Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

MAS

Modelling, Analysis and Simulation

Modelling, Analysis and Simulation

The Eggshell method in a nutshell

M. Nool, D. Lahaye

REPORT MAS-E0514 JULY 2005

CWI is the National Research Institute for Mathematics and Computer Science. It is sponsored by the Netherlands Organization for Scientific Research (NWO).
CWI is a founding member of ERCIM, the European Research Consortium for Informatics and Mathematics.

CWI's research has a theme-oriented structure and is grouped into four clusters. Listed below are the names of the clusters and in parentheses their acronyms.

Probability, Networks and Algorithms (PNA)

Software Engineering (SEN)

Modelling, Analysis and Simulation (MAS)

Information Systems (INS)

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

ISSN 1386-3703

The Eggshell method in a nutshell

ABSTRACT
The Eggshell method was introduced by F. Henrotte as a novel magnetic force computation method. It allows computation of the force by integrating the magnetic stress tensor over a shell surrounding the body of interest. We investigate the numerical properties of this method for current carrying wires, and permanent magnets immersed in two-dimensional stationary magnetic fields, discretized by first and second order isoparametric triangular finite elements. We do so by comparing the accuracy of the method, as a function of the mesh size and element order, with the result of three classical force computation methods: the Lorentz, the Virtual Work and the Maxwell Stress Tensor method. Our numerical results clearly show that for current carrying wires the Lorentz method is the method of choice. For permanent magnets (for which the Lorentz method no longer applies) the isoparametric second order Eggshell method is more accurate than the Virtual Work or the Maxwell Stress Tensor method. These results make the Eggshell method attractive for use in more complex problems. The Eggshell method is applied on second order isoparametric elements. Its implementation is presented in detail as a set of new MATLAB post-processing routines in the FEMLAB simulation environment.

2000 Mathematics Subject Classification: 65Y20, 78M10
Keywords and Phrases: Finite element method; force computation; magnetostatics; second order isoparametric discretization; Virtual Work; Maxwell stress tensor.
Note: This research is supported by the Dutch Ministry of Economical Affairs within the project IOP-EMVT 02201.

The Eggshell method in a Nutshell

Margreet Nool and Domenico Lahaye

Centrum voor Wiskunde en Informatica (CWI), Amsterdam, The Netherlands

Abstract

Purpose The *eggshell* method was introduced by F. Henrotte as a novel magnetic force computation method. It allows computation of the force by integrating the magnetic stress tensor over a shell surrounding the body of interest. We investigate the numerical properties of this method for current carrying wires, and permanent magnets immersed in two-dimensional stationary magnetic fields, discretized by first and second order isoparametric triangular finite elements.
Design/methodology/Approach We do so by comparing the accuracy of the method, as a function of the mesh size and element order, with the result of three classical force computation methods: the Lorentz, the Virtual Work and the Maxwell Stress Tensor method.
Findings Our numerical results clearly show that for current carrying wires the Lorentz method is the method of choice. For permanent magnets (for which the Lorentz method no longer applies) the isoparametric second order *eggshell* method is more accurate than the Virtual Work or the Maxwell Stress Tensor method.
Research limitations/implications These results make the *eggshell* method attractive for use in more complex problems.
Originality/value The *eggshell* method is applied on second order isoparametric elements. Its implementation is presented in detail as a set of new MATLAB post-processing routines in the FEMLAB simulation environment.
Keywords Finite element method, force computation, magnetostatics, second order isoparametric discretization, Virtual Work, Maxwell stress tensor.
Paper type Research paper

1 Introduction

Several methods exist for computing the magnetic force on a rigid body in the post-processing stage of a finite element (FE) magnetic field computation. It is not immediately clear whether there is an efficient method that gives accurate results for a broad class of problems. The issue of magