elementeissä. Lineaarisessa Nédélec-FEM:issä jokainen kantafunktio liittyy vain yhteen reunaan. Jokainen kantafunktio eroaa nollasta siten vain korkeintaan kahdessa elementissä. Kun kantafunktiot valitaan näin, tulee matriisista $\mathbf{K}$ harva, koska suurin osa sen alkioista $k_{ij} = a(\boldsymbol{\psi}_i, \boldsymbol{\psi}_j)$ ovat nollia. Matriisista muodostuu nauhamatriisi, mikä on hyvä ominaisuus: matriisi vie vähän tilaa sekä keskusmuistissa että kiintolevylle tallennettuna. Kuten aiemmin mainitsimme, matriisi $\mathbf{K}$ on myös symmetrinen ja positiivisesti definiitti. Näitä ominaisuuksia voidaan käyttää hyväksi ratkaistaessa lineaarista yhtälöryhmää (2.9) tehokkaasti.

Mitä enemmän elementtejä verkossa on, sitä suuremmaksi matriisi $\mathbf{K}$ kasvaa. Tästä tietenkin seuraa myös se, että matriisin kokoaminen ja lineaarisen yhtälöryhmän ratkaiseminen on laskennallisesti raskaampaa. Elementtimenetelmässä voidaan kuitenkin käyttää erikokoisia elementtejä. Matriisin kokoa voidaan pienentää käyttämällä verkkoa, joka on tiheä vain tärkeissä alueissa, ja harva vähemmän tärkeissä alueissa. Tämä on yksi elementtimenetelmän parhaimmista ominaisuuksista: laskentatarkkuutta voidaan parantaa halutuilla alueilla.

Ranskalainen matemaatikko J.C. Nédélec esitti $H(\mathrm{curl}, \Omega)$-avaruudessa konformin elementtimenetelmän vuonna 1980 [30]. Hänen toinen julkaisunsa [29] vuodelta 1986 laajentaa edelleen tätä teoriaa. Tämän tutkielman puitteissa kiinnostuksemme kohteena on Nédélecin elementin implementointi. Tämä luku perustuu sen takia suurimmalta osin Schneebelin raporttiin [38], missä esitetään implementoinnin vaatimia teknisiä ratkaisuja.

Tässä luvussa määrittelemme lineaarisen Nédélecin kolmioelementin kaksiulotteisessa avaruudessa. Elementti, jonka tulemme esittämään, on ns. ensimmäisen tyypin Nédélecin elementti, jonka Nédélec esitti julkaisussa [30]. Jatkossa viittaamme ensimmäisen tyypin Nédélecin elementtiin vain Nédélecin elementtinä. Käsittelemme vain alimman kertaluvun elementin, jossa elementin kantafunktiot ovat lineaarisia. Nédélecin elementtimenetelmän tietokonetoteutuksen testiajot esitetään luvun lopussa.

3.1 Elementin konstruktio

Ciarlet'n [12, luku 2] mukaan äärellisen elementin määrittelee kolmikko $(\hat{K}, \hat{R}, \hat{A})$, missä $\hat{K}$ on geometria, $\hat{R}$ on funktioavaruus ja $\hat{A}$ on vapausasteiden joukko:

Geometria: Määrittelemme referenssielementin $\hat{K}$ ja muuttujanvaihdon $\boldsymbol{F}_K(\hat{\boldsymbol{x}})$, jota kutsutaan elementtikuvaukseksi, sillä se kuvaa referenssielementin alueen Ω verkon lokaaliksi elementiksi K: $K = \boldsymbol{F}_K(\hat{K})$.

Funktioavaruus: Tarvitsemme äärellisulotteisen funktioavaruuden $\hat{R}$, joka on määritelty referenssielementillä. Lisäksi tarvitsemme muunnoksen, joka kuvaa tämän funktioavaruuden funktiot referenssielementiltä $\hat{K}$ elementtiverkon lokaalille elementille K.

Vapausasteet: Määrittelemme vapausasteet $\hat{A} = \{\hat{\alpha}_i(\cdot) : \hat{R} \to \mathbb{R}\}_{i=1}^{m}, m \in \mathbb{N}$, jotka ovat lineaarisia funktionaaleja kantafunktioista. Näiden vapausasteiden tulee olla lineaarisesti riippumattomia toisistaan.

Elementtiä suunniteltaessa täytyy funktioavaruus ja vapausasteet valita siten, että niiden muodostamat globaalit kantafunktiot kuuluvat haluttuun avaruuteen. Erityisesti elementtien lokaalien kantafunktioiden virittämien globaalien kantafunktioiden on toteutettava avaruuden jatkuvuusehdot. Tällöin menetelmää sanotaan konformiksi. Esimerkiksi avaruudessa H^1 jatkuvuusehtona on se, että elementtien avulla laskettu globaali FEM-funktio on jatkuva. Avaruudessa $H(\mathrm{curl})$ jatkuvuusehto on tangentiaalikomponentin jatkuvuus elementtien reunojen yli:

Lause 3.1. *Olkoon $K_1 \in \mathbb{R}^2$ ja $K_2 \in \mathbb{R}^2$ kaksi vierekkäistä ja erillistä Lipschitz-aluetta, joilla on yhteinen reuna e siten, että $\overline{K}_1 \cap \overline{K}_2 = e$. Merkitään näiden kahden alueen unionia $\Omega = K_1 \cup K_2$. Olkoon $\boldsymbol{n}_1$ alueen K_1 ulkonormaali ja $\boldsymbol{n}_2$ alueen K_2 ulkonormaali. Funktio $\boldsymbol{w}$ kuuluu avaruuteen $H(\mathrm{curl}, \Omega)$ jos ja vain jos*

1) sen rajoittuma $\boldsymbol{w}_1 = \boldsymbol{w}|_{K_1}$ elementtiin K_1 kuuluu avaruuteen $H(\mathrm{curl}, K_1)$,

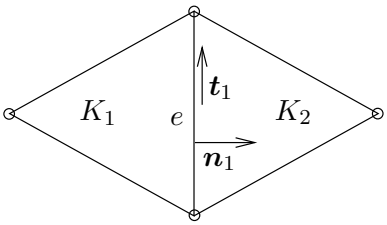
2) sen rajoittuma $\boldsymbol{w}_2 = \boldsymbol{w}|_{K_2}$ elementtiin K_2 kuuluu avaruuteen $H(\mathrm{curl}, K_2)$,

3) ja rajoittumien $\boldsymbol{w}_1$ ja $\boldsymbol{w}_2$ tangentiaalikomponentit reunalla e kumoavat toisensa: $\boldsymbol{w}_1 \times \boldsymbol{n}_1 + \boldsymbol{w}_2 \times \boldsymbol{n}_2 = 0$.

2D:ssä kolmas ehto on huomautuksen 2.2 mukaan yhtäpitävä ehdon $\boldsymbol{w}_1 \cdot \boldsymbol{t}_1 + \boldsymbol{w}_2 \cdot \boldsymbol{t}_2 = 0$ kanssa, missä $\boldsymbol{t}_1$ ja $\boldsymbol{t}_2$ ovat vastaavasti alueiden K_1 ja K_2 vastapäivään suunnistetut tangentit. Tilanne on visualisoitu kuvassa 3.3.

Todistus. Katso [28, sivu 107]. □

Seuraavaksi määrittelemme geometrian, funktioavaruuden ja vapausasteet siten, että ne toteuttavat tangentiaalikomponenttien jatkuvuusehdon.



Kuva 3.3: Kaksi vierekkäistä elementtiä joilla on yhteinen reuna.

3.2 Geometria

Tässä tutkielmassa valitsemme referenssielementiksi $\hat{K}$ kaksiulotteisen yksikkökolmion

$$\hat{K} = \{\, (\hat{x}_1, \hat{x}_2)^T \in \mathbb{R}^2 : \quad 0 \le \hat{x}_1 \le 1 \quad , \quad 0 \le \hat{x}_2 \le 1 - \hat{x}_1 \,\},$$

jolla on solmut

$$\hat{P}_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \qquad \hat{P}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \qquad \hat{P}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

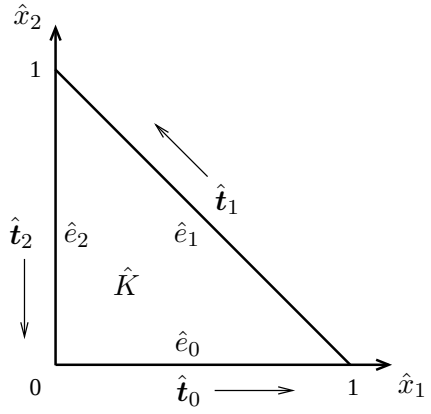
reunat

$$\hat{e}_0 = \overline{\hat{P}_0\,\hat{P}_1}, \qquad \hat{e}_1 = \overline{\hat{P}_1\,\hat{P}_2}, \qquad \hat{e}_2 = \overline{\hat{P}_2\,\hat{P}_0}$$

ja yksikkötangentit

$$\hat{\boldsymbol{t}}_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \qquad \hat{\boldsymbol{t}}_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \qquad \hat{\boldsymbol{t}}_2 = \begin{pmatrix} 0 \\ -1 \end{pmatrix}.$$

Kuten kuvasta 3.4 näkee, reunat ja tangentit ovat numeroitu vastapäivään aloittaen alimmasta reunasta. Samoin tangenttien suunta kulkee vastapäivään. Reunojen ja tangenttien suunnistus voitaisiin valita toisinkin, mutta on tärkeää muistaa minkälaista suunnistusta käytetään. Suunnistuksen on käytävä ilmi ohjelmakoodista. Jos esimerkiksi haluamme taulukon,



Kuva 3.4: Referenssielementti $\hat{K}$.

jossa jokainen elementtiverkon elementti ilmoitetaan solmupisteidensä avulla, on järkevää että jokaisen elementin solmut on ilmoitettu suunnistuksen mukaisesti. Suunnistuksen tärkeys käy ilmi tulevissa luvuissa.

Kuten tämän luvun alussa todettiin, tarvitsemme referenssielementin lisäksi kuvauksen, joka kuvaa referenssielementin alueen Ω elementiksi. Olkoon K verkon elementti, jonka solmupisteet ovat

$$P_0 = \begin{pmatrix} a_0 \\ b_0 \end{pmatrix}, \qquad P_1 = \begin{pmatrix} a_1 \\ b_1 \end{pmatrix}, \qquad P_2 = \begin{pmatrix} a_2 \\ b_2 \end{pmatrix}.$$

Tavoitteenamme on muodostaa affiinikuvaus $\boldsymbol{F}_K : \hat{K} \to K$, jolle pätee $\boldsymbol{F}_K(\hat{P}_i) = P_i$ kaikilla $i = 0, 1, 2$. Tulokseksi saamme

$$\boldsymbol{F}_K(\hat{\boldsymbol{x}}) = \mathbf{B}_K \hat{\boldsymbol{x}} + \mathbf{b}_K = \begin{pmatrix} a_1 - a_0 & a_2 - a_0 \\ b_1 - b_0 & b_2 - b_0 \end{pmatrix} \begin{pmatrix} \hat{x}_1 \\ \hat{x}_2 \end{pmatrix} + \begin{pmatrix} a_0 \\ b_0 \end{pmatrix}. \tag{3.1}$$

Tätä kuvausta kutsutaan elementtikuvaukseksi.

3.3 Funktioavaruus ja vapausasteet

Seuraavaksi määrittelemme funktioavaruuden ja vapausasteet, johon elementtimme perustuu. Näiden tarkka konstruktio löytyy artikkelista [30].

Yleisen tavan mukaan avaruus $\mathbb{P}_k$ merkitsee k-asteisia polynomeja. Avaruudella $\tilde{\mathbb{P}}_k$ merkitsemme k-asteisia homogeenisia polynomeja referenssielementillä $\hat{K}$.

Määritelmä 3.1. *2D:ssä ensimmäisen tyypin Nédélecin elementti perustuu funktioavaruuteen*

$$\mathcal{R}^k = \mathbb{P}_{k-1}(\hat{K}, \mathbb{R}^2) \oplus \tilde{\mathbb{P}}_{k-1} \begin{pmatrix} \hat{x}_2 \\ -\hat{x}_1 \end{pmatrix},$$

missä $k \geq 1$ on elementin kertaluku.

Vapausasteet tulee valita siten, että jatkuvuusehto 3.1 täyttyy: ratkaisun tangentiaalikomponenttien tulee olla jatkuvia elementtien reunoilla. Vapausasteiden tulee myös olla lineaarisesti riippumattomia toisistaan.

Määritelmä 3.2. *Olkoon $\hat{K}$ aiemmin tässä luvussa määritelty yksikkökolmio. Vapausasteet $\mathcal{A}^k$ avaruudessa $\mathcal{R}^k$ ovat seuraavat lineaariset funktionaalit: reunaintegraalit*

$$\left\{ \int_{\hat{e}_i} (\hat{\boldsymbol{t}}_i \cdot \hat{\boldsymbol{u}})\, \hat{\varphi}\, d\hat{s}\,, \quad \forall \hat{\varphi} \in \mathbb{P}_{k-1}(\hat{e}_i), \quad i = 0, 1, 2 \right\}$$

ja sisäintegraalit

$$\left\{ \int_{\hat{K}} \hat{\boldsymbol{u}} \cdot \hat{\boldsymbol{\varphi}}\, d\hat{x}\,, \quad \forall \hat{\boldsymbol{\varphi}} \in \mathbb{P}_{k-2}(\hat{K}, \mathbb{R}^2) \right\},$$

missä $k \geq 1$ on elementin kertaluku.

Väite 3.1. *Olkoon äärellinen elementti kolmikko* $(\hat{K}, \mathcal{R}^k, \mathcal{A}^k)$*. Tällöin*

1) Vapausasteet $\mathcal{A}^k$ *toteuttavat jatkuvuusehdon 3.1, ja näin ollen määritelty elementti on konformi avaruudessa* $H(\mathrm{curl})$*.*

2) Vapausasteiden joukko $\mathcal{A}^k$ *on lineaarisesti riippumaton avaruudessa* $\hat{R}$*, eli ratkaisu* $\hat{\boldsymbol{u}} \in \hat{R}$ *määräytyy yksikäsitteisesti vapausasteista* $\mathcal{A}^k$*.*

Todistus. Katso [30] tai [28, luku 5.5]. □

Tässä tutkielmassa keskitymme vain alimman kertaluvun elementtiin. Valitsemme siis $k = 1$. Tällöin funktioavaruutemme $\hat{R}$ on

$$\hat{R} := \mathcal{R}^1 = \left\langle \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} \hat{x}_2 \\ -\hat{x}_1 \end{pmatrix} \right\rangle . \tag{3.2}$$

ja vapausasteemme ovat

$$\hat{A} := \mathcal{A}^1 = \left\{ \hat{\alpha}_i = \int_{\hat{e}_i} (\hat{\boldsymbol{t}}_i \cdot \hat{\boldsymbol{u}})\, d\hat{s}\,, \quad i = 0, 1, 2 \right\} . \tag{3.3}$$

Alimman kertaluvun elementti $(\hat{K}, \hat{R}, \hat{A})$ on selvästi lineaarinen.

3.4 Lokaalit kantafunktiot referenssielementillä

Muodostaaksemme kannan $\hat{\boldsymbol{N}}_0, \hat{\boldsymbol{N}}_1, \hat{\boldsymbol{N}}_2$ avaruuteen $\hat{R}$ vapausasteiden (3.3) mukaan, vaadimme, että

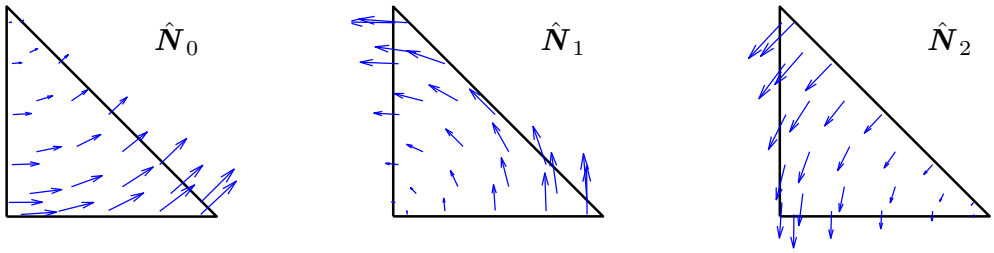
$$\hat{\alpha}_i(\hat{\boldsymbol{N}}_j) = \delta_{ij}\,, \quad i, j = 0, 1, 2\,. \tag{3.4}$$

Kun kantafunktiot saavat arvon yksi vain vastaavan vapausasteen kohdalla, ja häviävät muilla vapausasteilla, laskentataakka kevenee huomattavasti. Globaali matriisi saadaan koottua laskemalla elementtikohtaiset lokaalit matriisit ja lisäämällä lokaalien matriisien kontribuutiot globaaliin matriisiin. Lisäksi globaalista matriisista tulee harva, ns. nauhamatriisi.

Vaatimuksesta (3.4) seuraa lineaarinen yhtälöryhmä, jossa ratkaistaan kantavektoreiden $\hat{\boldsymbol{N}}_i$ kertoimet avaruuden $\hat{R}$ yleisessä kannassa (3.2). Kun yhtälöryhmä on ratkaistu, saamme

$$\hat{\boldsymbol{N}}_0(\hat{\boldsymbol{x}}) = \begin{pmatrix} 1 - \hat{x}_2 \\ \hat{x}_1 \end{pmatrix}, \quad \hat{\boldsymbol{N}}_1(\hat{\boldsymbol{x}}) = \begin{pmatrix} -\hat{x}_2 \\ \hat{x}_1 \end{pmatrix}, \quad \hat{\boldsymbol{N}}_2(\hat{\boldsymbol{x}}) = \begin{pmatrix} -\hat{x}_2 \\ \hat{x}_1 - 1 \end{pmatrix} . \tag{3.5}$$

On helppo varmistaa, että kanta täyttää vaatimuksen (3.4). Kantafunktiot ovat visualisoitu kuvassa 3.5.



Kuva 3.5: Kantafunktiot $\hat{\boldsymbol{N}}_0$, $\hat{\boldsymbol{N}}_1$, $\hat{\boldsymbol{N}}_2$.

3.5 Piola-muunnos

Olemme nyt määritelleet geometrian $\hat{K}$, funktioavaruuden $\hat{R}$ ja vapausasteet $\hat{A}$, joiden avulla laskimme kantafunktiot $\hat{\boldsymbol{N}}_i$.

Elementtimenetelmässä meidän täytyy kyetä muuntamaan kantafunktiot referenssielementiltä alueen Ω verkon elementille K, koska haluamme suorittaa integroinnin referenssielementillä. H^1-FEM:issä, missä vapausasteina ovat ratkaisun arvot solmupisteissä, tämä tehdään yksinkertaisesti kaavalla $\boldsymbol{N}_i = (\hat{\boldsymbol{N}}_i \circ \boldsymbol{F}_K^{-1})(\boldsymbol{x})$. Tässä $\boldsymbol{N}_i$ tarkoittaa elementin K reunaan i liittyvää lokaalia kantafunktiota. Avaruudessa $H(\mathrm{curl}, \Omega)$ emme voi käyttää tätä kuvausta, koska sitä käyttämällä tangentiaalikomponentista ei tule jatkuvaa.

Avaruuteen $H(\mathrm{curl}, \Omega)$ sopiva muunnos on ns. *Piola-muunnos* (katso [38, sivu 7] ja [11, sivu 97]):

$$\boldsymbol{N}_i = \mathcal{P}_K(\hat{\boldsymbol{N}}_i) = (D\boldsymbol{F}_K^{-T}\, \hat{\boldsymbol{N}}_i) \circ \boldsymbol{F}_K^{-1}(\boldsymbol{x})\,.$$

Affiinikuvauksen $\boldsymbol{F}_K$ (3.1) Jakobiaani $D\boldsymbol{F}_K$ on vakiomatriisi $\mathbf{B}_K$. Voimme siis kirjoittaa Piola-muunnoksen muotoon

$$\boldsymbol{N}_i = \mathcal{P}_K(\hat{\boldsymbol{N}}_i) = \mathbf{B}_K^{-T}(\hat{\boldsymbol{N}}_i \circ \boldsymbol{F}_K^{-1})(\boldsymbol{x})\,. \tag{3.6}$$

Elementtimenetelmän yksi perusideoista on suorittaa variaatiomuodon (2.4) integrointi referenssielementillä. Operaattori curl muuntuu seuraavasti (katso [38, sivu 8]):

$$\mathrm{curl}\, \boldsymbol{N}_i = \frac{1}{\det \mathbf{B}_K}\, \mathrm{curl}\, \hat{\boldsymbol{N}}_i\,,$$

joten voimme integroida referenssielementillä seuraavasti:

$$\int_K \mathrm{curl}\, \boldsymbol{N}_i\, \mathrm{curl}\, \boldsymbol{N}_j\, d\boldsymbol{x} = \frac{1}{\det \mathbf{B}_K} \int_{\hat{K}} \mathrm{curl}\, \hat{\boldsymbol{N}}_i\, \mathrm{curl}\, \hat{\boldsymbol{N}}_j\, d\hat{\boldsymbol{x}}\,. \tag{3.7}$$

3.6 Jatkuvuusehdon pakottaminen

Lineaarisessa Nédélecin kolmioelementissä on ainoastaan kolme kantafunktiota ja ne liittyvät elementin reunoihin. Jokaiseen reunaan liittyy joko yksi tai kaksi kolmioelementtiä. Meidän tarvitsee tutkia vain sitä tapausta, missä yksi mielivaltainen reuna e liittyy kahteen kolmioon $K_1 = \boldsymbol{F}_1(\hat{K})$ ja $K_2 = \boldsymbol{F}_2(\hat{K})$. Olkoon $\boldsymbol{t}_1$ reunaan e liittyvä tangentti elementin K_1 mukaan suunnistettuna, ja vastaavasti $\boldsymbol{t}_2$ reunaan e liittyvä tangentti elementin K_2 mukaan suunnistettuna. Koska suunnistamme molemmat kolmiot vastapäivään, pätee, että $t_1 = -t_2$. Olkoon $\boldsymbol{N}$ reunaan e liittyvää globaali kantafunktio. Funktio $\boldsymbol{N}$ on nollasta eroava vain kolmioissa K_1 ja K_2. Merkitään kantafunktion $\boldsymbol{N}$ rajoittumia elementteihin K_1 ja K_2 vastaavasti $\boldsymbol{N}_1$ ja $\boldsymbol{N}_2$. Tilanne on oleellisesti sama kuin kuvassa 3.3.

Jotta tangentiaalikomponenttien jatkuvuusehto täyttyisi, vaadimme, että

$$\boldsymbol{N}_1 \cdot \boldsymbol{t}_1 + \boldsymbol{N}_2 \cdot \boldsymbol{t}_2 = 0$$

reunalla e. Tämä ehto täyttyy, kun valitsemme

$$\boldsymbol{N}_1 := \mathcal{P}_1(\hat{\boldsymbol{N}}_i)\,, \quad \boldsymbol{N}_2 := -\mathcal{P}_2(\hat{\boldsymbol{N}}_j)\,,$$

missä i ja j ovat reunaan e liittyvien kantafunktioiden indeksit referenssielementillä. On selvää, että kantafunktioiden transformaatiossa täytyy ottaa huomioon myös reunan suunnistus. Tämä perustellaan tarkemmin Schneebelin raportissa [38].

3.7 Käytännön toteutus ja numeeriset testit

Tähän tutkielmaan liittyen laadin elementtimenetelmään perustuvan tietokonetoteutuksen aiemmissa luvuissa kuvatulle lineaariselle Nédélecin kolmioelementille. Ohjelmointialustaksi valittiin Matlab [24]. Käytimme Matlabin lisäosaa PDE-toolbox [25] elementtiverkkojen käsittelemiseen ja visualisointiin. Itse ratkaisijan tärkeimmät ohjelmakoodin osat on sisällytetty liitteisiin.

Matlab-funktiossa `solver_simple` (katso liite B.1) on kuvattu yksinkertainen ratkaisija. Malliongelman ratkaiseminen koostuu neljästä vaiheesta. Ensin kootaan kaikki data joka kuvaa ratkaistavaa ongelmaa: funktiot μ, κ ja $\boldsymbol{f}$ ja alueen Ω elementtiverkko. Seuraavaksi tehdään laskennallisesti vaativin osa: globaalien matriisien kokoaminen. Globaalit matriisit kootaan rutiinissa `assemble_matrices`. Tämän jälkeen asetetaan homogeeninen Dirichlet'n reunaehto funktiolla `set_homogenous_dirichlet`. Nämä rutiinit ovat sisällytettynä rutiiniin `solver` (katso liite B.2) Viimeisenä vaiheena ratkaistaan saatu lineaarinen yhtälöryhmä. Tämän yhtälöryhmän ratkaiseminen onnistuu tehokkaasti Matlabin omilla funktioilla. Tämän jälkeen ratkaisu voidaan esimerkiksi visualisoida tai jälkikäsitellä sitä jollain muulla tavalla.

Elementtiratkaisijan toimivuuden varmistamiseksi tutkitaan konvergoiko numeerinen ratkaisu $\boldsymbol{v}$ kohti tarkkaa ratkaisua $\boldsymbol{u}$ kun verkkoa tihennetään. Tätä varten tarvitsemme testitapauksia, joille tiedämme tarkan ratkaisun. Seuraavat kaksi testiä ovat Schneebelin raportista [38, sivu 16].

Testi 1: malliongelma (2.1)-(2.2), missä

$$\Omega = [-1,1]^2\,, \quad \mu \equiv 1\,, \quad \kappa \equiv 1\,, \quad \boldsymbol{f}(\boldsymbol{x}) = \begin{pmatrix} 3 - x_2^2 \\ 3 - x_1^2 \end{pmatrix}. \tag{3.8}$$

Tämän ongelman tarkka ratkaisu on

$$\boldsymbol{u}(\boldsymbol{x}) = \begin{pmatrix} 1 - x_2^2 \\ 1 - x_1^2 \end{pmatrix}.$$

Testi 2: malliongelma (2.1)-(2.2), missä

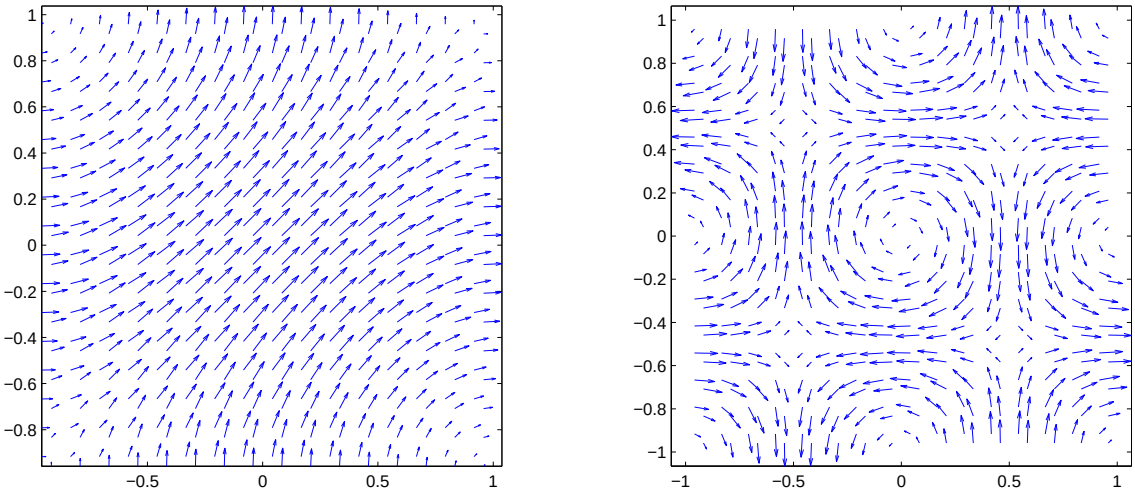
$$\Omega = [-1,1]^2\,, \quad \mu \equiv 1\,, \quad \kappa \equiv 1\,, \quad \boldsymbol{f}(\boldsymbol{x}) = (2\pi^2 + 1) \begin{pmatrix} \cos \pi x_1 \sin \pi x_2 \\ -\sin \pi x_1 \cos \pi x_2 \end{pmatrix}. \tag{3.9}$$

Sen tarkka ratkaisu on

$$\boldsymbol{u}(\boldsymbol{x}) = \begin{pmatrix} \cos \pi x_1 \sin \pi x_2 \\ -\sin \pi x_1 \cos \pi x_2 \end{pmatrix}.$$

On helppo varmistaa, että tarkat ratkaisut todellakin toteuttavat reunaehdon (2.2). Kuvassa 3.6 on näiden kahden testin numeeriset ratkaisut säännöllisessä verkossa. Ihmisen silmälle tarkka ratkaisu näyttää täsmälleen samalta joten tarkan ratkaisun kuvia ei ole laitettu esille lainkaan.

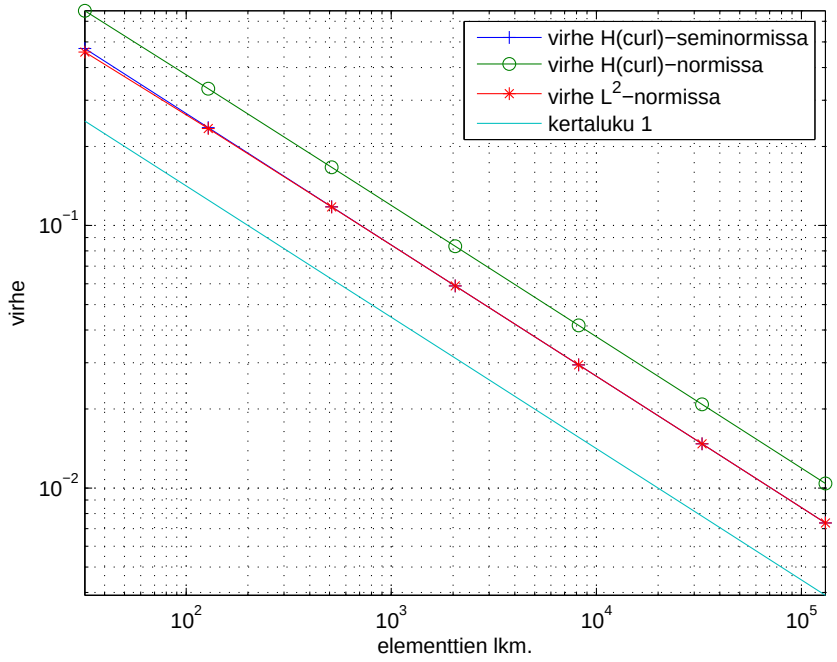
Molemmat testit ratkaistiin ensin verkossa, jossa on 32 elementtiä. Virheet tarkan ratkaisun ja numeerisen ratkaisun välillä laskettiin $H(\mathrm{curl})$-seminormissa, $H(\mathrm{curl})$-normissa sekä myös L^2-normissa. Tämän jälkeen tehtiin kuusi tihennystä, ja jokaisen tihennyksen jälkeen laskettiin virhe tarkan ratkaisun ja numeerisen ratkaisun välillä. Tulokset ovat taulukossa 3.1 ja kuvissa 3.7 ja 3.8. Taulukossa virheiden yhteyteen on liitetty myös virheen suhde edelliseen virheeseen. Tämä suhde on saatu kertomalla senhetkiseen elementtiverkkoon liittyvä virhe kahdella, ja jakamalla edellinen virhe tällä luvulla. Kuten taulukosta nähdään, jokaisen tihennyksen jälkeen elementtien määrä nelinkertaistuu, ja virheen määrä suurin piirtein puolittuu kaikilla normeilla. Konvergenssi on siis luokkaa $\mathcal{O}(h)$, missä h kuvaa elementtiverkon karakteristista kokoa.



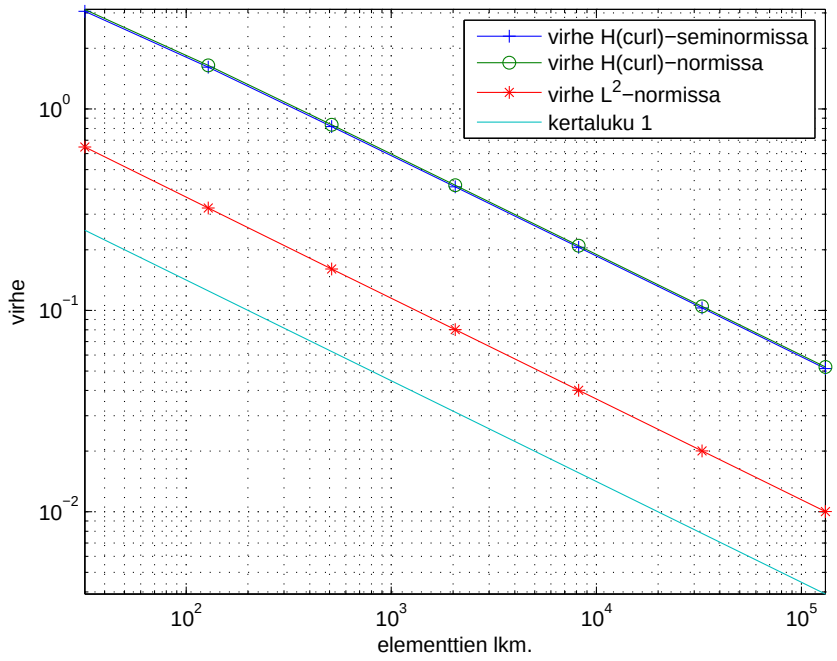
Kuva 3.6: Vasemmalla Testin 1 (3.8) ja oikealla Testin 2 (3.8) numeerinen ratkaisu verkolla jossa on 512 elementtiä.

Taulukko 3.1: Approksimaatioiden konvergenssi.

	verkko	# elem.	$\lvert\boldsymbol{u}-\boldsymbol{v}\rvert_{\mathrm{curl}}$		$\lVert\boldsymbol{u}-\boldsymbol{v}\rVert_{\mathrm{curl}}$		$\lVert\boldsymbol{u}-\boldsymbol{v}\rVert_{L^2}$	
Testi 1 (3.8)	1	32	4.72×10^{-1}	-	6.58×10^{-1}	-	4.58×10^{-1}	-
	2	128	2.36×10^{-1}	1.00	3.32×10^{-1}	0.99	2.34×10^{-1}	0.98
	3	512	1.18×10^{-1}	1.00	1.67×10^{-1}	1.00	1.18×10^{-1}	0.99
	4	2048	5.89×10^{-2}	1.00	8.33×10^{-2}	1.00	5.89×10^{-2}	1.00
	5	8192	2.95×10^{-2}	1.00	4.17×10^{-2}	1.00	2.95×10^{-2}	1.00
	6	32768	1.47×10^{-2}	1.00	2.08×10^{-2}	1.00	1.47×10^{-2}	1.00
	7	131072	7.37×10^{-3}	1.00	1.04×10^{-2}	1.00	7.37×10^{-3}	1.00
Testi 2 (3.9)	1	32	3.05×10^{-0}	-	3.12×10^{-0}	-	6.47×10^{-1}	-
	2	128	1.61×10^{-0}	0.95	1.65×10^{-0}	0.95	3.22×10^{-1}	1.00
	3	512	8.19×10^{-1}	0.99	8.34×10^{-1}	0.99	1.61×10^{-1}	1.00
	4	2048	4.11×10^{-1}	1.00	4.18×10^{-1}	1.00	8.02×10^{-2}	1.00
	5	8192	2.06×10^{-1}	1.00	2.09×10^{-1}	1.00	4.01×10^{-2}	1.00
	6	32768	1.03×10^{-1}	1.00	1.05×10^{-1}	1.00	2.00×10^{-2}	1.00
	7	131072	5.14×10^{-2}	1.00	5.24×10^{-2}	1.00	1.00×10^{-2}	1.00



Kuva 3.7: Testin 1 (3.8) numeerisen ratkaisun konvergenssi.



Kuva 3.8: Testin 2 (3.9) numeerisen ratkaisun konvergenssi.

4 Numeerisen ratkaisun virhe

Numeeristen ratkaisujen laskemisen lisäksi on välttämätöntä kyetä varmistamaan ratkaisun kelpoisuus. Kuitenkin ratkaisun kelpoisuuden varmistaminen ohitetaan usein sillä perusteella, että käytetyn numeerisen menetelmän ajatellaan olevan luotettava. Monimutkaisten ongelmien ratkaisut voivat kuitenkin muuttua hyvinkin paljon muuttamalla vain hieman käytettyä elementtiverkkoa tai osittaisdifferentiaaliyhtälön kertoimia. Numeerisen menetelmän luotettavuuteen ei voi luottaa myöskään silloin, jos ongelma sisältää vahvoja singulariteetteja. Tässäkin tapauksessa ratkaisun laatu periaatteessa paranee kun ongelma ratkaistaan uudestaan tiheämmällä elementtiverkolla, mutta mahdollisen singulariteetin vaikutus ei välttämättä katoa mihinkään. Tämän takia numeerisen menetelmän rinnalle on tarpeellista kehittää koneisto ratkaisujen virheen arvioimiseksi.

Tämän luvun sisältö perustuu suurelta osin Neittaanmäen ja Repinin kirjan [32] ensimmäiseen lukuun.

4.1 Virheen lähteet

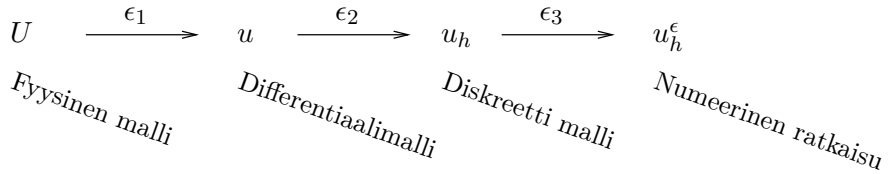
Matemaattisessa mallinnuksessa pyritään kuvaamaan luonnossa tapahtuvia fyysisiä prosesseja matemaattisten mallien avulla. Yleensä nämä matemaattiset mallit eivät täysin tarkasti kuvaa mallintamaansa fyysistä prosessia. Mallit sisältävät usein osittaisdifferentiaaleja ja kuten hyvin tiedetään, osittaisdifferentiaaliyhtälöitä kyetään ratkaisemaan tarkasti vain tietyissä erikoistapauksissa.

Joskus alkuperäiset matemaattiset mallit ovat niin monimutkaisia, että niiden ratkaiseminen edellyttää mallin yksinkertaistamista. Mallia voidaan yksinkertaistaa esimerkiksi jättämällä osittaisdifferentiaaliyhtälöstä pois termejä, joiden uskotaan tai tiedetään olevan mitättömän pieniä muihin termeihin verrattuna. Tässä on painotettava sitä, että joskus yksinkertaistukset voivat olla hyvinkin heuristisia. Kyse ei kuitenkaan ole täysin käsienheiluttelusta, vaan yleensä mallit kyetään jossain määrin validioimaan suorittamalla yksinkertaisia oikean elämän testejä mallinnettavalle ilmiölle. Olkoon U fyysinen prosessi ja u matemaattisen mallin ratkaisu. *Matemaattisen mallin virhe* on

$$\epsilon_1 = |U - u| .$$

Tässä $|\cdot|$ ymmärretään laajassa mielessä jonkinlaiseksi mitaksi.

Kuten aiemmin mainittiin, fyysisiä prosesseja kuvataan usein osittaisdifferentiaaliyhtälöillä. Tarkkojen ratkaisujen ominaisuuksia voidaan tutkia analyyttisen matematiikan keinoin, mutta käytännössä näitä tarkkoja ratkaisuja ei yleensä kyetä laskemaan. Tämän takia rat-



Kuva 4.1: Virheen lähteet.

kaisua u joudutaan arvioimaan äärellisellä joukolla yksinkertaisia funktiota. Tässä tutkielmassa arvioimme malliyhtälön (2.1)-(2.2) tarkkaa ratkaisua äärellisellä määrällä paloittain lineaarisilla polynomeilla. Matemaattisen mallin sijaan ratkaistaan siis matemaattista mallia approksimoiva ongelma. Tämä approksimoitu malli on määritelty diskreetissä verkossa, jonka karakteristista kokoa kuvaa parametri h. Olkoon u_h tämän approksimoidun ongelman ratkaisu. *Approksimointivirhe* on

$$\epsilon_2 = |u - u_h| \,.$$

Tämä virhe sisältää sen virheen mikä aiheutuu kun osittaisdifferentiaaliyhtälö ratkaistaan esimerkiksi elementtimenetelmällä.

Myöskään äärellisiä ongelmia ei ratkaista tarkasti. Olkoon u_h^ϵ äärellisen ongelman tietokoneella laskettu ratkaisu. Virhe

$$\epsilon_3 = |u_h - u_h^\epsilon|$$

on *numeerisen algoritmin virhe*. Se sisältää pyöristysvirheet, virheet jotka syntyvät kun iteratiiviset prosessit katkaistaan annetun toleranssin perusteella ja virheet ohjelmakoodissa. Kuvassa 4.1 on havainnollistettu matemaattiseen mallinnukseen liittyvät vaiheet ja virheet.

Fyysistä prosessia U arvioidaan siis funktiolla u_h^ϵ. Kun $\|\cdot\|$ on jokin sopiva normi, virheelle pätee

$$\|U - u_h^\epsilon\| \leq \epsilon_1 + \epsilon_2 + \epsilon_3 \,.$$

Jos matemaattinen malli vastaa hyvin fyysistä prosessia, voidaan virheen ϵ_1 olettaa olevan mitätön verrattuna virheisiin ϵ_2 ja ϵ_3. Yleensä näin oletetaankin, ja virheen estimoinnissa arvioidaan termiä

$$\|u - u_h^\epsilon\| \leq \epsilon_2 + \epsilon_3 \,. \tag{4.1}$$

Tavoitteena on siis saada luotettavia arvioita virheille ϵ_2 ja ϵ_3. Jos tämä tavoite saavutetaan, voidaan varmistua tietokonesimulointien luotettavuudesta sekä arvioida numeeristen ratkaisujen luotettavuutta.

Jatkossa numeerista ratkaisua u_h^ϵ merkitään lyhyemmin v:llä.

4.2 Virheen arviointi

Numeerisen ratkaisun virheen (4.1) arvioimiseen on kaksi eri lähestymistapaa: virhettä voidaan arvioida ennen numeerisen ratkaisun laskemista ja sen jälkeen.

Kun virhettä arvioidaan ennen numeerisen ratkaisun laskemista, puhutaan *a priori* -lähestymistavasta. Merkitään $\mathcal{D}$:llä ratkaistavaan ongelmaan liittyvää informaatiota, kuten reunaehtoja ja aluetta jossa ongelma on määritelty. Olkoon $\mathcal{F}_h$ ongelmaan liittyvä näytteistys, jota tarvitaan kun ongelmasta tehdään äärellisulotteinen tehtävä. Luku h tarkoittaa jälleen diskreetin mallin karakteristista kokoa. A priori -tyyppiset virhearviot ovat muotoa

$$\|u - v\| \leq M_{\oplus}(\mathcal{D}, \mathcal{F}_h, h, u)\,,$$

missä $M_{\oplus}$ on jokin funktionaali. Koska tämä funktionaali riippuu myös tuntemattomasta tarkasta ratkaisusta u, on a priori virhearvioilla lähinnä teoreettista merkitystä. Tämänkaltaiset virhearviot kuvaavat virheen asymptootista käyttäytymistä, eli näillä virhearvioilla voidaan varmistaa ovatko numeeriset ratkaisut periaatteen tasolla oikeita. A priori -tyyppisten virhearvioiden teoria on nykypäivänä hyvin tunnettu (katso esim. [12, 40]).

A posteriori -lähestymistavassa virhettä arvioidaan numeerisen ratkaisun laskemisen jälkeen. A posteriori -tyyppisissä virhearvioissa käytetään tarkan ratkaisun u sijaan numeerista ratkaisua. Yleinen muoto a posteriori -tyyppiselle virhearviolle on

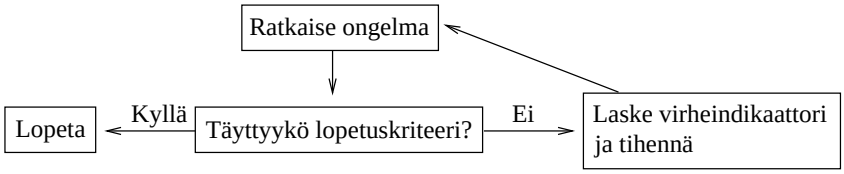
$$\|u - v\| \leq M_{\oplus}(\mathcal{D}, \mathcal{F}_h, h, v)\,. \tag{4.2}$$

Alunperin tällaisten virhearvioiden tarve syntyi adaptiivisten FEM-ratkaisijoiden myötä. Näissä ratkaisijoissa käytettyä elementtiverkkoa tihennetään adaptiivisesti niissä elementeissä, joissa numeerisen ratkaisun virhe on suurta. Suosittu Babuškan ja Rheinboldtin esittämä menetelmä perustuu ns. residuaalin negatiivisen normin arviointiin (katso [6, 7]). Toinen suosittu menetelmä on Zienkiewiczin ja Zhun kehittämä ns. gradientin keskiarvoistaminen (katso [44, 45]). Tässä menetelmässä oletetaan, että numeerisen ratkaisun gradienttien keskiarvojen erot kuvaavat virheen jakaumaa. Molemmille menetelmille on tyypillistä että ne riippuvat vahvasti numeerisesta ratkaisumenetelmästä.

Toisin kuin nämä kaksi menetelmää, Repinin kehittämät funktionaaliset a posteriori virheestimaatit (katso [32, 35]) eivät riipu siitä, millä menetelmällä numeerinen ratkaisu v on laskettu. Tämän vuoksi yhtälön (4.2) funktionaali $M_{\oplus}$ riippuu vain $\mathcal{D}$:stä ja numeerisesta ratkaisusta v. Funktionaalisen ylärajan $M_{\oplus}$ lisäksi voidaan laskea myös funktionaalinen alaraja $M_{\ominus}$. Myöskin alaraja riippuu vain $\mathcal{D}$:stä ja numeerisesta ratkaisusta v. Yleinen muoto funktionaalisille a posteriori virhearvioille on

$$M_{\ominus}(\mathcal{D}, v) \leq \|u - v\| \leq M_{\oplus}(\mathcal{D}, v).$$

Painotettakoon, että funktionaalisten virhe-estimaattien kontekstissa ylärajalla tarkoitetaan funktionaalia, jonka antamat arvot ovat aina suurempia tai yhtäsuuria kuin todellinen virhe. Samoin alaraja on funktionaali, jonka antamat arvot ovat aina pienempiä tai yhtäsuuria kuin todellinen virhe.



Kuva 4.2: Adaptiivisen ratkaisijan toiminta.

4.3 Adaptiiviset ratkaisijat ja virheindikaattorit

Kuten jo mainittiin, a posteriori -virhearvioiden tarve syntyi alunperin kun alettiin käyttämään adaptiivisia FEM-ratkaisijoita. Adaptiivisten ratkaisijoiden ideana on tarkentaa ratkaisua koko verkon tihentämisen sijaan tihentämällä verkkoa vain niissä elementeissä, joissa virhe on suurinta.

Adaptiivisten ratkaisijoiden toimintaperiaate on seuraava. Ongelma ratkaistaan ensin harvassa verkossa. Tämän jälkeen lasketaan virhe jokaisessa elementissä, ja tihennetään niissä elementeissä, joissa virhe on suurinta. Sitten lasketaan uusi ratkaisu uudella verkolla. Tämä lopetetaan sitten kun ratkaisun katsotaan olevan tarpeeksi hyvä.

Kuten varmasti on jo käynyt selväksi, tarkkaa virhettä ei tiedetä, sillä ongelman tarkkaa ratkaisua ei tiedetä. Virhettä tulee sen takia arvioida numeerisin keinoin. Jos elementeittäin lasketun virheen tiedetään olevan tarpeeksi tarkka, voidaan adaptiivinen ratkaiseminen lopettaa esimerkiksi silloin, kun yhteenlaskettu virhe on tarpeeksi pieni. Edes jossain määrin tarkan ylärajan laskeminen virheelle on kuitenkin laskennallisesti raskasta. Koska adaptiivisessa prosessissa lasketaan ongelmalle ratkaisu monta kertaa peräkkäin, on adaptiivinen ratkaiseminen itsessäänkin jo hyvin raskasta. Tämän takia on tarpeellista saada tietoa virheestä nopeasti. Adaptiivisten ratkaisijoiden yhteydessä käytetäänkin ns. *virheindikaattoreita*, jotka arvioivat virheen suhteellista jakaumaa käytetyssä elementtiverkossa. Adaptiivista prosessia virheindikaattoria käyttäen on havainnollistettu kuvassa 4.2.

Yleensä virheindikaattorit eivät tarjoa ylärajoja virheelle. Indikaattorien tarkoitus onkin vain kertoa nopeasti virheen suhteellinen jakauma, jotta verkkoa osataan tihentää oikeissa alueissa. Virheindikaattoreita käytettäessä adaptiiviselle prosessille ei yleensä ole luotettavaa lopettamiskriteeriä. Adaptiivinen prosessi voidaan lopettaa esimerkiksi määrätyn iteraatioluvun jälkeen, tai tekemällä virheindikaattorista itsestään jotain johtopäätöksiä. Isoja teollisuusongelmia ratkaistaessa on järkevää ratkaista ongelma valmiiksi tiheässä verkossa, ja sen jälkeen suorittaa 1-3 adaptiivista iteraatiota.

5 Funktionaaliset virhe-estimaatit

A posteriori virhearviointi Maxwellin yhtälöille on suhteellisen uusi tutkimuskohde. Varsinkin tämän tutkielman malliyhtälölle tutkimusta on yllättävän vähän. Kaikki aikaisempi julkaistu materiaali perustuu suurelta osin residuaalipohjaisiin menetelmiin. Residuaaliin perustuvia estimaatteja on tutkittu artikkeleissa [9, 10, 27, 28]. Artikkelissa [18] esitetään estimaatteja epäkonformeille approksimaatioille. Artikkelissa [33] esitetään ns. Zienkiewicz-Zhu -tyyppinen virhe-estimaatti, joka on ekvivalentti artikkelissa [9] esitetyn estimaatin kanssa.

Funktionaaliset virhe-estimaatit tarjoavat täysin uuden lähestymistavan virheen arviointiin. Tämän tyyppisiin estimaatteihin liittyvä teoria on esitetty kattavasti kirjoissa [32, 35]. Funktionaaliset virhe-estimaatit eivät ole riippuvaisia menetelmästä, jolla numeerinen ratkaisu $\boldsymbol{v}$ on saatu. Tämä tarkoittaa sitä, että estimaatit ovat sovellettavissa kaikille konformeille menetelmille. Näiden estimaattien toinen tärkeä ominaisuus on se, että ne sisältävät vain globaaleja vakioita. Globaalit vakiot eivät riipu käytetystä elementtiverkosta, vaan ainoastaan alueesta Ω. Globaalit vakiot tarvitsee siis laskea vain kerran tietylle alueelle.

Funktionaalisia virhe-estimaatteja Maxwellin yhtälöille on johtanut Repin [31, 36] ja Hannukainen [17]. Artikkelissa [36] johdetaan virheylärajat tapauksille $\kappa > 0$ ja $\kappa = 0$. Tapauksen $\kappa > 0$ yläraja on *tarkka*[1], mutta *herkkä*[2] pienillä funktion κ arvoilla. Tapaus $\kappa = 0$ on myös tarkka. Sama yläraja tapaukselle $\kappa > 0$ esitetään myös artikkelissa [17]. Tässä artikkelissa esitetään lisäksi yläraja kompleksiselle κ, $\mathrm{Real}(\kappa) \geq 0$. Tämä yläraja ei ole tarkka. Artikkelissa [31] esitellään jälleen sama yläraja tapaukselle $\kappa > 0$ ja sen lisäksi kaksi muuta uutta ylärajaa. Ensimmäinen uusi yläraja on tapaukselle $\kappa \geq 0$. Tämä yläraja toimii hyvin pienillä funktion κ arvoilla, mutta se on herkkä isoilla funktion κ arvoilla. Sen tarkkuutta ei myöskään kyetä todistamaan. Toinen uusi yläraja tapaukselle $\kappa \geq 0$ on johdettu hienostuneemmalla tavalla, ja se ei ole herkkä millään funktion κ arvolla. Sen lisäksi tämä yläraja on tarkka. Artikkelissa [31] esitellään myös alaraja.

Tässä tutkielmassa keskitytään ainoastaan ylärajaan tapaukselle $\kappa > 0$. Tämä yläraja ei sisällä lainkaan globaaleja vakioita ja se on herkkä pienillä funktion κ arvoilla. Numeerisia testejä tälle ylärajalle on tähän mennessä tehnyt ainoastaan Hannukainen [17]. Molemmat esimerkit ovat avaruudessa $\mathbb{R}^3$, ja funktio $\kappa = 1$. Tätä ylärajaa on kuitenkin syytä tutkia myös muilla funktion κ arvoilla. Ainakin teoria antaa ymmärtää, että ylärajan ei pitäisi käyttäytyä hyvin pienillä funktion κ arvoilla. Itse asiassa, funktion κ ei edes tarvitse olla vakio, se voi olla positiivinen rajoitettu funktio tai jopa paloittain vakio. Tämän takia voimme tehdä virhearvioita myös sellaisille ongelmille, jossa alue Ω koostuu useammasta eri materiaalista,

[1] Tarkkuudella tarkoitetaan sitä, että virhe-estimaatti antaa mielivaltaisen tarkkoja arvioita numeerisen ratkaisun virheelle, kun elementtien määrä kasvaa äärettömään.

[2] Herkkyydellä tarkoitetaan sitä, että yläraja antaa suhteettoman korkeita arvioita todelliselle virheelle, ellei ylärajan laskemiseen käytetä hyvin paljon laskentakapasiteettia.

kuten vedestä ja ilmasta. Tällaisessa tapauksessa κ olisi paloittain vakio.

Esitämme myös alarajan tämän tutkielman malliongelmalle. Tämä alaraja on hyvin tunnettu elliptisille ongelmille (katso [26]). Artikkelissa [31] tämä sama alaraja johdetaan eri lähestymistavalla. Ylä- ja alarajan lisäksi tässä tutkielmassa esitellään ja testataan kaksi uutta Repinin johtamaa virheindikaattoria. Tämän tutkielman testituloksia hyödynnettiin artikkelissa [3], jossa allekirjoittanut oli yksi tekijöistä. Kyseisessä artikkelissa tehtiin numeeriset testit kaikille artikkelissa [31] esitetyille virherajoille, sekä tässä tutkielmassa esitetyille virheindikaattoreille.

Numeerisen ratkaisun virhettä on tapana mitata ns. *energianormissa*, joka on

$$\| \boldsymbol{v} \|^2 := a(\boldsymbol{v}, \boldsymbol{v}) \,.$$

Energianormi voidaan ymmärtää painotettuna $H(\mathrm{curl})$-normina. Jos malliongelmassa permeabiliteetti $\mu \equiv 1$ ja $\kappa \equiv 1$, on energianormi yhtäpitävä $H(\mathrm{curl})$-normin kanssa.

5.1 Minorantti

Alarajasta käytetään myös nimitystä minorantti. Tässä esitetty minorantti pätee kaikille elliptisille ongelmille, eli ongelmille jotka voidaan esittää muodossa (2.5).

Määritelmä 5.1 (Minorantti)**.** *Olkoon $\boldsymbol{v} \in V$ malliongelman numeerinen ratkaisu. Numeerisen ratkaisun virheen alaraja, eli minorantti, on*

$$M_{\ominus}(\boldsymbol{v}, \boldsymbol{w}) := 2(J(\boldsymbol{v}) - J(\boldsymbol{w}))\,, \tag{5.1}$$

missä $\boldsymbol{w} \in V$ on mielivaltainen funktio.

Tällä minorantilla on seuraava ominaisuus:

Lause 5.1. *Olkoon $\boldsymbol{u}$ malliongelman tarkka ratkaisu, ja $\boldsymbol{v} \in V$ sen numeerinen ratkaisu. Tällöin*

$$\| \boldsymbol{u} - \boldsymbol{v} \|^2 \geq M_{\ominus}(\boldsymbol{v}, \boldsymbol{w})\,, \quad \forall \boldsymbol{w} \in V\,.$$

Todistus. Tiedetään, että $\boldsymbol{u}$ toteuttaa yhtälön (2.5). Käyttämällä energianormia ja muodon $a(\cdot,\cdot)$ bilineaarisuutta voimme kirjoittaa

$$\begin{aligned} \| \boldsymbol{u} - \boldsymbol{v} \|^2 &= a(\boldsymbol{u} - \boldsymbol{v}, \boldsymbol{u} - \boldsymbol{v}) \\ &= a(\boldsymbol{u}, \boldsymbol{u}) - 2a(\boldsymbol{u}, \boldsymbol{v}) + a(\boldsymbol{v}, \boldsymbol{v})\,. \end{aligned} \tag{5.2}$$

Yhtälössä (5.2) termi $2a(\boldsymbol{u}, \boldsymbol{v})$ on yhtäpitävä termin $2l(\boldsymbol{v})$ kanssa, joten voimme korvata sen.

Lisäämällä $2(-a(\boldsymbol{u},\boldsymbol{u})+l(\boldsymbol{u}))=0$ yhtälöön (5.2) saamme

$$\begin{aligned}\| \boldsymbol{u}-\boldsymbol{v} \|^2 &= 2\left(\frac{1}{2}a(\boldsymbol{v},\boldsymbol{v})-l(\boldsymbol{v})-\frac{1}{2}a(\boldsymbol{u},\boldsymbol{u})+l(\boldsymbol{u})\right)\\ &= 2\left(J(\boldsymbol{v})-J(\boldsymbol{u})\right)\,.\end{aligned}$$

Koska $\boldsymbol{u}$ minimoi energiafunktionaalin (2.6), on totta että $J(\boldsymbol{u}) \leq J(\boldsymbol{w})$ jokaiselle $\boldsymbol{w} \in V$, joten saamme

$$\| \boldsymbol{u}-\boldsymbol{v} \|^2 \geq 2(J(\boldsymbol{v})-J(\boldsymbol{w}))\,, \quad \forall \boldsymbol{w} \in V\,.$$

□

Jotta minorantille $M_{\ominus}$ saataisiin järkeviä arvoja, pitää ensin laskea numeerinen ratkaisu $\boldsymbol{v}$ ja sen jälkeen laskea uusi numeerinen ratkaisu $\boldsymbol{w}$ tihennetyllä verkolla. Numeerinen ratkaisu $\boldsymbol{w}$ on tarkempi ratkaisu kuin $\boldsymbol{v}$, joten minorantin $M_{\ominus}$ arvo on varmasti alaraja.

5.2 Majorantti

Ylärajaa kutsutaan myös majorantiksi. Majorantti määritellään seuraavasti.

Määritelmä 5.2 (Majorantti)**.** *Olkoon $\boldsymbol{v} \in V$ malliongelman numeerinen ratkaisu. Numeerisen ratkaisun virheen yläraja, eli majorantti, on*

$$M_{\oplus}(\boldsymbol{v},y) := \|\kappa^{-1/2}\,\boldsymbol{r}(\boldsymbol{v},y)\|^2 + \|\mu^{1/2}\,d(\boldsymbol{v},y)\|^2\,, \tag{5.3}$$

missä $y \in H^1(\Omega)$ on mielivaltainen funktio ja

$$\boldsymbol{r}(\boldsymbol{v},y) := \boldsymbol{f}-\kappa\,\boldsymbol{v}-\underline{\operatorname{curl}}\,y\,, \tag{5.4}$$

$$d(\boldsymbol{v},y) := y-\mu^{-1}\operatorname{curl}\boldsymbol{v}\,. \tag{5.5}$$

Majorantissa esiintyvää funktiota y kutsutaan vapaaksi parametriksi. Majorantilla on seuraava ominaisuus:

Lause 5.2. *Olkoon $\boldsymbol{u}$ malliongelman tarkka ratkaisu ja $\boldsymbol{v} \in V$ sen numeerinen ratkaisu. Tällöin*

$$\| \boldsymbol{u}-\boldsymbol{v} \|^2 \leq M_{\oplus}(\boldsymbol{v},y)\,, \quad \forall y \in H^1(\Omega)\,.$$

Todistus. Lisäämällä termi $\int_\Omega(-\mu^{-1}\operatorname{curl}\boldsymbol{v}\operatorname{curl}\boldsymbol{w}-\kappa\,\boldsymbol{v}\cdot\boldsymbol{w})\,d\boldsymbol{x}$ yhtälöön (2.4) saamme

$$\begin{aligned}\int_\Omega(\mu^{-1}\operatorname{curl}(\boldsymbol{u}-\boldsymbol{v})\operatorname{curl}\boldsymbol{w}+\kappa(\boldsymbol{u}-\boldsymbol{v})\cdot\boldsymbol{w})\,d\boldsymbol{x} &=\\ &= \int_\Omega(\boldsymbol{f}\cdot\boldsymbol{w}-\mu^{-1}\operatorname{curl}\boldsymbol{v}\operatorname{curl}\boldsymbol{w}-\kappa\,\boldsymbol{v}\cdot\boldsymbol{w})\,d\boldsymbol{x}\,.\end{aligned} \tag{5.6}$$

Huomautuksen 2.3 mukaan voimme esittää nollan yhtälönä $\int_\Omega (\underline{\operatorname{curl}}\, y \cdot \boldsymbol{w} - y \operatorname{curl} \boldsymbol{w})\, d\boldsymbol{x}$, missä $y \in H^1(\Omega)$ on vapaa parametri. Lisäämällä tämä yhtälön (5.6) oikealle puolelle saamme

$$\begin{aligned} &\int_\Omega (\boldsymbol{f} \cdot \boldsymbol{w} - \mu^{-1} \operatorname{curl} \boldsymbol{v} \operatorname{curl} \boldsymbol{w} - \kappa\, \boldsymbol{v} \cdot \boldsymbol{w} - \underline{\operatorname{curl}}\, y \cdot \boldsymbol{w} + y \operatorname{curl} \boldsymbol{w})\, d\boldsymbol{x} \\ = &\int_\Omega (\boldsymbol{f} - \kappa\, \boldsymbol{v} - \underline{\operatorname{curl}}\, y) \cdot \boldsymbol{w}\, d\boldsymbol{x} + \int_\Omega (y - \mu^{-1} \operatorname{curl} \boldsymbol{v}) \operatorname{curl} \boldsymbol{w}\, d\boldsymbol{x} \end{aligned} \tag{5.7}$$

Käyttämällä määritelmiä (5.4) ja (5.5) voimme kirjoittaa yhtälön (5.7) muotoon

$$\begin{aligned} &\int_\Omega \boldsymbol{r}(\boldsymbol{v}, y) \cdot \boldsymbol{w}\, d\boldsymbol{x} + \int_\Omega d(\boldsymbol{v}, y) \operatorname{curl} \boldsymbol{w}\, d\boldsymbol{x} \\ = &\int_\Omega \kappa^{-1/2} \boldsymbol{r}(\boldsymbol{v}, y) \cdot \kappa^{1/2} \boldsymbol{w}\, d\boldsymbol{x} + \int_\Omega \mu^{1/2} d(\boldsymbol{v}, y) \mu^{-1/2} \operatorname{curl} \boldsymbol{w}\, d\boldsymbol{x} \\ \leq &\, \|\kappa^{-1/2} \boldsymbol{r}(\boldsymbol{v}, y)\|\, \|\kappa^{1/2} \boldsymbol{w}\| + \|\mu^{1/2} d(\boldsymbol{v}, y)\|\, \|\mu^{-1/2} \operatorname{curl} \boldsymbol{w}\| \end{aligned} \tag{5.8}$$

Viimeisessä epäyhtälössä käytimme Hölderin epäyhtälöä[3]. Muuttamalla yhtälön (5.6) oikea puoli muotoon (5.8) ja valitsemalla $\boldsymbol{w} = \boldsymbol{u} - \boldsymbol{v}$, yhtälöstä (5.6) tulee

$$\begin{aligned} &\int_\Omega (\mu^{-1} |\operatorname{curl}(\boldsymbol{u} - \boldsymbol{v})|^2 + \kappa |\boldsymbol{u} - \boldsymbol{v}|^2)\, d\boldsymbol{x} \leq \\ &\qquad \leq \|\kappa^{-1/2}\, \boldsymbol{r}(\boldsymbol{v}, y)\|\, \|\kappa^{1/2}\, (\boldsymbol{u} - \boldsymbol{v})\| + \|\mu^{1/2}\, d(\boldsymbol{v}, y)\|\, \|\mu^{-1/2} \operatorname{curl}(\boldsymbol{u} - \boldsymbol{v})\|\,. \end{aligned} \tag{5.9}$$

Yhtälön (5.9) vasen puoli on selvästi $\|\!|\, \boldsymbol{u} - \boldsymbol{v}\, |\!\|^2$. Käyttämällä Cauchy-Schwarzin epäyhtälöä[4] saamme

$$\begin{aligned} \|\!|\, \boldsymbol{u} - \boldsymbol{v}\, |\!\|^2 \leq &\left(\|\kappa^{-1/2}\, \boldsymbol{r}(\boldsymbol{v}, y)\|^2 + \|\mu^{1/2}\, d(\boldsymbol{v}, y)\|^2 \right)^{1/2} \times \\ &\times \left(\|\kappa^{1/2}\, (\boldsymbol{u} - \boldsymbol{v})\|^2 + \|\mu^{-1/2} \operatorname{curl}(\boldsymbol{u} - \boldsymbol{v})\|^2 \right)^{1/2}. \end{aligned}$$

Oikean puolen jälkimmäinen termi on $\|\!|\, \boldsymbol{u} - \boldsymbol{v}\, |\!\|$, joten saamme

$$\|\!|\, \boldsymbol{u} - \boldsymbol{v}\, |\!\| \leq \left(\|\kappa^{-1/2}\, \boldsymbol{r}(\boldsymbol{v}, y)\|^2 + \|\mu^{1/2}\, d(\boldsymbol{v}, y)\|^2 \right)^{1/2},$$

ja korottamalla puolittain toiseen potenssiin saamme lopullisen epäyhtälön

$$\|\!|\, \boldsymbol{u} - \boldsymbol{v}\, |\!\|^2 \leq \|\kappa^{-1/2}\, \boldsymbol{r}(\boldsymbol{v}, y)\|^2 + \|\mu^{1/2}\, d(\boldsymbol{v}, y)\|^2\,.$$

□

[3] Hölderin epäyhtälö: $\int_\Omega \boldsymbol{f} \cdot \boldsymbol{g}\, d\boldsymbol{x} \leq \|\boldsymbol{f}\|\, \|\boldsymbol{g}\|$

[4] Cauchy-Schwarzin epäyhtälö: $ab + cd \leq \sqrt{a^2 + c^2}\, \sqrt{b^2 + d^2}$.

Numeeriselle virheelle $\|\!| \boldsymbol{u} - \boldsymbol{v} |\!\|^2$ saadaan yläraja kun majorantti (5.3) minimoidaan vapaan parametrin y suhteen. Koska tunnemme majorantin minimoivan tarkan parametrin $y = \mu^{-1}\mathrm{curl}\,\boldsymbol{u}$, voimme esittää seuraavan lauseen.

Lause 5.3. *Olkoon $\boldsymbol{u}$ malliongelman tarkka ratkaisu ja $\boldsymbol{v}$ sen numeerinen ratkaisu. Tällöin*

$$\|\!| \boldsymbol{u} - \boldsymbol{v} |\!\|^2 = \inf_y M_\oplus(\boldsymbol{v}, y)\,.$$

Todistus. Valitsemalla $y = \mu^{-1}\mathrm{curl}\,\boldsymbol{u}$ majorantista tulee

$$\begin{aligned} M_\oplus(\boldsymbol{v}, y) &= \|\kappa^{-1/2}\, r(\boldsymbol{v}, y)\|^2 + \|\mu^{1/2}\, d(\boldsymbol{v}, y)\|^2 \\ &= \|\kappa^{1/2}(\boldsymbol{u} - \boldsymbol{v})\|^2 + \|\mu^{-1/2}\mathrm{curl}(\boldsymbol{u} - \boldsymbol{v})\|^2 \\ &= \int_\Omega \kappa|\boldsymbol{u} - \boldsymbol{v}|^2 + \mu^{-1}|\mathrm{curl}(\boldsymbol{u} - \boldsymbol{v})|^2\, d\boldsymbol{x} \\ &= \|\!| \boldsymbol{u} - \boldsymbol{v} |\!\|^2\ . \end{aligned}$$

□

Majorantin minimoiva tarkka parametri sisältää tuntemattoman termin $\mathrm{curl}\,\boldsymbol{u}$. Saadaksemme arvon majorantille on kyettävä arvioimaan tätä termiä. Mitä tarkemmin sitä kyetään arvioimaan, sitä tarkempi yläraja saadaan. Luvussa 5.4 esitetään kaksi tapaa vapaan parametrin y laskemiseen.

5.3 Virheindikaattorit

On myös hyödyllistä kyetä arvioimaan virheen jakaumaa eri normeissa. Käyttämällä määritelmän 5.2 majorantin kahta termiä erikseen voimme määritellä virheindikaattorit

$$I_r(\boldsymbol{v}, y) := \|\kappa^{-1/2}\boldsymbol{r}(\boldsymbol{v}, y)\|^2\,, \tag{5.10}$$

$$I_d(\boldsymbol{v}, y) := \|\mu^{1/2} d(\boldsymbol{v}, y)\|^2\,. \tag{5.11}$$

Virheindikaattorien johtaminen tällä tavoin suoraan majorantin lausekkeesta on Repinin ideoima. Myös nämä kaksi virheindikaattoria ovat Repinin käsialaa.

Asettamalla $y = \mu^{-1}\mathrm{curl}\,\boldsymbol{u}$ näemme, että jos vapaa parametri y on tarpeeksi hyvä, indikaattori (5.10) arvioi virheen jakaumaa painotetussa L_2-normissa

$$\|\kappa^{1/2}(\boldsymbol{u} - \boldsymbol{v})\|^2\,.$$

Vastaavasti näemme että indikaattori (5.11) arvioi virheen jakaumaa painotetussa $H(\mathrm{curl})$-seminormissa

$$\|\mu^{-1/2}\mathrm{curl}(\boldsymbol{u} - \boldsymbol{v})\|^2\,.$$

Virheindikaattoreita käytetään yleensä adaptiivisien ratkaisijoiden yhteydessä, joten on tärkeää että ne ovat laskennallisesti kevyitä. Usein virheindikaattorit lasketaankin tekemällä tarkkuutta alentavia kompromisseja, jotta ne olisivat nopeita. Virheindikaattoreiden ei kuitenkaan ole tarkoitus olla ylärajoja tai alarajoja niille virheille joita ne approksimoivat. Niiden tarkoitus on tarjota mahdollisimman tarkka kuvaus siitä, miten virhe on suhteellisesti jakautunut käytetyn elementtiverkon elementeille.

5.4 Vapaan parametrin valinta

Numeerisen ratkaisun $\boldsymbol{v}$ laskemisen jälkeen täytyy majorantti $M_\oplus(\boldsymbol{v}, y) = M_\oplus(y)$ minimoida vapaan parametrin y suhteen. Ensin esitämme miten majorantti minimoidaan globaalisti y:n suhteen. Matemaattiselta kannalta tämä on helppoa, sillä majorantti on kvadraattisessa muodossa. Tämä lähestymistapa johtaa uuteen variaatio-ongelmaan.

Vapaa parametri y voidaan laskea myös GA-tyyppisellä menetelmällä (katso [1, 8, 44, 45]). GA-tyyppiset menetelmät perustuvat gradientin keskiarvoistamiseen, eli *silottamiseen*. Tämän tutkielman malliongelman tapauksessa puhuttaisiin curl:in keskiarvoistamisesta. Silottamiseen perustuvat menetelmät ovat laskennallisesti edullisempia kuin globaali minimointi. Globaalilla minimoinnilla saatu y on kuitenkin tarkempi kuin silottamisella saadut vapaat parametrit, varsinkin jos ongelma on haastava. Virheindikaattoreissa on kuitenkin järkevää käyttää tällaisia menetelmiä niiden nopeuden vuoksi.

5.4.1 Globaali minimointi

Olkoon $\varphi \in H^1(\Omega)$. Ensin laskemme majorantin Gateaux-derivaatan $D(M_\oplus)$:

$$\begin{aligned} D(M_\oplus) &= \left\{ \frac{d}{dt} M_\oplus(y + t\,\varphi) \right\}_{t=0} \\ &= \int_\Omega \left(-2\kappa^{-1}(\boldsymbol{f} - \kappa\,\boldsymbol{v} - \underline{\operatorname{curl}}\, y) \cdot \underline{\operatorname{curl}}\, \varphi + 2\mu(y - \mu^{-1}\operatorname{curl} \boldsymbol{v})\varphi \right) d\boldsymbol{x}\,. \end{aligned}$$

Vaadimme, että $D(M_\oplus) = 0$ jokaiselle $\varphi \in H^1(\Omega)$:

$$\int_\Omega \left(-2\kappa^{-1}(\boldsymbol{f} - \kappa\,\boldsymbol{v} - \underline{\operatorname{curl}}\, y) \cdot \underline{\operatorname{curl}}\, \varphi + 2\mu(y - \mu^{-1}\operatorname{curl} \boldsymbol{v})\varphi \right) d\boldsymbol{x} = 0\,, \quad \forall \varphi \in H^1(\Omega)\,.$$

Järjestämällä termit uudelleen saamme

$$\begin{aligned} \int_\Omega \left(\kappa^{-1}\, \underline{\operatorname{curl}}\, y \cdot \underline{\operatorname{curl}}\, \varphi + \mu\, y\, \varphi \right) d\boldsymbol{x} = \\ = \int_\Omega \left((\kappa^{-1}\boldsymbol{f} - \boldsymbol{v}) \cdot \underline{\operatorname{curl}}\, \varphi + \operatorname{curl} \boldsymbol{v}\, \varphi \right) d\boldsymbol{x}\,, \quad \forall \varphi \in H^1(\Omega)\,. \quad (5.12) \end{aligned}$$

Merkitsemällä vasenta puolta symmetrisellä bilineaarimuodolla $\tilde{a}(y, \varphi)$ ja oikeaa puolta lineaarisella muodolla $\tilde{l}(\varphi)$, voimme kirjoittaa yhtälön (5.12) muotoon

$$\tilde{a}(y, \varphi) = \tilde{l}(\varphi)\,, \quad \forall \varphi \in H^1(\Omega)\,. \tag{5.13}$$

Variaatiomuoto (5.13) tulee ratkaista H^1-FEM:illä. Kun ratkaisu on saatu, saamme arvon majorantille. Kun vapaa parametri y minimoidaan globaalisti, pitää y laskea samassa tai tiheämmässä verkossa kuin alkuperäinen verkko, jolla malliongelmalle laskettiin numeerinen ratkaisu $\boldsymbol{v}$. Varsinkin silloin kun voimafunktio $\boldsymbol{f}$ yhtälössä (2.1) on monimutkainen, täytyy verkkoa tihentää monta kertaa ennen kuin y minimoi majorantin tarpeeksi hyvin.

On tärkeää huomata se, että käyttäessämme kaksiulotteista lineaarista Nédélec-FEM:iä vapausasteemme ovat kolmioiden reunat. Lineaarisessa H^1-FEM:issä vapausasteemme ovat solmupisteet. Kolmioihin jaetussa kaksiulotteisessa elementtiverkossa on n. kolme kertaa enemmän reunoja kuin solmupisteitä. Vapaan parametrin y tarkkuus on siten selvästi alakynnessä verrattuna numeeriseen ratkaisuun $\boldsymbol{v}$, tai minorantin vapaaseen parametriin $\boldsymbol{w}$. Tämän takia on odotettavissa, että minorantti konvergoi tarkkaan virheeseen nopeammin kuin majorantti. Verkkoa on todennäköisesti tihennettävä useasti ennen kuin y on tarpeeksi hyvä, varsinkin jos voimafunktio $\boldsymbol{f}$ on monimutkainen. Yksi vaihtoehto kasvattaa y:n laatua olisi käyttää korkeamman kertaluvun elementtiä. Kvadraattisessa H^1-FEM:issä vapausasteina olisivat sekä reunat että solmupisteet. Tämä tietenkin kasvattaa laskentataakkaa. Kolmiulotteisessa avaruudessa käyttäisimme Nédélec-FEM:iä myös y:n laskemiseen, joten tätä ongelmaa ei tulisi vastaan ollenkaan, ja y:n arvot olisivat selvästi laadukkaampia jo ennen ensimmäistäkään tihennystä.

5.4.2 Silottamiseen perustuva menetelmä

Esimerkiksi adaptiivisissa ratkaisijoissa halutaan mahdollisimman nopeasti saada tietoa virheen jakautumasta käytetyssä elementtiverkossa. Tällöin edellisessä luvussa esitetty globaali minimointi on melko raskas tapa laskea vapaa parametri y. Tällöin on järkevää käyttää silottamiseen perustuvia menetelmiä. Silottamiseen perustuvissa menetelmissä ideana on jälkikäsitellä numeerista ratkaisua keskiarvoistamalla sen differentiaalien arvoja. Malliongelmamme tapauksessa tarkka vapaa parametri on $\mu^{-1}\mathrm{curl}\,\boldsymbol{u}$. Tälle ongelmalle sopiva silottaminen olisi siten arvojen $\mu^{-1}\mathrm{curl}\,\boldsymbol{v}$ keskiarvoistaminen.

Tässä esitettävässä silottamisessa keskiarvoistetaan arvoja solmuihin. Silottaminen suoritetaan samalla verkolla kuin millä numeerinen ratkaisu $\boldsymbol{v}$ on laskettu. Toisin kuin globaalissa minimoinnissa, silottamiseen perustuvissa menetelmissä verkon tihentämisellä ei ole mitään vaikutusta. Solmupisteen arvo saadaan, kun lasketaan yhtälön $\mu^{-1}\mathrm{curl}\,\boldsymbol{v}$ arvot jokaisessa elementissä, joka liittyy solmuun. Nämä arvot kerrotaan elementtinsä pinta-aloilla ja lasketaan yhteen. Tulos jaetaan kaikkien solmuun liittyvien elementtien pinta-alojen summalla.

Formaalisti menetelmä voidaan esittää seuraavasti. Olkoon alueen Ω elementtiverkossa

solmut P_i, $i = 1, \ldots, m$. Jokaiseen solmuun P_i liittyy elementit K_j, $j = 1, \ldots, n$. Vapaan parametrin arvo solmussa P_i saadaan kaavasta

$$y_i = \frac{\sum_{j=1}^{n} (\mu^{-1} \mathrm{curl}\, \boldsymbol{v})|_{K_j} \, |K_j|}{\sum_{j=1}^{n} |K_j|}, \tag{5.14}$$

missä $|K_j|$ on elementin K_j pinta-ala. Parametrin arvot muualla kuin solmupisteissä saadaan lineaarisesti interpoloimalla solmupisteiden arvoista. Näin saatu funktio kuuluu avaruuteen $H^1(\Omega)$, ja on oleellisesti samanlainen FEM-funktio kuin mitä globaalilla minimoinnilla saadaan kun käytetään lineaarista elementtiä. Voimme siis käyttää H^1-FEM:in lineaarista kolmioelementtiä tämän yhtälön arvojen laskemiseen.

On huomattava, että tällaiseen silottamiseen perustuvassa menetelmässä oletetaan että keskiarvoistettavat arvot ovat vakioita. Numeerisen ratkaisun $\boldsymbol{v}$ curl-arvot sekä magneettisen permeabiliteetin μ arvot täytyvät siten olla elementeittäin vakioita. Luvussa 3.4 esitetyt kantafunktiomme ovat lineaarisia joten niiden curl-arvot ovat vakioita. Tämän takia myös numeerisen ratkaisun $\boldsymbol{v}$ curl:in arvot ovat vakioita jokaisessa elementissä. Ainoaksi ongelmaksi voi siten muodostua funktio μ. Jos sen arvot eivät ole elementeittäin vakioita, tätä silottamismenetelmää ei voi käyttää.

6 Numeeriset testit

Tässä luvussa teemme numeerisia testejä edellisessä luvussa esitetyille funktionaalisille a posteriori virhe-estimaateille. Ensin testaamme minorantin ja majorantin suorituskykyä. Sitten testaamme edellisessä luvussa johdettujen kahden virheindikaattorin suorituskyvyn adaptiivisessa kontekstissa.

Määritelmien 5.1 ja 5.2 minorantti ja majorantti arvioivat termiä $\| \boldsymbol{u} - \boldsymbol{v} \|^2$. Virhettä ei kuitenkaan yleensä ilmoiteta tässä muodossa. Virhe ilmoitetaan yleensä suhteessa numeerisen ratkaisun energianormiin. Suhteellinen arvo

$$\frac{\| \boldsymbol{u} - \boldsymbol{v} \|}{\| \boldsymbol{v} \|}$$

on hyödyllisempi tieto kuin majorantin arvo itsessään. Majorantin arvo voi nimittäin olla hyvin suuri, mutta yleensä tällaisessa tapauksessa numeerisen ratkaisun normi on myös suuri luku. Tällöin majorantin arvo ei anna hyvää kuvaa virheestä. Vastaavasti majorantin arvo voi olla todella pieni, mutta yleensä silloin myös numeerisen ratkaisun normi on samaa kertaluokkaa. Kaikissa tämän luvun taulukoissa on ilmoitettu minorantin ja majorantin arvojen sijaan suhteellinen arvo, vaikkei taulukkojen otsikkokentät niin annakaan ymmärtää.

Majorantin $M_\oplus$ tehokkuutta voidaan mitata *tehokkuusindeksillä* (katso [32, s. 180])

$$I_{eff} := \frac{\sqrt{M_\oplus}}{\| \boldsymbol{u} - \boldsymbol{v} \|} \geq 1 \,. \tag{6.1}$$

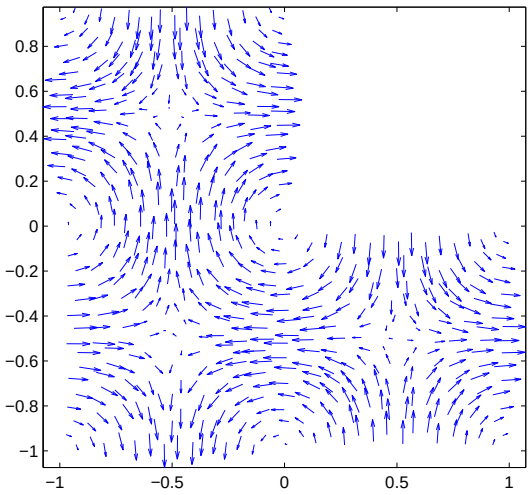
Tehokkuusindeksi kertoo kuinka hyvin majorantti käyttäytyy globaalisti. Mitä lähempänä indeksi on numeroa 1, sitä parempi majorantin arvo on kyseessä. Majorantti jolla kyetään saamaan tehokkuusindeksejä väliltä 1-3 katsotaan olevan hyvä majorantti.

Kaksi ensimmäistä testiä ovat samat kuin luvun 3.7 Testit 1 (3.8) ja 2 (3.9). Käytimme näitä testejä sen takia, koska tiedämme niiden tarkat ratkaisut, ja voimme siten laskea tarkan virheen.

Testi 3: malliongelma (2.1)-(2.2), missä

$$\Omega = [-1,1]^2 \backslash [0,1]^2 \,, \quad \mu \equiv 1 \,, \quad \kappa \equiv 1 \,, \quad \boldsymbol{f}(\boldsymbol{x}) = (2\pi^2 + 1) \begin{pmatrix} \cos \pi x_1 \sin \pi x_2 \\ -\sin \pi x_1 \cos \pi x_2 \end{pmatrix} . \tag{6.2}$$

Funktiot ovat siis samoja kuin Testissä 2, mutta alue on L-kirjaimen muotoinen. Tämän testin numeerinen ratkaisu on kuvassa 6.1. L-muotoiset alueet ovat suosittuja kun testataan menetelmien toimivuutta. Olemme erityisen kiinnostuneita suorituskyvystä origon lähettyvillä, sillä se on potentiaalisesti ongelmia aiheuttava alue.



Kuva 6.1: Testin 3 (6.2) numeerinen ratkaisu verkolla jossa on 544 elementtiä.

Testille 3 emme tiedä tarkkaa ratkaisua. Haluamme kuitenkin kyetä mittaamaan estimaattien suorituskykyä luotettavasti myös tälle tehtävälle. Tällaisessa tilanteessa ainoa mahdollisuus on laskea ns. *referenssiratkaisu* hyvin tiheällä verkolla. Jos verkko on todella tiheä, voimme olettaa sillä lasketun numeerisen ratkaisun olevan hyvin lähellä tarkkaa ratkaisua. Laskimme Testille 3 referenssiratkaisun verkolla, jossa on n. 600 000 elementtiä. 2D:ssä tämän pitäisi olla riittävän suuri verkko referenssiratkaisun laskemiseen, etenkin jos laskemme numeerisia ratkaisuja ja virhe-estimaatteja alle 10 000 elementin verkoilla.

Tämän luvun testeihin liittyvät Matlab-koodilistaukset ovat kommentteineen liitteessä B. Liitteessä B.3 on rutiinin `solver_global` ohjelmakoodi. Siinä lasketaan minorantin ja majorantin arvot peräkkäin tihennetyillä verkoilla. Tämä rutiini toimii seuraavasti:

1) Muodosta elementtiverkko ja kerää ongelmaan liityvät funktiot $\mu, \kappa, \boldsymbol{f}$ ja tarkka ratkaisu/referenssiratkaisu.

2) Laske numeerinen ratkaisu $\boldsymbol{v}$.

3) Laske vapaa parametri y ja majorantin $M_{\oplus}(\boldsymbol{v}, y)$ arvo.

4) Tihennä verkkoa.

5) Laske uudella verkolla numeerinen ratkaisu $\boldsymbol{w}$ ja minorantin $M_{\ominus}(\boldsymbol{v}, \boldsymbol{w})$ arvo.

6) Laske uudella verkolla vapaa parametri y ja majorantin $M_{\oplus}(\boldsymbol{v}, y)$ arvo.

7) Jos verkkoa on tihennetty käyttäjän asettaman määrän verran, lopeta. Muutoin mene kohtaan 4.

Ohjelmakoodi vapaan parametrin y laskemiseen globaalilla minimoinnilla on liitteessä B.4. Tämä rutiini on tavallinen FEM-ratkaisija, joten se on rakenteeltaan hyvin samanlainen kuin elementtiratkaisija joka on kuvattu liitteessä B.2. Ohjelmakoodi vapaan parametrin y laskemiseen silottamisella on liitteessä B.5. Itse majorantin laskemiseen liittyvä ohjelmakoodi on liitteessä B.6. Minorantin vapaa parametri $\boldsymbol{w}$ lasketaan samalla elementtiratkaisijalla (liite B.2), millä alkuperäinen numeerinen ratkaisu $\boldsymbol{v}$ laskettiin.

6.1 Minorantti ja majorantti

Seuraavissa testeissä majorantin vapaa parametri y on laskettu luvussa 5.4.1 esitetyllä globaalilla minimoinnilla. Tätä varten toteutin Matlabilla tavallisen H^1 elementtiratkaisijan lineaarisilla ja kvadraattisilla kantafunktioilla.

Kaikissa kolmessa testitapauksessa numeerinen ratkaisu $\boldsymbol{v}$ laskettiin hyvin harvalla verkolla. Testeissä 1 ja 2 verkon koko oli 32 elementtiä, ja Testissä 3 käytimme verkkoa, jossa oli 34 elementtiä. Tämän jälkeen tehtiin 4 tihennysiteraatiota. Jokaisessa iteraatiossa suoritettiin edellisen elementtiverkon tihennys, ja minorantin ja majorantin arvojen laskeminen. Minorantin laskemista varten laskimme jokaiselle tihennetylle verkolle uuden numeerisen ratkaisun. Majorantin laskemista varten laskimme jokaiselle tihennetylle verkolle vapaan parametrin y globaalilla minimoinnilla (majorantin arvo laskettiin myös ensimmäisellä verkolla). Lineaarisilla kantafunktioilla laskettua parametria merkitään y_{lin} ja kvadraattisilla kantafunktioilla laskettua parametria merkitään y_{kvad}. Testitulokset ovat taulukoissa 6.1, 6.2 ja 6.3. Samat tulokset on visualisoitu kuvassa 6.2.

Taulukossa 6.1 on Testin 1 (3.8) tulokset. Tiedämme tämän ongelman tarkan ratkaisun, joten kykenemme laskemaan tarkan virheen ja majorantin minimoivan tarkan parametrin:

$$\frac{\| \boldsymbol{u} - \boldsymbol{v} \|}{\| \boldsymbol{v} \|} = 0.17278\,, \qquad y = \mu^{-1}\mathrm{curl}\,\boldsymbol{u} = 2x_2 - 2x_1\,.$$

Kuten taulukosta 6.1 ja kuvan 6.2 vasemmanpuoleisesta graafista havaitaan, konvergoi minorantti nopeasti. Se on hyvin tarkka jo toisen tihennyksen jälkeen. Käyttämällä lineaarista approksimaatiota vapaalle parametrille majorantti saa tarkan virheen arvon jo ennen ensimmäistä tihennystä. Tämä johtuu siitä, että käyttämämme lineaarinen H^1-FEM kykenee kuvaamaan tarkasti kaikkia funktiota, jotka ovat käytetyn elementtiverkon mielessä paloittain lineaarisia. Se, että y on tällainen yksinkertainen funktio, johtuu yksinkertaisesta voimafunktiosta $\boldsymbol{f}$. Kvadraattinen approksimaatio parametrille y ei luonnollisesti tässä tapauksessa muuta majorantin arvoja.

Taulukko 6.1: Testin 1 (3.8) tulokset.

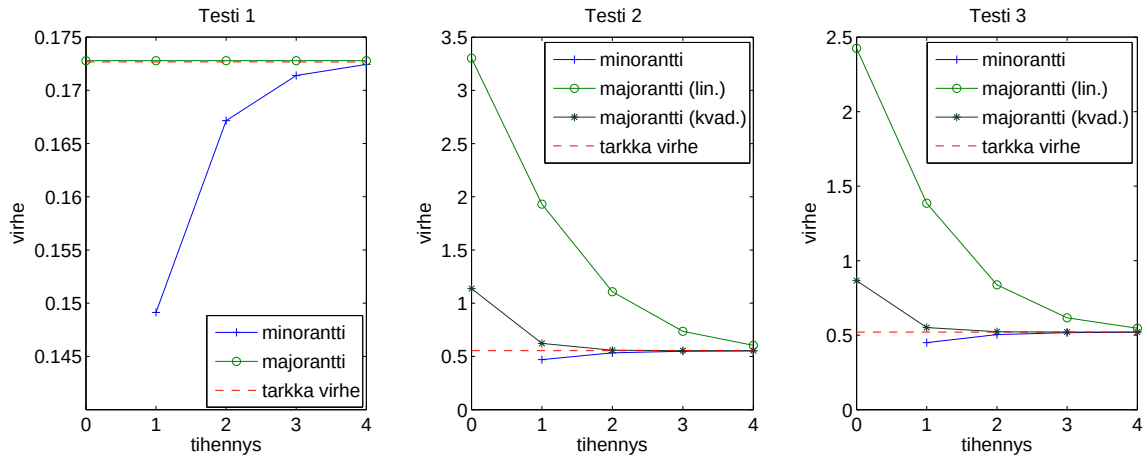
				y_{lin}		y_{kvad}	
tih.	# elem.	$M_\ominus$	$\| \boldsymbol{u} - \boldsymbol{v} \| \, / \, \| \boldsymbol{v} \|$	$M_\oplus$	I_{eff}	$M_\oplus$	I_{eff}
-	32	-	**0.17278**	0.17278	1	0.17278	1
1	128	0.14913	**0.17278**	0.17278	1	0.17278	1
2	512	0.16715	**0.17278**	0.17278	1	0.17278	1
3	2048	0.17139	**0.17278**	0.17278	1	0.17278	1
4	8192	0.17243	**0.17278**	0.17278	1	0.17278	1

Taulukko 6.2: Testin 2 (3.9) tulokset.

				y_{lin}		y_{kvad}	
tih.	# elem.	$M_\ominus$	$\| \boldsymbol{u} - \boldsymbol{v} \| \, / \, \| \boldsymbol{v} \|$	$M_\oplus$	I_{eff}	$M_\oplus$	I_{eff}
-	32	-	**0.55377**	3.30240	5.96	1.13780	2.05
1	128	0.47048	**0.55377**	1.93060	3.49	0.62204	1.12
2	512	0.53361	**0.55377**	1.10790	2.00	0.55867	1.01
3	2048	0.54876	**0.55377**	0.73589	1.33	0.55408	1.00
4	8192	0.55252	**0.55377**	0.60473	1.09	0.55379	1.00

Taulukko 6.3: Testin 3 (6.2) tulokset.

				y_{lin}		y_{kvad}	
tih.	# elem.	$M_\ominus$	$\| \boldsymbol{u} - \boldsymbol{v} \| \, / \, \| \boldsymbol{v} \|$	$M_\oplus$	I_{eff}	$M_\oplus$	I_{eff}
-	34	-	**0.52039**	2.42500	4.66	0.86629	1.66
1	136	0.44970	**0.52039**	1.38540	2.66	0.55072	1.06
2	544	0.50378	**0.52039**	0.83804	1.61	0.52276	1.00
3	2176	0.51655	**0.52039**	0.61678	1.19	0.52088	1.00
4	8704	0.51970	**0.52039**	0.54644	1.05	0.52076	1.00



Kuva 6.2: Minorantin ja majorantin konvergenssi Testeissä 1 (3.8), 2 (3.9) ja 3 (6.2).

Testin 2 tulokset ovat taulukossa 6.2 ja kuvan 6.2 keskimmäisessä graafissa. Myös tämän testin tarkka ratkaisu tunnetaan, joten tiedämme myös tarkan virheen ja majorantin minimoivan tarkan parametrin:

$$\frac{\| \boldsymbol{u} - \boldsymbol{v} \|}{\| \boldsymbol{v} \|} = 0.55377\,, \qquad y = \mu^{-1}\mathrm{curl}\,\boldsymbol{u} = -2\pi \cos \pi x_1 \cos \pi x_2\,.$$

Minorantti konvergoi jälleen nopeasti. Toisin kuin edellisessä testissä, majorantti konvergoi huomattavasti hitaammin kun käytetään lineaarista approksimaatiota y_{lin}. Tämä on odotettua, sillä voimafunktio $\boldsymbol{f}$ on nopeasti muuttuva funktio. Tarkka vapaa parametri y sisältää trigonometrisia lausekkeita, joten sen approksimoiminen lineaarisella H^1-elementillä ei tuota hyviä tuloksia. Tästä huolimatta majorantti on lähellä tarkkaa virhettä jo kolmannen tihennyksen jälkeen. Kvadraattisia approksimaatiota y_{kvad} käyttämällä majorantin arvot paranevat huomattavasti.

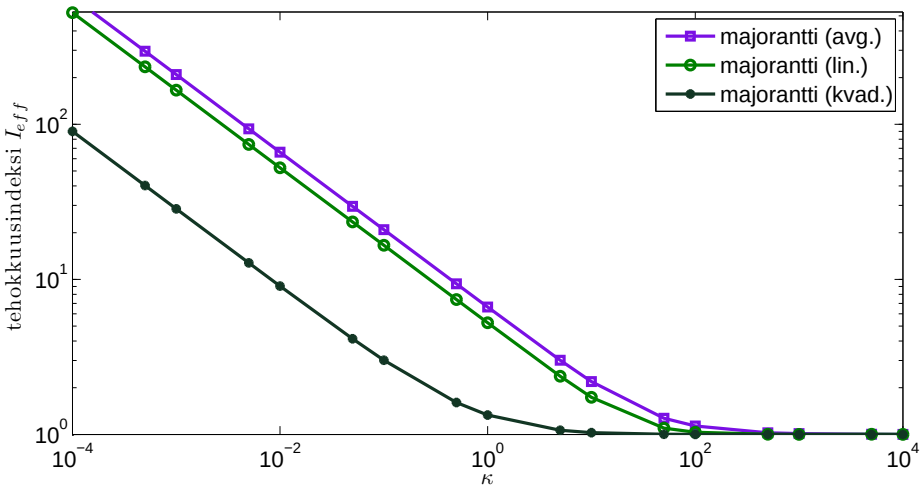
Testin 3 tulokset ovat taulukossa 6.3 ja kuvan 6.2 oikeanpuoleisessa graafissa. Tämän testin tulokset ovat vastaavia Testiin 2 verrattuna. Minorantti konvergoi nopeasti ja raskaan voimafunktion $\boldsymbol{f}$ vuoksi majorantti konvergoi hitaasti suhteessa minoranttiin, kun käytetään lineaarista approksimaatiota y_{lin}. Majorantin arvot paranevat jälleen huomattavasti kun vapaata parametria approksimoidaan kvadraattisilla elementeillä.

Kuten luvun 5 alussa mainittiin, tässä tutkielmassa käsiteltävä majorantti on teorian perusteella herkkä pienillä funktion κ arvoilla. Tätä on syytä tutkia myös numeerisesti. Tämän testaamiseksi laskimme Testille 2 (3.9) majorantin tehokkuusindeksien arvoja eri funktion κ arvoilla. Tulokset ovat taulukossa 6.4. Samat tulokset on visualisoitu kuvassa 6.3. Laskut tehtiin epäsäännöllisellä verkolla, jossa oli 88 elementtiä. Majorantin vapaalle parametrille laskettiin lineaarinen approksimaatio y_{lin} ja kvadraattinen approksimaatio y_{kvad} tihentämättä elementtiverkkoa. Tämän lisäksi käytimme luvussa 5.4.2 esiteltyä silottamiseen perustuvaa menetelmää vapaan parametrin laskemiseen. Tätä parametria merkitään y_{avg}. Tuloksista nähdään, että käyttämällä parametreja y_{avg} ja y_{lin} tehokkuusindeksit ovat lähellä toisiaan. Silottamisella laskettu parametri toimii yllättävänkin hyvin. Kvadraattinen approksimaatio y_{kvad} parantaa majorantin laatua, mutta majorantti käyttäytyy olennaisesti samalla tavalla kaikilla parametrilla. Kun κ on iso, majorantin arvo on hyvin lähellä tarkkaa virhettä. Kun funktion κ arvo laskee tarpeeksi alas, alkaa tehokkuusindeksien arvot nousta lineaarisesti ylöspäin.

Teorian mukaan majorantti $M_\oplus$ on tarkka. Tämä tarkoittaa sitä, että jos vapaa parametri y on tarpeeksi hyvä, pitäisi majorantin antaa tarkkoja arvioita virheelle. Osoittaaksemme tämän numeerisesti asetimme Testissä 2 (3.9) funktion κ arvoksi 10^{-3}, ja laskimme majorantin vapaan parametrin peräkkäisillä tihennetyillä verkoilla. Ensimmäisessä verkossa oli 88 elementtiä. Taulukosta 6.5 nähdään, että myös pienellä funktion κ arvoilla majorantti antaa mielivaltaisen tarkkoja tuloksia edellyttäen että vapaan parametrin laskemiseen käytetään paljon aikaa. Tämä testi tehtiin lähinnä sen mielenkiitoisuuden takia. Käytännössä majorantin arvojen laskemiseen ei ole mitään järkeä käyttää näin paljon aikaa.

Taulukko 6.4: Tehokkuusindeksit Testille 2 (3.9) eri muuttujan κ arvoilla.

κ	$I_{eff}(M_{\oplus}(y_{avg}))$	$I_{eff}(M_{\oplus}(y_{lin}))$	$I_{eff}(M_{\oplus}(y_{kvad}))$
10^{-4}	661.0308	524.39	90.02
10^{-3}	209.0367	165.83	28.48
10^{-2}	66.1044	52.44	9.05
10^{-1}	20.9079	16.58	3.01
10^{0}	6.6261	5.24	1.33
10^{1}	2.1911	1.74	1.03
10^{2}	1.1347	1.03	1.00
10^{3}	1.0128	1.00	1.00
10^{4}	1.0013	1.00	1.00



Kuva 6.3: Tehokkuusindeksit Testille 2 (3.9) eri muuttujan κ arvoilla.

Taulukko 6.5: Tehokkuusindeksit Testille 2 (3.9) kun $\kappa \equiv 10^{-3}$.

tih.	# elem.	$I_{eff}(M_{\oplus}(y_{lin}))$	$I_{eff}(M_{\oplus}(y_{kvad}))$
-	88	165.83	28.48
1	325	85.40	7.49
2	1408	43.17	2.13
3	5632	21.67	1.11
4	22528	10.88	1.01

6.2 Virheindikaattorit

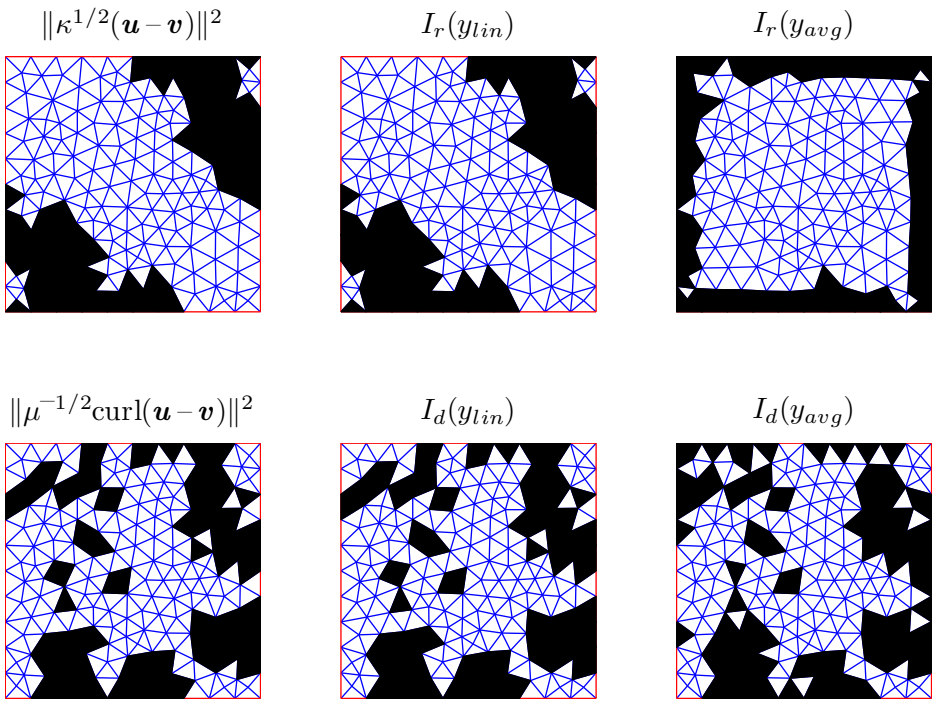
Seuraavaksi testaamme luvussa 5.3 esitetyt virheindikaattorit I_r (5.10) ja I_d (5.11). Vapaalle parametrille laskettiin verkkoa tihentämättä globaalilla minimoinnilla lineaarinen approksimaatio y_{lin} ja silottamisella y_{avg}. Kvadraattinen approksimaatio y_{kvad} jätettiin kokonaan pois, sillä sen laskeminen on suhteettoman raskasta virheindikaattoreita varten.

Tämän luvun kuvat rakentuvat siten, että ylemmässä rivissä ovat indikaattoriin I_r liittyvät kuvat, ja alemmassa rivissä indikaattoriin I_d liittyvät kuvat. Jokaisessa kuvassa mustat elementit ovat niitä elementtejä, joissa virhe on tietyn virhejakauman mukaan suurempaa kuin elementtien keskimääräinen virhe. Rivin ensimmäinen kuva on tarkka virheen jakauma. Keskimmäisessä kuvassa on indikaattorin antama virhejakauma parametrilla y_{lin}. Viimeisessä kuvassa on indikaattorin antama virhejakauma parametrilla y_{avg}.

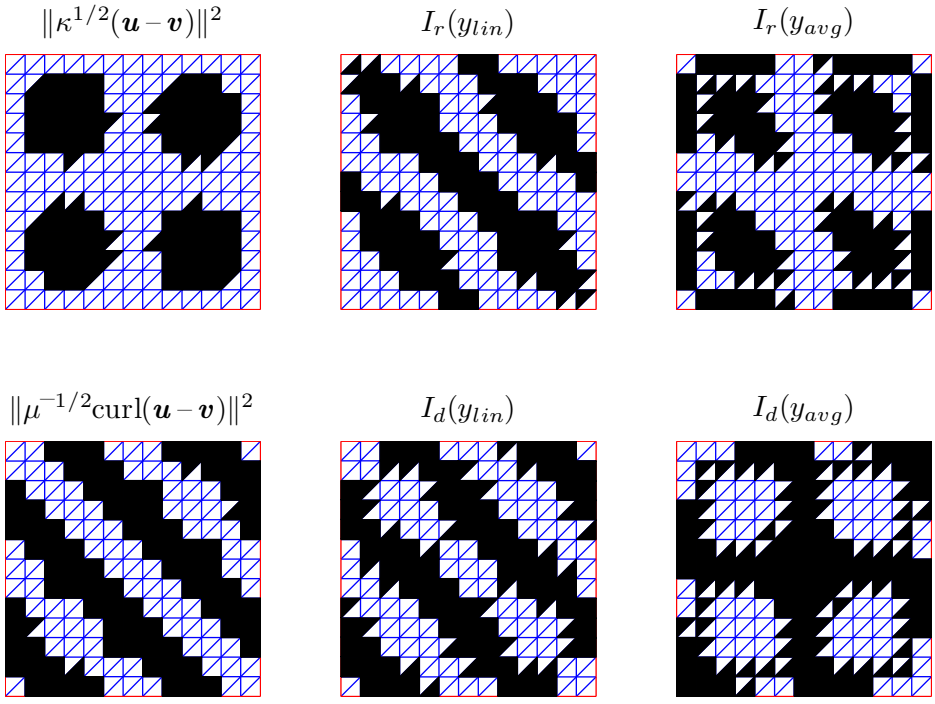
Kuvassa 6.4 ovat Testiin 1 (3.8) liittyvät kuvat. Majorantin testeissä huomasimme että majorantti antaa jo tihentämättömällä verkolla tarkan virheen arvon. Samasta syystä indikaattorit antavat tarkan jakauman parametrilla y_{lin}. Indikaattorin I_r tapauksessa parametri y_{avg} ei toimi laisinkaan, mutta indikaattorissa I_d se toimii tyydyttävästi.

Kuvissa 6.5 ja 6.6 ovat Testiin 2 (3.9) liittyvät kuvat. Kuvien ero on se, että kuvassa 6.5 on käytetty säännöllistä verkkoa ja kuvassa 6.6 epäsäännöllistä verkkoa. Näistä kuvista havaitaan, että parametrilla y_{lin} lasketut virhejakaumat ovat hyvinkin tarkkoja molemmilla indikaattoreilla. Myös parametrilla y_{avg} lasketut virhejakaumat ovat hyviä, joskin eivät niin tarkkoja kuin parametrilla y_{lin} lasketut jakaumat. Etenkin indikaattorilla I_d parametri y_{avg} antaa suhteellisen epätarkan jakauman. Verkon säännöllisyydellä ei tunnu olevan vaikutusta indikaattoreiden toimintaan.

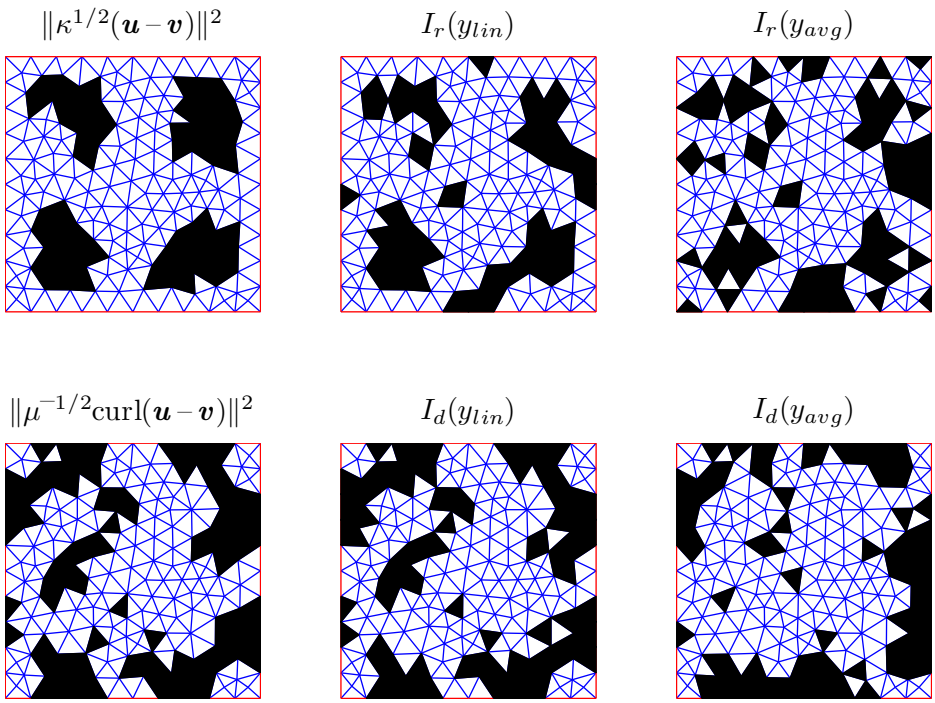
Testin 3 (6.2) kuvasta 6.7 voidaan tehdä samankaltaisia päätelmiä. Parametrilla y_{lin} lasketut virhejakaumat ovat hyviä. Parametrilla y_{avg} lasketut jakaumat sen sijaan epätarkempia. Eräs havainto, mikä voidaan kaikkien testien perusteella tehdä on se, että kummallakin parametrilla indikaattori I_r indikoi painotettua virheen L^2-normia heikommin kuin indikaattori I_d indikoi painotettua virheen $H(\mathrm{curl})$-seminormia.



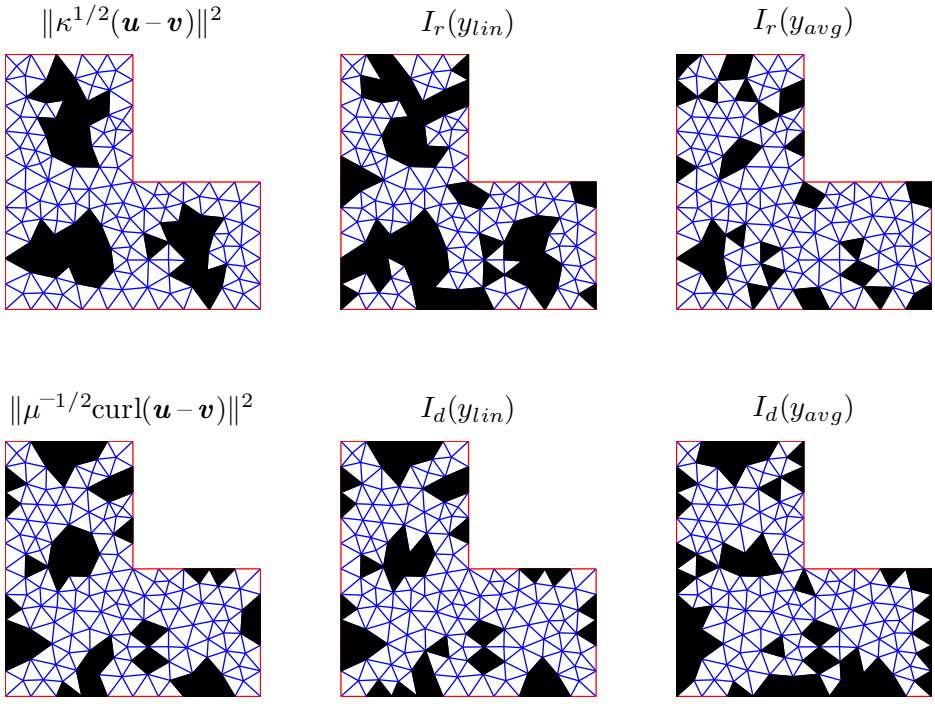
Kuva 6.4: Testi 1 (3.8): Virheindikaattorien suorituskyky epäsäännöllisessä verkossa.



Kuva 6.5: Testi 2 (3.9): Virheindikaattorien suorituskyky säännöllisessä verkossa.



Kuva 6.6: Testi 2 (3.9): Virheindikaattorien suorituskyky epäsäännöllisessä verkossa.



Kuva 6.7: Testi 3 (6.2): Virheindikaattorien suorituskyky epäsäännöllisessä verkossa.

7 Johtopäätökset

Tämän työn tarkoituksena oli tutkia funktionaalisia a posteriori virhe-estimaatteja Maxwellin yhtälöistä johdetulle toisen kertaluvun reuna-arvotehtävälle. Maxwellin yhtälöiden numeeriseksi ratkaisumenetelmäksi valitsimme elementtimenetelmän ja ensimmäisen tyypin Nédélecin elementin. Elementtiratkaisija ja virhe-estimaatit toteutettiin ohjelmointialustalla Matlab [24,25]. Ohjelmakoodilistaukset tärkeimmistä ohjelmakoodin osista ovat tämän tutkielman liitteessä B.

Majorantin $M_\oplus$ ja minorantin $M_\ominus$ testitulokset olivat odotusten mukaiset. Minorantti, vaikkakin raskas laskettava, osoittautui nopeasti konvergoivaksi kaikissa testeissä. Majorantin numeeriset testit käyttäytyivät myös odotetusti. Majorantti toimi erinomaisesti suurilla funktion κ arvoilla, mutta pienillä arvoilla tulokset olivat pessimistisiä. Osoitimme numeeristen testien avulla, että pienillä funktion κ arvoilla majorantti antaa mielivaltaisen tarkkoja tuloksia, jos parametrin y laskemiseen käytetään tarpeeksi aikaa. Majorantti on siis tarkka, mutta herkkä funktion κ suhteen.

Indikaattorien I_r ja I_d testitulokset olivat lineaarisella parametrilla y_{lin} hyvin rohkaisevia. Myös silottamisella laskettu parametri y_{avg} toimi hyvin, mutta ei yhtä hyvin kuin parametri y_{lin}. Testikuvista oli havaittavissa, että parametreista riippumatta indikaattori I_r indikoi painotettua virheen L^2-normia heikommin kuin indikaattori I_d indikoi painotettua virheen $H(\mathrm{curl})$-seminormia. Yleisesti ottaen indikaattorien testitulokset olivat positiivisia.

Tämän tutkielman anti oli jo aiemmin johdetun majoranin $M_\oplus$ testaaminen sekä Repinin johtamien uusien virheindikaattoreiden I_r ja I_d testaaminen. Tämän tutkielman testituloksia hyödynnettiin artikkelissa [3], johon allekirjoittanut toteutti kaksi muuta majoranttia, teki testitulokset ja suurimman osan kirjoitustyöstä.

Seuraavaksi olisi aiheellista tutkia tehokkaita menetelmiä vapaan parametrin y laskemiseen. Yksi vaihtoehto olisi käyttää ns. multigrid-menetelmiä. Valdman [41] on tutkinut tätä lähestymistapaa Poissonin yhtälölle ja saanut hyviä tuloksia. Toinen vaihtoehto on kehittää jälkikäsittelymenetelmiä silottamisella lasketuille parametreille. Tätä lähestymistapaa olen tutkimusryhmäni kanssa tutkinut Poissonin yhtälölle ja diffuusioyhtälölle artikkeleissa [4, 5]. Tulokset ovat lupaavia, ja vastaavia jälkikäsittelymenetelmiä voidaan kehittää myös Maxwellin yhtälöille.

8 Lähteet

[1] M. Ainsworth ja J. T. Oden, *A posteriori error estimation in finite element analysis*, Wiley and Sons, New York, 2000.

[2] A. Alonso ja A. Valli, *An optimal domain decomposition preconditioner for low-frequency time-harmonic Maxwell equations*, Math. Comp., 68(226), 607–631, 1999.

[3] I. Anjam, O. Mali, A. Muzalevsky, P. Neittaanmäki ja S. Repin, *A posteriori error estimates for a Maxwell type problem*, Russian J. Num. Anal. Math. Mod., 24(5), 395–408, 2009.

[4] I. Anjam, O. Mali, P. Neittaanmäki ja S. Repin, *A new error indicator for the Poisson problem*, Proceedings of the 10th Finnish Mechanics Days, 324–330, 2009.

[5] I. Anjam, O. Mali, P. Neittaanmäki ja S. Repin, *New indicators of approximation errors for problems in continuum mechanics*, Proceedings of the V ECCOMAS CFD 2010 Conference, 2010.

[6] I. Babuška ja W. C. Rheinboldt, *A-posteriori error estimates for the finite element method*, Internat. J. Numer. Meth. Engrg., 12, 1597–1615, 1978.

[7] I. Babuška ja W. C. Rheinboldt, *Error estimates for adaptive finite element computations*, SIAM J. Numer. Anal., 15, 736–754, 1978.

[8] I. Babuška ja R. Rodriguez, *The problem of the selection of an a posteriori error indicator based on smoothing techniques*, Internat. J. Numer. Meth. Engrg., 36, 539–567, 1993.

[9] R. Beck, R. Hiptmair, R. Hoppe ja B. Wohlmuth, *Residual based a posteriori error estimators for eddy current computation*, Math. Model. Numer. Anal., 34(1), 159–182, 2000.

[10] D. Braess ja J. Schöberl, *Equilibrated residual error estimator for edge elements*, Math. Comp., 77(262), 651–672, 2008.

[11] F. Brezzi ja M. Fortin, *Mixed and hybrid finite element methods*, Springer Series in Computational Mathematics, 15, Springer-Verlag, New York, 1991.

[12] P. G. Ciarlet, *The finite element method for elliptic problems*, North Holland, New York, 1978.

[13] CSC, *Elmer*, `<http://www.csc.fi/elmer/>` (15.4.2010).

[14] A. Friedman, *Foundations of modern analysis*, Dover, New York and London, 1970.

[15] G. H. Golub ja C. F. Van Loan, *Matrix computations*, kolmas painos, The Johns Hopkins University Press, London, 1996.

[16] J. Haataja, J. Heikonen, Y. Leino, J. Rahola, J. Ruokolainen ja V. Savolainen, *Numeeriset menetelmät käytännössä*, `<http://www.csc.fi/csc/julkaisut/oppaat/>` (15.4.2010), Picaset, Helsinki, 2002.

[17] A. Hannukainen, *Functional type a posteriori error estimates for Maxwell's equations*, Proceedings of the ENUMATH 2007 Conference, 41–48, 2008.

[18] P. Houston, I. Perugia ja D. Schötzau, *An a posteriori error indicator for discontinuous Galerkin discretizations of H(curl)-elliptic partial differential equations*, J. Numer. Anal., 27(1), 122–150, 2007.

[19] T. J. R. Hughes, *The finite element method: linear static and dynamic finite element analysis*, Dover, New York, 2000.

[20] Lauri Kahanpää, *Funktionaalianalyysi*, luentomoniste, 2006.

[21] Tero Kilpeläinen, *Mitta- ja integraaliteoria*, luentomoniste, <http://users.jyu.fi/~tuheli/MIT2008/mitta_ja_int.pdf> (15.4.2010), 2004.

[22] M. Křížek ja P. Neittaanmäki, *Finite Element Approximation of Variational Problems and Applications*, Longman Scientific and Technical, Harlow, 1990.

[23] Ch. G. Makridakis ja P. Monk, *Time-discrete finite element schemes for Maxwell's equations*, M2AN Math. Modelling and Numer. Anal., 29(2), 171–197, 1995.

[24] The MathWorks, *Matlab*, <http://www.mathworks.com/products/matlab/> (15.4.2010).

[25] The MathWorks, *PDE Toolbox*, <http://www.mathworks.com/products/pde/> (15.4.2010).

[26] S. G. Mikhlin, *Variational methods in mathematical physics*, Pergamon, Oxford, 1964.

[27] P. Monk, *A posteriori error indicators for Maxwell's equations*, J. Comp. and Appl. Math., 100, 173–190, 1998.

[28] P. Monk, *Finite element methods for Maxwell's equations*, Clarendon Press, Oxford, 2003.

[29] J. C. Nédélec, *A new family of mixed finite elements in* $\mathbb{R}^3$, Numeriche Matematik, 50, 57–81, 1986.

[30] J. C. Nédélec, *Mixed finite elements in* $\mathbb{R}^3$, Numeriche Matematik, 35, 315–341, 1980.

[31] P. Neittaanmäki ja S. Repin, *Guaranteed error bounds for conforming approximations of a Maxwell type problem*, Computational Methods in Applied Sciences, 15, 199-211, 2010.

[32] P. Neittaanmäki ja S. Repin, *Reliable methods for computer simulation. Error control and a posteriori estimates*, Elsevier, Amsterdam, 2004.

[33] S. Nicaise, *On Zienkiewicz-Zhu error estimators for Maxwell's equations*, Comptes Rendus Mathematique, 340(9), 697–702, 2005.

[34] G. L. Pollac ja D. R. Stump, *Electromagnetism*, Pearson Education, San Francisco, 2002.

[35] S. Repin, *A posteriori estimates for partial differential equations*, Walter de Gruyter, Berlin, 2008.

[36] S. Repin, *Functional a posteriori estimates for Maxwell's equation*, J. Math. Sci., 142(1), 1821–1827, 2007.

[37] W. Rudin, *Principles of Mathematical Analysis*, 1976.

[38] A. Schneebeli, *An* $H(\mathrm{curl};\Omega)$*-conforming FEM: Nédélec's elements of first type*, raportti, <http://www.dealii.org/developer/reports/nedelec/nedelec.pdf> (15.4.2010), 2003.

[39] P. Solin, *Partial differential equations and the finite element method*, John Wiley & Sons, 2005.

[40] G. Strang ja G. Fix, *An analysis of the finite element method*, Prentice Hall, Englewood Cliffs, 1973.

[41] J. Valdman, *Minimization of functional majorant in a posteriori error analysis based on* $H(\mathrm{div})$ *multigrid-preconditioned CG method*, <http://www.hindawi.com/journals/ana/2009/164519.html> (17.5.2010), Advances in Numerical Analysis, Vol. 2009, 2009.

[42] K. Yee, *Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media*, IEEE Trans. on Antennas and Propagation, AP-14(3), 302–307, 1966.

[43] J. Zhao, *Analysis of finite element approximation and iterative methods for time-dependent Maxwell problems*, väitöskirja, 2002.

[44] O. C. Zienkiewicz ja J. Z. Zhu, *A simple error estimator and adaptive procedure for practical engineering analysis*, Internat. J. Numer. Meth. Engrg., 24, 337–357, 1987.

[45] O. C. Zienkiewicz ja J. Z. Zhu, *Adaptive techniques in the finite element method*, Commun. Appl. Numer. Methods, 4, 197–204, 1988.

A Malliongelman johtaminen

Tässä liitteessä johdamme luvussa 2 esitetyn malliongelman. Johto mukailee Zhaon esitystä [43], jossa malliongelma johdetaan aikariippuvaisesta Maxwellin ongelmasta. Malliongelman voi johtaa myös Maxwellin yhtälöiden aikaharmonisesta muodosta. Tämänkaltainen johto löytyy esim. Alonson ja Vallin artikkelista [2]. Myös Monkin kirjan [28] ensimmäisessä luvussa johdetaan samankaltaisia toisen kertaluvun järjestelmiä.

Elektromagneettinen kenttä syntyy staattisista sähkövarauksista ja sähkövarauksen virtauksesta. Sähkövarauksen jakauman antaa skalaariarvoinen varaustiheysfunktio ρ ja virtauksia kuvaa virtaustiheysfunktio $\boldsymbol{J}$. Klassista makroskooppista elektromagneettista kenttää kuvaavat paikasta $\boldsymbol{x} \in \mathbb{R}^D, D \in \{2,3\}$ ja ajasta $t \in \mathbb{R}$ riippuvat vektorikentät $\boldsymbol{E}, \boldsymbol{D}, \boldsymbol{H}$ ja $\boldsymbol{B}$. Kenttiä $\boldsymbol{E}$ ja $\boldsymbol{H}$ kutsutaan sähkökentäksi ja magneettikentäksi. Kenttiä $\boldsymbol{D}$ ja $\boldsymbol{B}$ kutsutaan sähkön siirtymäksi ja magneettiseksi induktioksi. Maxwellin yhtälöt kuvaavat miten varaustiheys ρ ja virtaustiheys $\boldsymbol{J}$ ovat suhteessa näihin vektorikenttiin:

$$\frac{\partial \boldsymbol{D}}{\partial t} - \operatorname{curl} \boldsymbol{H} = -\boldsymbol{J}\,, \tag{1}$$

$$\frac{\partial \boldsymbol{B}}{\partial t} + \operatorname{curl} \boldsymbol{E} = 0\,, \tag{2}$$

$$\operatorname{div} \boldsymbol{D} = \rho\,, \tag{3}$$

$$\operatorname{div} \boldsymbol{B} = 0\,, \tag{4}$$

kaikille $(t, \boldsymbol{x}) \in (0,T) \times \Omega$. Tässä Ω on rajoitettu ja yhtenäinen Lipschitz-alue avaruudessa $\mathbb{R}^D$. Yhtälö (1) on Maxwellin muokkaama Ampéren laki. Yhtälöä (2) kutsutaan Faradayn laiksi ja se kertoo miten muuttuva magneettinen kenttä vaikuttaa sähkökenttään. Yhtälö (3) on Gaussin laki ja se kuvaa varaustiheyden vaikutuksen sähkön siirtymään. Yhtälö (4) tarkoittaa sitä, että magneettinen induktio $\boldsymbol{B}$ on solenoidi, eli lähteetön.

Yhtälöiden (1)-(4) lisäksi on myös asetettava reunaehdot vektorikentille $\boldsymbol{E} \in H(\operatorname{curl}, \Omega)$ ja $\boldsymbol{B} \in H(\operatorname{div}, \Omega)$, esim.

$$\boldsymbol{E} \times \boldsymbol{n} = 0 \qquad \text{ja} \qquad \boldsymbol{B} \cdot \boldsymbol{n} = 0\,, \tag{5}$$

missä $\boldsymbol{n}$ on alueen Ω reunan ulkonormaali. Koska kyseessä on aikariippuvainen ongelma, on asetettava myös alkuehdot ajanhetkellä $t = 0$:

$$\boldsymbol{E}(0, \boldsymbol{x}) = \boldsymbol{E}_0(\boldsymbol{x}) \qquad \text{ja} \qquad \boldsymbol{B}(0, \boldsymbol{x}) = \boldsymbol{B}_0(\boldsymbol{x})\,, \tag{6}$$

missä $\boldsymbol{B}_0$ toteuttaa ehdon

$$\operatorname{div} \boldsymbol{B}_0 = 0\,, \quad \boldsymbol{x} \in \Omega\,. \tag{7}$$

Seuraavaksi määrittelemme konstituutiolait, jotka yhdistävät vektorikentät $\boldsymbol{E}$ ja $\boldsymbol{H}$ kenttiin $\boldsymbol{D}$ ja $\boldsymbol{B}$. Riippuen eri materiaalien ominaisuuksista voidaan erotella kolme eri tapausta konstituutiolaeille: tyhjiö, epähomogeeniset isotrooppiset materiaalit ja epähomogeeniset anisotrooppiset materiaalit. Tässä tutkielmassa oletamme materiaalien olevan epähomogeenisia ja isotrooppisia. Jos materiaalin ominaisuudet eivät riipu magneettikentän suunnasta ja materiaali on lineaarista, konstituutiolait ovat

$$\boldsymbol{D} = \epsilon \boldsymbol{E} \qquad \text{ja} \qquad \boldsymbol{B} = \mu \boldsymbol{H}\,, \tag{8}$$

missä $\epsilon(\boldsymbol{x}) > 0$ on sähkökentän permittiivisyys ja $\mu(\boldsymbol{x}) > 0$ on magneettinen permeabiliteetti. Sekä ϵ että μ voivat olla positiivisia vakioita tai positiivisia rajoitettuja funktioita.

Kirjallisuudessa ja tieteellisissä julkaisuissa Maxwellin yhtälöt ilmaistaan usein vain Ampéren ja Faradayn lailla (1) ja (2). Osoittautuu että divergenssiehdot (3) ja (4) ovat seurauksia näille kahdelle laille. Ottamalla divergenssi Ampéren laista (1) ja integroimalla puolittain alueen Ω yli, saamme

$$\frac{\partial}{\partial t}\int_\Omega \operatorname{div} \boldsymbol{D}\, d\boldsymbol{x} = -\int_\Omega \operatorname{div} \boldsymbol{J}\, d\boldsymbol{x}\,,$$

sillä divergenssi operaattorista curl on nolla jokaiselle vektorikentälle. Käyttämällä Gaussin divergenssilausetta 2.1 yhtälön oikeaan puoleen saamme

$$\frac{\partial}{\partial t}\int_\Omega \operatorname{div} \boldsymbol{D}\, d\boldsymbol{x} = -\int_{\partial\Omega} \boldsymbol{J}\cdot\boldsymbol{n}\, d\boldsymbol{s}\,.$$

Toisaalta, tiedämme, että jokaiselle alueelle $\omega \subset \Omega$ pätee yhtälö

$$\frac{\partial}{\partial t}\int_\omega \rho\, d\boldsymbol{x} = -\int_{\partial\omega} \boldsymbol{J}\cdot\boldsymbol{n}\, ds\,,$$

eli voimme päätellä että $\operatorname{div} \boldsymbol{D} = \rho$. Vastaavasti kun Faradayn laista (2) otetaan divergenssi, saamme

$$\frac{\partial}{\partial t}\operatorname{div} \boldsymbol{B} = 0\,,$$

eli kentän $\boldsymbol{B}$ divergenssi on vakio. Tästä yhtälöstä ja alkuehdosta (7) seuraa divergenssiehto (4). Yhtälöiden (1)-(4) sijaan voimme siis tarkastella yhtälöitä

$$\epsilon\frac{\partial \boldsymbol{E}}{\partial t} - \operatorname{curl}\left(\mu^{-1}\boldsymbol{B}\right) = -\boldsymbol{J}\,, \tag{9}$$

$$\frac{\partial \boldsymbol{B}}{\partial t} + \operatorname{curl} \boldsymbol{E} = 0\,, \tag{10}$$

reunaehtojen (5) ja alkuehtojen (6) kanssa. Näissä yhtälöissä $\boldsymbol{D}$ ja $\boldsymbol{H}$ on korvattu konstituutiolakien (8) mukaisesti kentillä $\boldsymbol{E}$ ja $\boldsymbol{B}$.

Aikaderivaatta voidaan diskretoida usealla eri tavalla (katso [23, 42]). Tässä käytämme ns. käänteistä Euleria. Käänteinen Euler on *implisiittinen* menetelmä, ja on suosittu sen takia, että sitä käyttämällä aika-askele voidaan asettaa suhteellisen isoksi. Käyttämällä tätä menetelmää yhtälöihin (9)-(10) saamme

$$\begin{aligned} \frac{\epsilon}{\triangle t}\left(\boldsymbol{E}_n - \boldsymbol{E}_{n-1}\right) - \operatorname{curl}\left(\mu^{-1}\boldsymbol{B}_n\right) &= -\boldsymbol{J}\,, \\ \frac{1}{\triangle t}\left(\boldsymbol{B}_n - \boldsymbol{B}_{n-1}\right) + \operatorname{curl}\boldsymbol{E}_n &= 0\,, \qquad n = 1, \ldots, \frac{T}{\triangle t}\,, \end{aligned}$$

missä $\triangle t$ on aika-askeleen kesto. Eliminoimalla $\boldsymbol{B}_n$ ja siirtämällä $\boldsymbol{E}_{n-1}$ ja $\boldsymbol{B}_{n-1}$ oikealle puolelle saamme

$$\operatorname{curl}\left(\mu^{-1}\operatorname{curl}\boldsymbol{E}_n\right) + \frac{\epsilon}{(\triangle t)^2}\boldsymbol{E}_n = \frac{1}{\triangle t}\left(-\boldsymbol{J} + \frac{\epsilon}{\triangle t}\boldsymbol{E}_{n-1} + \operatorname{curl}\left(\mu^{-1}\boldsymbol{B}_{n-1}\right)\right)\,.$$

Merkitsemällä oikeaa puolta $\boldsymbol{f} \in L_2(\Omega, \mathbb{R}^D)$ ja asettamalla $\kappa = \epsilon(\triangle t)^{-2}$ saamme

$$\begin{aligned} \operatorname{curl}\left(\mu^{-1}\operatorname{curl}\boldsymbol{E}_n\right) + \kappa\boldsymbol{E}_n &= \boldsymbol{f}\,, & \boldsymbol{x} &\in \Omega\,, \\ \boldsymbol{E}_n \times \boldsymbol{n} &= 0\,, & \boldsymbol{x} &\in \partial\Omega\,, \end{aligned}$$

yhdessä alkuehtojen (6) kanssa. Ylläoleva ongelma mallintaa sähkökentän käyttäytymistä suljetussa ja rajoitetussa alueessa, jonka seinät johtavat sähköä ilman resistanssia.

Tämän tyyppinen ongelma tulee esille monessa erilaisessa asetelmassa. Jos funktio $\kappa \equiv 0$, puhutaan *magnetostatiikasta*. Jos $\kappa \in \mathbb{R}$ on positiivinen, yhtälöä kutsutaan ns. *pyörrevirtausyhtälöksi*. Aikaharmonisessa tapauksessa κ on kompleksinen. Tällöin Fourier-muunnokseen liittyvä taajuus määrää yhtälön fysikaalisen merkityksen. Jos taajuus on tarpeeksi matala, yhtälöllä voidaan mallintaa esim. muuntajia, sähkömoottoreita ja dynamoja. Korkeilla taajuuksilla yhtälö mallintaa esim. antenneja, laser-resonaattoreita ja optisia kuituja.

B Lähdekoodit

Kaikki tähän tutkielmaan liittyvä ohjelmakoodi on tehty ohjelmointialustalla Matlab [24]. Matlabin lisäosaa PDE-toolbox [25] käytimme elementtiverkkojen käsittelemiseen ja visualisointiin. PDE-toolbox sisältää myös valmiita elementtiratkaisijoita, mutta päätimme tehdä tarvitsemamme ratkaisijat itse. Päädyimme tähän sen takia, koska tekemällä ratkaisijan itse kykenemme myös hallitsemaan sitä juuri niin kuin haluamme. Valmiit ratkaisijat piilottavat osan toimintaperiaatteistansa, ja tämän tuoma epävarmuus ei sopinut meille.

B.1 Malliongelman ratkaiseminen (solver_simple.m)

Tiedostossa `solver_simple.m` kuvataan elementtiratkaisija, joka ratkaisee tämän tutkielman malliongelman. Ratkaisija olettaa, että malliongelman kuvaava data on säilötty yhteen kansioon. Kansion nimi annetaan parametrissa `testfolder`. Käyttäjä määrittää myös parametrin `h`, joka kertoo generoitavan elementtiverkon koon. Ratkaisu piirretään käyttäjän näkyville jos parametrille `draw` on annettu arvoksi 1.

```
function solver_simple( testfolder, edgepoints, draw )

% SOLVER_SIMPLE
%   This function solves the model problem
%
% SYNTAX:   solver_simple( testfolder, h, draw )
%
% IN:       testfolder      the folder in which the problem data is
%
%           h               if the mesh is a rectangle, this parameter
%                           is the number of edgepoints in the mesh.
%                           Otherwise it means the maximum edge size
%                           of an element
%
%           draw            1=draw, 0=do not draw the solution
```

Ensin siirrytään annettuun kansioon ja haetaan ongelman kuvaava data. Funktiot μ, $\boldsymbol{f}$ ja κ ovat matlab-funktioita. Tämän jälkeen luodaan matlabin geometriakuvaustiedostosta `geom.m` elementtiverkko.

```
cd(testfolder)

% the problem data
myy = @my;
ff = @f;
kappa = @kappaa;

% create mesh. regular if the mesh is an rectangle, otherwise non-regular
try
    [p,e,t] = poimesh('geom',h,h);
catch
```

```
    [p,e,t] = initmesh('geom','Hmax',h);
end

cd ..
```

Koska Nédélec-FEM:ssä vapausasteet ovat reunaintegraaleja, täytyy elementtiverkko kuvata reunojen avulla. Funktio `edgedata` palauttaa kolme taulukkoa `elems`, `all_edges` ja `b_edges`. Taulukko `all_edges` ilmoittaa elementtiverkon jokaisen reunan solmupisteidensä avulla. Koska vapausasteemme ovat reunaintegraaleja, taulukko `all_edges` ilmoittaa reunan järjestysnumeron lisäksi myös vapausasteen järjestysnumeron. Taulukossa `elems` on jokainen kolmio kuvattu reunojensa suhteen. Reunat ovat tietenkin ilmoitettu vastapäiväisessä järjestyksessä. Taulukko `b_edges` sisältää niiden reunojen indeksit taulukosta `all_edges`, jotka kuuluvat alueen Ω reunalle $\partial\Omega$.

```
% get additional mesh data
[elems,all_edges,b_edges] = edgedata(e,t);

% gather the mesh data to a single structure
meshdata.p = p;
meshdata.e = e;
meshdata.t = t;
meshdata.elems = elems;
meshdata.all_edges = all_edges;
meshdata.b_edges = b_edges;
```

Lopuksi ratkaistaan ongelma rutiinilla `solver`. Tässä numeerista ratkaisua $\boldsymbol{v}$ on merkitty muuttujalla `u_h`. Rutiinin `solver` koodi löytyy liitteestä B.2.

```
u_h = solver(meshdata,myy,kappaa,ff);

% visualize the solution
if ( draw )
   [p_plot,v_plot]=visualize(meshdata,u_h,0,0,0,0,0);
   quiver(p_plot(1,:),p_plot(2,:),v_plot(1,:),v_plot(2,:));
   axis equal; axis tight;
end
```

B.2 Matriisien kokoaminen ja reunaehdon asettaminen (solver.m)

Tässä rutiinissa lasketaan yhtälön (2.10) globaalit matriisit $\mathbf{S}$ ja $\mathbf{M}$ sekä globaali voimavektori $\mathbf{b}$, asetetaan homogeeninen Dirichlet'n reunaehto ja ratkaistaan saatu yhtälöryhmä.

```
function [u_h,S,M,b] = solver(meshdata,my,kappa,f)

% SOLVER Nedelec-FEM solver
%   Solve the second order Maxwell system
%
%       curl( my^(-1) * curl(u) ) + kappa^2 * u = f
%       u x n = 0
%
```

```
% IN:   meshdata.
%        p,e,t                    the mesh data
%        elems,all_edges,b_edges  the additional mesh data returned by
%                                 the function 'edgedata'
%
%       my    = my(x,y) > 0 is the magnetic permeability
%
%       kappa = kappa(x,y) > 0 is the electric permittivity
%
%       f is the source function
%
% OUT:  u_h     the numerical solution's coefficients
%       S       stiffness matrix
%       M       mass matrix
%       b       load vector

dof = size(meshdata.all_edges,2);

disp(' ');
disp( '    ------------------------------------------------------');
disp( '    SOLVER (num solution u_h):');
disp(['    Degree of Nedelec-element:        ',num2str(1)]);
disp(['    Total degrees of freedom:         ',num2str(dof)]);
```

Rutiinilla `assemble_matrices` kootaan globaalit matriisit $\mathbf{S}$ ja $\mathbf{M}$ sekä globaali voimavektori $\mathbf{b}$. Rutiinilla `set_homogenous_dirichlet` asetetaan homogeeninen Dirichlet'n reunaehto. Molempien rutiinien ohjelmakoodi esitellään myöhemmin tässä liitteessä.

```
% calculate the matrices
%------------------------------------------------------------------------
tic;
[S,M,b] = assemble_matrices(meshdata,my,kappa,f);
time1 = toc;
disp(['    Assembling matrices:              [',num2str(time1),']']);

% set homogenous dirichlet bc
%------------------------------------------------------------------------
tic;
[S,M,b] = set_homogenous_dirichlet(meshdata,S,M,b);
time2 = toc;
disp(['    Setting homogenous dirichlet bc:  [',num2str(time2),']']);
```

Lopuksi ratkaistaan saatu yhtälöryhmä. Numeerista ratkaisua $\boldsymbol{v}$ on merkitty muuttujalla `u_h`.

```
% calculate the solution coefficients
%------------------------------------------------------------------------
tic;
u_h = (S+M)\b;
time3 = toc;
disp(['    Solving matrix system:            [',num2str(time3),']']);

disp(['    Done! Total time used: [',num2str(time1+time2+time3),']']);
disp( '    ------------------------------------------------------');
disp(' ');

end
```

Funktiossa `assemble_matrices` suoritetaan laskennallisesti vaativin prosessi: globaalien matriisien kokoaminen.

```
%########################################################################
%########################################################################
%
% ASSEMBLING THE FEM MATRICES
%
%   The variational formulation is
%
%       int_omega( my^(-1) * curl(u) * curl(w) + k * u * w )dx =
%       = int_omega( f * w ) dx
%
%########################################################################
%########################################################################

function [STIFF,MASS,LOAD] = assemble_matrices( meshdata, my, kappa, f )

%extract the mesh data
%-----------------------------------------------------------------------
p         = meshdata.p;
e         = meshdata.e;
t         = meshdata.t;
elems     = meshdata.elems;
all_edges = meshdata.all_edges;

nedges = size(all_edges,2);         %number of edges
```

Tässä alustetaan globaalit matriisit. Ohjelmakoodissa yhtälön (2.10) jäykkyysmatriisia $\mathbf{S}$, massamatriisia $\mathbf{M}$ ja voimavektoria $\mathbf{b}$ on merkitty vastaavasti muuttujilla `STIFF`, `MASS` ja `LOAD`.

```
%initialise global matrices
%-----------------------------------------------------------------------
LOAD    = zeros(nedges,1);
STIFF   = spdiags(LOAD,0,nedges,nedges);
MASS    = STIFF;
```

Kuten yleensä elementtimenetelmässä, integrointi suoritetaan muuttujanvaihdon jälkeen referenssielementillä. Integrointi suoritetaan numeerisesti käyttäen integrointikvadratuuria, joka sisältää integrointipisteet ja painokertoimet.

```
%get the integration points and weighs on the ref element
%-----------------------------------------------------------------------
qr = quad_rule();       %the number of quadrature rule
[ip,w] = inttri(qr);
nip = size(ip,2);       %number of integration points
```

Nyt voimme hakea referenssielementin kantafunktioiden (3.5) ja niiden curl:ien arvot integrointipisteissä.

```
%get the ref basis func and curl values on the integration points
%-----------------------------------------------------------------------
[val,cval] = Nedelec1_basis(ip);
```

Seuraava for-silmukka on hyvin tyypillinen osa elementtiratkaisijaa. Siinä käydään jokainen verkon elementti yksitellen lävitse, kootaan elementtiin liittyvät lokaalit matriisit ja lisätään niiden kontribuutio globaaleihin matriiseihin.

```
%cycle through all triangular elements and compute the local
%STIFFNESS, MASS matrices and the local LOAD vector. Then add the local
%contributions to the global matrices and vector.
%--------------------------------------------------------------------

unit = ones(1, nip);     %help variable

ne = size(t,2);          %number of elements

for i=1:ne
    %reset the local matrices and vector
    L_STIFF = zeros(3,3);
    L_MASS  = zeros(3,3);
    L_LOAD  = zeros(3,1);
```

Seuraavat rivit muodostavat tarvittavan informaation vuorossa olevaan elementtiin liittyvän affiinikuvauksen $\boldsymbol{F}_K$ (3.1) muodostamiseen, jotta saamme sen arvot integrointipisteissä. Elementtiin liittyvän informaation antaa rutiini `element_info`, ja elementin reunoihin liittyvän merkki-informaation (tangentin suunnistuksen) antaa rutiini `element_signs`.

```
    %calculate element information
    [coord,B_K,b_K,B_K_inv,B_K_det] = element_info(meshdata,i);

    %the transpose of the inverse matrix of B_K
    B_K_invT = transpose(B_K_inv);

    %calculate F_K( integration points ) and get a 2 x nip matrix
    F_K_ip = B_K * ip + [ b_K(1)*unit ; b_K(2)*unit ];

    %element's local signs
    loc_signs = element_signs(meshdata,i);
```

Seuraavat kaksi for-silmukka kokoavat lokaalin jäykkyysmatriisin `L_STIFF` ja massamatriisin `L_MASS` sekä lokaalin voimavektorin `L_LOAD`. Numeerinen integrointi on suoritettu aikaavievän for-silmukan sijasta matriisioperaatiolla. Koska lokaalit matriisit ovat symmetrisiä, ohjelmakoodia voisi optimoida siten, että lasketaan vain diagonaalin ja diagonaalin yläpuoliset (tai alapuoliset) alkiot.

Lokaalin jäykkyysmatriisin alkiot on integroitu kaavan (3.7) mukaisesti referenssielementillä. Samoin lokaalin massamatriisin ja voimavektorin alkiot integroidaan Piola-muunnoksen (3.6) avulla referenssielementillä. Huomaa myös, että tangentin suunnistus on otettu huomioon merkkimatriisia `loc_signs` käyttäen.

```
    %calculate the local STIFFNESS and MASS matrix
    %---------------------------------------------

    myval = my(F_K_ip);
    kappaval = kappa(F_K_ip);

    for m=1:3
        for k=1:3
            L_STIFF(m,k) = ...
               L_STIFF(m,k) + ...
               sum( w' .* 1/B_K_det .* ...
                    myval'.^(-1) .* ...
```

```
                    loc_signs(m) .* cval(1,:,m) .* ...
                    loc_signs(k) .* cval(1,:,k) ...
                );

            L_MASS(m,k) = ...
                L_MASS(m,k) + sum( w' .* B_K_det .* kappaval' .* ...
                sum( loc_signs(m) .* ( B_K_invT * val(:,:,m) ) ...
                    .* ...
                    loc_signs(k) .* ( B_K_invT * val(:,:,k) ) ...
                   ) ...
                );
        end
    end

    %calculate the local LOAD vector
    %------------------------------

    %get the load function values at transformed integration points
    fval = f(F_K_ip);

    for k=1:3
        L_LOAD(k) = L_LOAD(k) + sum( w' .* B_K_det .* ...
                    sum ( fval ...
                          .* ...
                          loc_signs(k) .* ( B_K_invT * val(:,:,k) ) ...
                        ) ...
                    );
    end
```

Lopuksi lokaalien matriisien kontribuutiot lisätään globaaleihin matriiseihin oikeiden vapausasteiden kohdalle.

```
    %add the local matrice's contributions to the global matrices
    %------------------------------------------------------------
    loc_elems = elems(:,i);

    STIFF( loc_elems , loc_elems ) = ...
       STIFF( loc_elems , loc_elems ) + L_STIFF;

    MASS( loc_elems , loc_elems )  = ...
       MASS( loc_elems , loc_elems ) + L_MASS;

    LOAD( loc_elems ) = LOAD( loc_elems ) + L_LOAD;
end

end
```

Homogeeninen Dirichlet'n reunaehto on helpointa asettaa vasta sen jälkeen kun yhtälön (2.10) globaalit matriisit $\mathbf{S}$, $\mathbf{M}$ ja globaali vektori $\mathbf{b}$ on laskettu. Reunaehto asetetaan muokkaamalla suoraan globaaleita matriiseja ja vektoria.

```
%########################################################################
%########################################################################
%
% SETTING THE HOMOGENOUS BOUNDARY CONDITION
%
%   Set homogenous dirichlet boundary condition 'u x n = 0' to
%   WHOLE boundary.  The boundary is then called a perfectly
%   conducting wall.
%
```

```
%   If we have homogenous Dirichlet condition 'u x n = 0', where u is the
%   sought FEM solution, and n is the outer normal, then the dofs related
%   to boundaries are zero. This means that the basis functions on the
%   boundaries are zero, as well as their curls. This means that both the
%   stiffness matrix S, mass matrix M and the load vector b has to be
%   changed.
%
%   The algorithm is as follows: Find those dofs that are on the boundary
%   and store the global indices of these dofs. Then, according to these
%   indices set those columns and rows of S and M to zero. Also, set the
%   diagonal of each index x index pair to one. For the load bector, just
%   set the data on these indices to zero.
%
%   For example, if we have 5 x 5 matrix and the dofs to be set to 0 are
%   indices 3 and 5, we would do:
%
%   [ 1 2 3 4 5 ;       [ 1 2 0 4 0 ;
%     1 2 3 4 5 ;         1 2 0 4 0 ;
%     1 2 3 4 5 ;   -->   0 0 1 0 0 ;
%     1 2 3 4 5 ;         1 2 0 4 0 ;
%     1 2 3 4 5 ];        0 0 0 0 1 ];
%
%   NOTE: The above works ONLY for homogenous case
%
%########################################################################
%########################################################################
```

Koska jokaiseen reunaan liittyy yksi vapausaste, on matriiseissa $\mathbf{S}$ ja $\mathbf{M}$ vapausasteiden määrä sarakkeita ja rivejä. Vastaavasti vektorissa $\mathbf{b}$ on vapausasteiden määrän verran alkioita. Esim. vapausastetta n:o i vastaa matriisin $\mathbf{S}$ i:s sarake ja i:s rivi.

Taulukko `b_edges` sisältää niiden reunojen indeksit taulukosta `all_edges`, jotka kuuluvat alueen Ω reunalle $\partial\Omega$. Taulukko `b_edges` kertoo siten suoraan ne vapausasteet, jotka liittyvät reunaan $\partial\Omega$. Homogeeninen Dirichlet'n reunaehto asetetaan siten, että jokaista taulukon `b_edges` reunaa kohden asetetaan vastaavat $\mathbf{S}$:n ja $\mathbf{M}$:n sarakkeet ja rivit nolliksi lukuunottamatta sitä alkiota, joka osuu vapausasteeseen liittyvän sarakkeen ja rivin diagonaalille. Koska kyseessä on homogeeninen reunaehto, tämä diagonaalialkio voi olla mikä tahansa nollasta eroava luku. Vektorin $\mathbf{b}$ tapauksessa riittää, että asetetaan vastaavat vapausasteet nolliksi.

```
function [ S, M, b ] = set_homogenous_dirichlet( meshdata, S, M, b )

% extract needed mesh data
%-------------------------------------------------------------------------
b_edges   = meshdata.b_edges;

%do the elimination
%-------------------------------------------------------------------------

S(b_edges,:) = 0;
M(b_edges,:) = 0;

S(:,b_edges) = 0;
M(:,b_edges) = 0;

for i=1:size(b_edges,2)
   index = b_edges(i);
   S(index,index) = 1/2;  %can set this to any number except zero.
   M(index,index) = 1/2;  %this is because the boundary condition
                          %is homogenous.
```

```
end

b(b_edges) = 0;

end
```

B.3 Virherajojen laskeminen (solver_global.m)

Tässä tiedostossa lasketaan virherajat annetulle ongelmalle. Majorantin $M_\oplus$ arvo lasketaan myös samalla verkolla kuin numeerinen ratkaisu. Sen jälkeen majorantin $M_\oplus$ ja minorantin $M_\ominus$ arvot lasketaan tihennetyissä verkoissa.

```
function solver_global( testfolder, exactinc, h, refiterno, yeorder )

% SOLVER_GLOBAL
%   This solver is an "normal" solver that uses global minimisation
%   method to calculate the free parameter in the majorant.
%
% SYNTAX:   solver_global( testfolder, exactinc, h, refiterno, yeorder )
%
% IN:       testfolder      the folder in which the problem data is
%
%           exactinc        0 = exact solution not included
%                           1 = exact solution included as files 'exact'
%                           and 'exact_curl' / reference solution as
%                           files 'referencemesh' and 'referencesolution'
%
%           h               if the mesh is a rectangle, this parameter
%                           means the number of edgepoints in the mesh.
%                           Otherwise it means the maximum edge size of
%                           an element
%
%           refiterno       how many refinement iterations to perform
%
%           yeorder         the order of elements used to approximate the
%                           free parameter y in the majorant
```

Ensin kootaan tarvittava informaatio ratkaistavasta ongelmasta

```
%get the problem data for this particular test case
%------------------------------------------------------------------------

cd(testfolder)

myy = @my;
ff = @f;
kappaa = @kappa;
```

Jotta majorantille voitaisiin laskea tehokkuusindeksin (6.1) arvo, täytyy meidän kyetä laskemaan tarkka virhe $\| \boldsymbol{u} - \boldsymbol{v} \|$. Tätä varten tarvitsemme tietenkin tarkan ratkaisun $\boldsymbol{u}$. Jos ongelmalle tunnetaan analyyttinen ratkaisu, se ja sen curl-funktio on tallennettu matlab-funktioihin `exact.m` ja `exact_curl.m`. Jos taas malliongelmalle ei tunneta analyyttistä ratkaisua, on sille laskettu referenssiratkaisu todella tiheällä verkolla. Tässä tapauksessa tarkka

ratkaisu ja siihen liittyvä elementtiverkko on tallennettu tiedostoihin `referencesolution.m` ja `referencemesh.m`.

```
exacttype = 0;
if ( exactinc )
    try
        u = @exact;
        u_curl = @exact_curl;
        u([0;0]);
        exacttype = 1;
    catch
        load referencemesh;
        load referencesolution;
        meshdata_ref.p = p;
        meshdata_ref.e = e;
        meshdata_ref.t = t;
        meshdata_ref.elems = elems;
        meshdata_ref.all_edges = all_edges;
        meshdata_ref.b_edges = b_edges;
        exacttype = 2;
    end
end

%gather mesh data
%-------------------------------------------------------------------------
% create mesh. regular if the mesh is an rectangle, otherwise non-regular
[p,e,t] = initmesh('geom','Hmax',h);

cd ..

[elems,all_edges,b_edges] = edgedata(e,t);  %get additional mesh data

meshdata.p = p;
meshdata.e = e;
meshdata.t = t;
meshdata.elems = elems;
meshdata.all_edges = all_edges;
meshdata.b_edges = b_edges;

disp(' ')
disp(['Using quadrature rule n:o ',num2str(quad_rule())]);
disp(' ')
disp('Initial solution')
disp('-----------------------------------------------------------------');
disp(' ')
disp(['    NOF nodes    ',num2str(size(meshdata.p,2))]);
disp(['    NOF elements ',num2str(size(meshdata.t,2))]);
disp(['    NOF edges    ',num2str(size(meshdata.all_edges,2))]);
```

Seuraavaksi lasketaan ongelman numeerinen ratkaisu, jolle virherajat lasketaan. Numeerista ratkaisua $\boldsymbol{v}$ on merkitty muuttujalla `u_h`. Ratkaisun energiafunktionaalin arvo (2.6) on laskettu muuttujaan `Ju`. Tarvitsemme tätä arvoa kun laskemme minorantin arvoja.

```
%calculate the problem solution
%-------------------------------------------------------------------------
[u_h,S,M,b] = solver(meshdata,myy,kappaa,ff);

u_h_Enorm = u_h'*(S+M)*u_h;
Ju = (1/2) * u_h_Enorm - u_h'*b;
disp(['    Energy:       ', num2str(Ju)]);
```

Seuraavaksi lasketaan numeerisen ratkaisun virhe $\boldsymbol{u} - \boldsymbol{v}$ eri normeissa. Tehokkuusindeksiä varten lasketaan virhe myös energianormissa. Energianormissa laskettu virhe on tallennettu muuttujaan `Eerr_tot`. Jos tarkka ratkaisu on annettu referenssiratkaisuna, ei virhettä lasketa muissa normeissa lainkaan. Referenssiratkaisulle virheen normien laskeminen on hyvin hidasta.

```
if ( exacttype == 1 )

    %calculate L2 and H(curl) error
    [L2err,HCerr,u_L2norm,u_HCnorm] = ...
       L2_Hcurl_error(meshdata, u_h, u, u_curl);

    %calculate the energy error
    [Eerr,u_Enorm] = energy_error(meshdata, myy, kappaa, u_h, u, u_curl);
    Eerr_tot = sum(sum(Eerr));

    disp(' ')
    disp( '    || u - u_h ||');
    disp( '    -------------');
    disp(['    H(curl) seminorm: ', num2str( sqrt(HCerr)           )]);
    disp(['    H(curl) norm:     ', num2str( sqrt(HCerr + L2err)   )]);
    disp(['    L2 norm:          ', num2str( sqrt(L2err)           )]);
    disp(['    Energy norm:      ', num2str( sqrt(Eerr_tot)        )]);
    disp(' ')
    disp( '    || u - u_h || / || u ||')
    disp( '    ----------------------')
    disp(['    H(curl) seminorm: ', num2str( sqrt( HCerr/u_HCnorm   ))]);
    disp(['    H(curl) norm:     ', num2str( sqrt( (HCerr + L2err) ...
                                      / (u_HCnorm + u_L2norm) ))]);
    disp(['    L2 norm:          ', num2str( sqrt( L2err/u_L2norm   ))]);
    disp(['    Energy norm:      ', num2str( sqrt( Eerr_tot/u_Enorm ))]);
    disp(' ')
    disp( '    || u - u_h || / || u_h ||')
    disp( '    -------------------------')
    disp(['    Energy norm:  ', num2str( sqrt( Eerr_tot/u_h_Enorm ))]);
    disp(' ')

elseif ( exacttype == 2)

    %calculate the energy error
    [Eerr,u_Enorm] = ...
       energy_error(meshdata, myy, kappaa, u_h, meshdata_ref, u_ref);
    Eerr_tot = sum(sum(Eerr));

    disp( '    || u - u_h ||');
    disp( '    -------------');
    disp(['    Energy norm:      ', num2str( sqrt(Eerr_tot)       )]);
    disp(' ')
    disp( '    || u - u_h || / || u ||')
    disp( '    ----------------------')
    disp(['    Energy norm:      ', num2str( sqrt( Eerr_tot/u_Enorm ))]);
    disp(' ')
    disp( '    || u - u_h || / || u_h ||')
    disp( '    -------------------------')
    disp(['    Energy norm:  ', num2str( sqrt( Eerr_tot/u_h_Enorm ))]);
    disp(' ')

end
```

Tässä majorantin $M_\oplus$ (5.3) arvo lasketaan samalla verkolla kuin missä numeerinen ratkaisu on laskettu. Majoranttiin liittyvät funktiot ovat kansiossa `majorant`. Ensin lasketaan vapaa parametri y rutiinilla `majorant/solver.m` (katso liite B.4). Ohjelmakoodissa tätä pa-

rametria on merkitty muuttujalla `y_h`. Tämän jälkeen lasketaan majorantin arvo rutiinilla `majorant/majorant.m` (katso liite B.6). Majorantin kaksi termiä on annettu tässä erikseen (ja elementtikohtaisesti) vektoreihin `left` ja `right`.

```
%calculate Majorant for the first mesh
%------------------------------------------------------------------------

disp('  MAJORANT');

cd majorant

% calculate the free parameter
y_h = solver(meshdata,meshdata,myy,kappaa,ff,u_h,yeorder);

% calculate the upper bound
[left,right] = majorant(meshdata,meshdata,myy,kappaa,ff,u_h,y_h,yeorder);

left = sum(left);
right = sum(right);

maj = left + right
maj2 = sqrt( maj / u_h_Enorm );

disp(['    Upper Bound:       sqrt( (',num2str(left),' + ...
           ',num2str(right),') / ||| u_h |||^2 ) = ' ...
           ,num2str(maj2)]);

if ( exactinc )
    % calculate the effectivity index EFFI
    effi = sqrt(maj/Eerr_tot);
    disp(['    Effectivity index: ',num2str(effi)]);
end

cd ..

disp(' ')
disp( '  SUMMARY');
disp(['    ||| u - u_h ||| / ||| u_h |||  <  ',num2str(maj2)])
disp(' ')
disp(' ')
```

Seuraavaksi aloitetaan tihennysiteraatiot: jokaisessa iteraatiossa verkkoa tihennetään kerran ja lasketaan majorantin ja minorantin arvot tihennetyssä verkossa.

```
% start calculating the upper and lower bounds.
%------------------------------------------------------------------------

p2 = p; e2 = e; t2 = t;

for i=1:refiterno

    % do a refinement
    %--------------------------------------------------------------------

    cd(testfolder)

    %one step refinement on the mesh
    [p2,e2,t2] = refinemesh('geom',p2,e2,t2,'regular');

    cd ..
```

```
%get additional mesh data
[elems2,all_edges2,b_edges2] = edgedata(e2,t2);

%gather the new meshdata
meshdata2.p = p2;
meshdata2.e = e2;
meshdata2.t = t2;
meshdata2.elems = elems2;
meshdata2.all_edges = all_edges2;
meshdata2.b_edges = b_edges2;

disp(['Refinement step ',num2str(i)])
disp('------------------------------------------------------------');
disp(' ')
disp(['    NOF nodes     ',num2str(size(meshdata2.p,2))]);
disp(['    NOF elements  ',num2str(size(meshdata2.t,2))]);
disp(['    NOF edges     ',num2str(size(meshdata2.all_edges,2))]);
disp(' ')
```

Minorantin $M_{\ominus}$ (5.1) arvon laskemiseen tarvitsemme uuden numeerisen ratkaisun tihennetyllä verkolla. Tämän ratkaisun energiafunktionaalin arvo on asetettu muuttujaan `Ju2`.

```
% calculate the lower bound
%--------------------------------------------------------------

disp('  MINORANT');

%calculate the solution in the refined mesh

%calculate the matrices
[u_h2,S2,M2,b2] = solver(meshdata2,myy,kappaa,ff);

Ju2 = (1/2) * u_h2'*(S2+M2)*u_h2 - u_h2'*b2;
disp(['    Energy:       ', num2str(Ju2)]);

%minorant
min = 2 * ( Ju - Ju2 );
min2 = sqrt( ( 2 * ( Ju - Ju2 ) ) / u_h_Enorm );

disp(['    Lower Bound:  sqrt( (',num2str(min),') / ...
           ||| u_h |||^2 ) = ',num2str(min2)]);
disp(' ')
```

Majorantti lasketaan kuten ensimmäiselle verkolle aikaisemmin tässä liitteessä.

```
% calculate the upper bound
%--------------------------------------------------------------------

disp('  MAJORANT');

cd majorant

% calculate the free parameter
y_h = solver(meshdata,meshdata2,myy,kappaa,ff,u_h,yeorder);

% calculate the value of the majorant
[left,right] = ...
   majorant(meshdata,meshdata2,myy,kappaa,ff,u_h,y_h,yeorder);

left = sum(left);
right = sum(right);
```

```
    maj = left + right;
    maj2 = sqrt( maj / u_h_Enorm );

    disp(['    Upper Bound:       sqrt( (',num2str(left),' + ...
               ',num2str(right),') / ||| u_h |||^2 ) = '...
               ,num2str(maj2)]);

    if ( exactinc )
        % calculate the effectivity index EFFI
        effi = sqrt(maj/Eerr_tot);
        disp(['    Effectivity index: ',num2str(effi)]);
    end

    cd ..

    disp(' ')
    disp( '  SUMMARY');
    disp(['    ',num2str(min2),'  <  ||| u - u_h ||| / ||| u_h ||| ...
          <  ',num2str(maj2)])
    disp(' ')
    disp(' ')

end
```

B.4 Parametri y: globaali minimointi (majorant/solver.m)

Minimoimalla majorantti $M_\oplus$ globaalisti muuttujan y suhteen saamme variaatiomuodon (5.13). Tämä ongelma voidaan ratkaista normaalilla H^1-elementtiratkaisijalla. Tämä ratkaisija muistuttaa liitteen B.2 Nédélec-elementtiratkaisijaa. Huomattavin ero on se, että tässä ratkaisijassa ei aseteta lainkaan reunaehtoja. Sen lisäksi käytetty elementti on tietenkin erilainen. Tässä ratkaisijassa on itse asiassa mahdollisuus käyttää lineaaristen elementtien lisäksi myös kvadraattisia elementtejä.

Tässä ratkaisijassa funktiolle annetaan kaksi eri verkkoa, `meshdata1` ja `meshdata2`. Numeerinen ratkaisu on ratkaistu verkolla `meshdata1`, ja vapaa parametri y on tarkoitus ratkaista verkolla `meshdata2`. Vapaan parametrin verkko voi olla sama verkko kuin `meshdata1`, tai siitä verkosta tihennetty verkko.

```
function [y_h,S,M,b1,b2] = ...
         solver( meshdata1, meshdata2, my, kappa, f, u_h, eorder )

% SOLVER H^1-FEM solver
%   Obtain the free parameter y by global minimisation of the majorant.
%
% IN:  meshdata1.
%        p,e,t                    the mesh data (NON-refined)
%        elems,all_edges,b_edges  the additional mesh data returned by
%                                 the function 'edgedata'
%      meshdata2.
%        p,e,t                    the mesh data (same as above or refined)
%        elems,all_edges,b_edges  the additional mesh data returned by
%                                 the function 'edgedata'
%
```

```
%       my    = my(x,y) > 0 is the magnetic permeability
%
%       kappa = kappa(x,y) > 0 is the electric permittivity
%
%       f is the source function
%
%       u_h                     the approximate solution coefficients for
%                               the second order maxwell system
%
%       eorder                  the order of the H1 element
%
% OUT:  y_h     the numerical solution's coefficients
%       S       stiffness matrix
%       M       mass matrix
%       b1      load vector, first part
%       b2      load vector, second part

dof = 0;
switch eorder
    case 1
        dof = size(meshdata2.p,2);
    case 2
        dof = size(meshdata2.p,2) + size(meshdata2.all_edges,2);
end

disp(' ');
disp( '    ------------------------------------------------------');
disp( '    SOLVER (free param y_h):');
disp(['    Degree of H^1-element:            ',num2str(eorder)]);
disp(['    Total degrees of freedom:         ',num2str(dof)]);

% calculate the matrices
%-------------------------------------------------------------------------
tic;
[S,M,b1,b2] = ...
   assemble_matrices(meshdata1,meshdata2,my,kappa,f,u_h,eorder);
time1 = toc;
disp(['    Assembling matrices:              [',num2str(time1),']']);

% solve matrix system
%-------------------------------------------------------------------------
tic;
y_h = (S+M)\(b1+b2);
time2 = toc;
disp(['    Solving matrix system:            [',num2str(time2),']']);

disp(['    Done! Total time used: [',num2str(time1+time2),']']);
disp( '    ------------------------------------------------------');
disp(' ');

end

%#########################################################################
%#########################################################################
%
% ASSEMBLING THE FEM MATRICES
%
%   Our space of test functions is H^1, and a test function is denoted
%   by w. The variational formulation is as follows:
%
%       int_omega( 1/kappa^2 * curl(y) * curl(w)  +  my*y*w )dx =
%     = int_omega( 1/kappa^2 * (f - kappa^2 * u_h) * curl(w)  +
%       curl(u_h) * w )dx
%
%   Solve the variational formulation to get an approximation to the
```

```
%   majorant's free parameter y (obtaining y this way is called global
%   minimisation of the majorant)
%
%   NOTE: The approximate solution u_h is for meshdata1.
%   The free parameter y is calculated for the mesh meshdata2, which
%   can be the same mesh as meshdata1, or a refined mesh.
%
%#########################################################################
%#########################################################################

function [STIFF,MASS,LOAD_LEFT,LOAD_RIGHT] = ...
   assemble_matrices( meshdata1, meshdata2, my, kappa, f, u_h, eorder )

% extract the mesh data
%-------------------------------------------------------------------------

% NON-refined
p1         = meshdata1.p;
e1         = meshdata1.e;
t1         = meshdata1.t;
elems1     = meshdata1.elems;
all_edges1 = meshdata1.all_edges;

% refined
p2         = meshdata2.p;
e2         = meshdata2.e;
t2         = meshdata2.t;
elems2     = meshdata2.elems;
all_edges2 = meshdata2.all_edges;

nnodes = size(p2,2);              %number of nodes in the refined mesh
nedges = size(all_edges2,2);      %number of edges in the refined mesh

%number of global degrees of freedom
%-------------------------------------------------------------------------
switch eorder
    case 1
        ngdof = nnodes;
    case 2
        ngdof = nnodes + nedges;
end

%initialise global matrices
%-------------------------------------------------------------------------
LOAD_LEFT  = zeros(ngdof,1);
LOAD_RIGHT = zeros(ngdof,1);
STIFF   = spdiags(LOAD_LEFT,0,ngdof,ngdof);
MASS    = STIFF;

%get the integration points and weighs on the ref element
%-------------------------------------------------------------------------
qr = quad_rule();       %the number of quadrature rule
[ip,w] = inttri(qr);
nip = size(ip,2);       %number of integration points

%get the ref basis func and gradient values on the integration points
%-------------------------------------------------------------------------
[val,gval,nbasis] = tri_basis(ip,eorder);

%cycle through all triangular elements and compute the local
%STIFFNESS, MASS matrices and the local LOAD vector. Then add the local
%contributions to the global matrices and vector.
%-------------------------------------------------------------------------

unit = ones(1, nip);     % help variable
```

```
R = [0 1 ; -1 0];        % curl(v) = R*gradient(v)

ne = size(t2,2);         % number of elements in the refined mesh

for i=1:ne
    %reset the local matrices and vector
    L_STIFF = zeros(nbasis,nbasis);
    L_MASS  = zeros(nbasis,nbasis);
    L_LOAD_LEFT  = zeros(nbasis,1);
    L_LOAD_RIGHT = zeros(nbasis,1);

    % get element info
    %-----------------

    [coord,B_K,b_K,B_K_inv,B_K_det] = element_info(meshdata2,i);

    %the transpose of inverse of B_K
    B_K_invT = transpose(B_K_inv);

    %calculate F_K( integration points ) and get a 2 x nip matrix
    F_K_ip = B_K * ip + [ b_K(1)*unit ; b_K(2)*unit ];

    % get parent element info (if the given meshes are not the same)
    %---------------------------------------------------------------
    if ( size(t1,2) == size(t2,2) )
        parent = i;
        ip_p = ip;
    else
        parent = tri_parent(meshdata1, meshdata2, i);

        [coord_p,B_K_p,b_K_p,B_K_p_inv,B_K_p_det] = ...
           element_info(meshdata1,parent);

        %transform the integration points F_K_ip to the ref element
        ip_p = B_K_p_inv * F_K_ip - B_K_p_inv * ...
               [ b_K_p(1)*unit ; b_K_p(2)*unit ];
    end

    % calculate the local STIFFNESS and MASS matrix
    %----------------------------------------------

    %get my function values at transformed integration points
    myval = my(F_K_ip);
    kappaval = kappa(F_K_ip);

    for m=1:nbasis
        for k=1:nbasis
            L_STIFF(m,k) = ...
               L_STIFF(m,k) + sum( w' .* B_K_det .* ...
               kappaval'.^(-1) .* ...
               sum( ( R * ( B_K_invT * gval(:,:,m) ) ) ...
                    .* ...
                    ( R * ( B_K_invT * gval(:,:,k) ) ) ...
                  ) ...
               );

            L_MASS(m,k) = ...
               L_MASS(m,k) + sum( w' .* B_K_det .* myval' .* ...
               ( val(:,:,m) .* val(:,:,k) ) );
        end
    end

    % calculate the local LOAD vectors
    %---------------------------------
```

```
    %get the load function values at transformed integration points
    fval = f(F_K_ip);

    %get the approximate soultion values and curl values
    %at the integration points
    cd ..
    [u_h_val,u_h_cval] = solution(meshdata1, u_h, parent, ip_p);
    cd majorant

    for k=1:nbasis
        L_LOAD_LEFT(k) = L_LOAD_LEFT(k) + sum( w' .* B_K_det .* ...
                ( kappaval'.^(-1) .* ...
                  sum( ( fval - [kappaval'; kappaval'] .* u_h_val ) ...
                       .* ...
                       ( R * ( B_K_invT * gval(:,:,k) ) ) ...
                     ) ...
                ) ...
                );

        L_LOAD_RIGHT(k) = L_LOAD_RIGHT(k) + sum( w' .* B_K_det .* ...
                          ( ...
                            u_h_cval .* val(:,:,k) ...
                          ) ...
                          );
    end

    %add the local matrice and vector's contributions to the global ones
    %-----------------------------------------------------------------------
    switch eorder
        case 1
            loc_elems = t2(1:3,i);
        case 2
            loc_elems = [ t2(1:3,i) ; elems2(1:3,i) + nnodes ];
    end

    STIFF( loc_elems , loc_elems ) = ...
       STIFF( loc_elems , loc_elems ) + L_STIFF;

    MASS( loc_elems , loc_elems ) = ...
       MASS( loc_elems , loc_elems ) + L_MASS;

    LOAD_LEFT( loc_elems ) = LOAD_LEFT( loc_elems ) + L_LOAD_LEFT;

    LOAD_RIGHT( loc_elems ) = LOAD_RIGHT( loc_elems ) + L_LOAD_RIGHT;
end

end
```

B.5 Parametri y: silottaminen (majorant/gradient_averaging.m)

Majorantin vapaa parametri y voidaan laskea myös luvussa 5.4.2 esitetyn silottamisen avulla. Ideana on keskiarvoistaa numeerisen ratkaisun curl:in arvoja solmupisteisiin.

```
function y_h = gradient_averaging( meshdata, u_h )

% GRADIENT_AVERAGING
%   Calculate the free parameter y_h by "gradient averaging" technique:
```

```
%   1) For each point find those elements that include this point.
%   2) Calculate the approx solution curl values on each element
%      (the curl of the approx solution is constant in the element,
%       because we use linear nedelec elements)
%   3) Average the curl values by weighting each value by the
%      corresponding element's area, summing the weighted values and
%      dividing by the area of all elements. This value is the value
%      of y_h on that point.

% extract the mesh data
%-------------------------------------------------------------------------
p         = meshdata.p;
t         = meshdata.t;

% calculate y_h by "gradient averaging" technique
%-------------------------------------------------------------------------

np = size(p,2);     % number of nodes in refined mesh

y_h = zeros(np,1);
```

Seuraavassa for-silmukassa käydään läpi jokainen verkon solmu. Funktio `point2triangles` etsii tiettyyn solmuun liittyvät elementit.

```
for i=1:np
    tri = point2triangles(meshdata,i);      % triangle indexes

    nt = size(tri,2);                       % no of triangles
```

Kun käytetään lineaarista Nédélecin elementtiä, on numeerisen ratkaisun curl vakio jokaisessa elementissä. Tämän takia lasketaan numeerisen ratkaisun curl vain yhdessä pisteessä kutakin elementtiä kohden.

```
    % calculate the curl values. when using linear nedelecs elements,
    % the curl of the solution is constant in the whole element.
    % that is why we evaluate the solution only in the point [1/3; 1/3]
    cd ..
    cval = zeros(1,nt);
    for j=1:nt
        [u_h_val,u_h_cval] = ...
           solution(meshdata, u_h, tri(1,j), [1/3; 1/3]);
        cval(1,j) = u_h_cval;
    end
    cd majorant
```

Painotettua keskiarvoistusta varten lasketaan myös elementtien pinta-alat.

```
    % calculate the areas of the triangles
    tri_a = zeros(1,nt);
    for j=1:nt
        coord = p(:,t(1:3,tri(1,j)));    %triangle coordinates
        tri_a(1,j) = 1/2 * det([coord; 1 1 1]);
    end
```

Lopuksi suoritetaan keskiarvoistus kaavan (5.14) mukaisesti. Tässä oletetaan, että funktio $\mu \equiv 1$.

```
    % sum over the curl values, weighed with the areas
    y_h(i) = sum( tri_a .* cval ) / sum(tri_a);
end
```

Kun jokaiselle elementin solmupisteelle saadaan arvo, saadaan arvot muualla elementissä interpoloimalla lineaarisesti. Itse asiassa, tällä tavoin saatu parametri on oleellisesti samanlainen kuin lineaarisen H^1-FEM:in numeeriset ratkaisut.

B.6 Majorantin arvon laskeminen (majorant/majorant.m)

Kun vapaa parametri y on laskettu (globaalilla minimoinnilla tai silottamisella), voidaan laskea majorantin $M_\oplus$ (5.3) arvo. Seuraava rutiini laskee majorantin kaksi eri termiä erikseen (ja elementeittäin) vektoreihin `left` ja `right`.

```
function [left,right] = ...
   majorant( meshdata1, meshdata2, my, kappa, f, u_h, y_h, yeorder )

% MAJORANT
%   Calculate the value of majorant
%
%       M_plus(u_h,y_h) = || 1/kappa * r ||^2  +  || my^(1/2) * d ||^2 ,
%
%   where
%
%       r(u_h,y_h) = f - kappa^2 * u_h - curl(y)               and
%       d(u_h,y_h) = y_h - my^(-1) * curl(u_h)                 .
%
% SYNTAX: majorant( meshdata1,meshdata2,my,kappa,f,u_h,y_h,yeorder )
%
% IN:   meshdata1.              the mesh on which u_h is defined
%        p,e,t                  the mesh data
%        elems,all_edges        the additional mesh data returned by
%                               the function 'edgedata'
%
%       meshdata2.              the mesh on which y_h is defined
%        p,e,t                  the mesh data
%        elems,all_edges        the additional mesh data returned by
%                               the function 'edgedata'
%
%       my = my(x,y) > 0 is the magnetic permeability
%
%       kappa = kappa(x,y) > 0 is the electric permittivity
%
%       f is the source function
%
%       u_h                the approximate solution coefficients
%
%       y_h                the free parameter in the majorant, calculated
%                          with global minimization or by a GA technique
%
%       yeorder            the element order of y_h
%
% OUT:  Let N be the number of elements in the refined mesh
%
%       left   1xN      elementwise values of the left term of majorant
%       right  1xN      elementwise values of the right term of majorant
```

Funktio `integrate` suorittaa integroinnin.

```
%calculate the value for majorant by integrating
[left,right] = integrate(meshdata1,meshdata2,my,kappa,f,u_h,y_h,yeorder);
```

Funktio `integrate` ilmoittaa integraalien arvot verkon `meshdata2` elementtien mukaisesti. Olemme kuitenkin kiinnostuneempia tietää elementtikohtaiset virheet alkuperäisen verkon `meshdata1` elementtien mukaisesti. Funktio `convert` tekee tämän jälkikäsittelyn.

```
%the values of majorant are given elementwise according to the refined
%mesh meshdata2. it is more convenient to give the elemenwise values
%according to the original mesh meshdata1.
[left,right] = convert(meshdata1,meshdata2,left,right);

end
```

Tämä funktio laskee majorantin $M_\oplus$ arvon ja on hyvin tyypillinen integrointirutiini.

```
function [left,right] = ...
   integrate( meshdata1, meshdata2, my, kappa, f, u_h, y_h, yeorder )

% extract the mesh data
%--------------------------------------------------------------------------

% NON-refined
p1         = meshdata1.p;
e1         = meshdata1.e;
t1         = meshdata1.t;
elems1     = meshdata1.elems;
all_edges1 = meshdata1.all_edges;

% refined
p2         = meshdata2.p;
e2         = meshdata2.e;
t2         = meshdata2.t;
elems2     = meshdata2.elems;
all_edges2 = meshdata2.all_edges;
```

Integrointi suoritetaan jälleen referenssielementillä, joten tarvitsemme integrointipisteet ja painot.

```
%get the integration points and weighs on the ref element
%--------------------------------------------------------------------------
qr = quad_rule();       %the number of quadrature rule
[ip,w] = inttri(qr);
nip = size(ip,2);       %number of integration points

unit = ones(1, nip);    %help variable

R = [0 1 ; -1 0];       %curl(v) = R*gradient(v)

ne = size(t2,2);        %number of elements in the refined mesh

% initialise OUT variables
%--------------------------------------------------------------------------
left = zeros(1,ne);
right = left;

for i=1:ne
```

Seuraavaksi lasketaan affiinikuvaus F_K.

```
%calculate the element info
%-------------------------------------------------------------------

[coord,B_K,b_K,B_K_inv,B_K_det] = element_info(meshdata2,i);

%calculate F_K( integration points ) and get a 2 x nip matrix
F_K_ip = B_K * ip + [ b_K(1)*unit ; b_K(2)*unit ];
```

Jos vapaa parametri y on laskettu tiheämmällä verkolla kuin numeerinen ratkaisu, täytyy selvittää tällä hetkellä kyseessä olevan elementin ns. isäelementti verkosta `meshdata1`, ja kuvata integrointipisteet oikein.

```
% get parent element info (only if the given meshes are not the same)
%-------------------------------------------------------------------
if ( size(t1,2) == size(t2,2) )
    parent = i;
    ip_p = ip;
else
    parent = tri_parent(meshdata1, meshdata2, i);

    [coord_p,B_K_p,b_K_p,B_K_p_inv] = element_info(meshdata1,parent);

    %transform the integration points F_K_ip to the reference element
    ip_p = B_K_p_inv * F_K_ip - B_K_p_inv * ...
           [ b_K_p(1)*unit ; b_K_p(2)*unit ];
end
```

Seuraavaksi kootaan tarvittavien funktioiden arvot.

```
%get all needed function values
%-------------------------------------------------------------------

%get my and kappa function values at transformed integration points
myval = my(F_K_ip);
kappaval = kappa(F_K_ip);

%get the load function values at transformed integration points
fval = f(F_K_ip);

%get the approximate soultion values and curl values
%at the integration points
cd ..
[u_h_val,u_h_cval] = solution(meshdata1, u_h, parent, ip_p);
cd majorant

%get the free parameter y values and gradient at the int points
[y_h_val,y_h_gval] = solution(meshdata2, y_h, i, ip, yeorder);
```

Lopuksi integroidaan elementin ylitse.

```
%integrate

r = fval - [kappaval'; kappaval'] .* u_h_val - R * y_h_gval ;
d = y_h_val - ( (myval').^(-1) .* u_h_cval ) ;

left(1,i) = sum( w' .* B_K_det .* ...
                ( kappaval'.^(-1) .* sum( r .* r ) ) ...
```

```
                  );

    right(1,i) = sum( w' .* B_K_det .* ...
                    ( myval' .* ( d.^2 ) ) ...
                  );

end

end
```

Vektoreiden `left` ja `right` sisältämät elementtikohtaiset virheet on ilmoitettu tiheän verkon `meshdata2` elementtien mukaisesti. Funktio `convert` muuttaa vektorit siten, että virheet on ilmoitettu elementtikohtaisesti verkon `meshdata1` mukaisesti.

```
function [L,R] = convert( meshdata1, meshdata2, left, right)

% extract the mesh data
%-------------------------------------------------------------------------

% NON-refined
p1         = meshdata1.p;
e1         = meshdata1.e;
t1         = meshdata1.t;
elems1     = meshdata1.elems;
all_edges1 = meshdata1.all_edges;

% refined
p2         = meshdata2.p;
e2         = meshdata2.e;
t2         = meshdata2.t;
elems2     = meshdata2.elems;
all_edges2 = meshdata2.all_edges;

% if the meshes are the same size, no need to convert.
if size(t1,2) == size(t2,2)
    L = left;
    R = right;
    return;
end

ne = size(t1,2);     % number of elements in the non-refined mesh

% initialise the OUT variables
L = zeros(1,ne);
R = L;

for i=1:ne
    children = tri_child(meshdata1,meshdata2,i);
    L(i) = sum(  left(children) );
    R(i) = sum( right(children) );
end

end
```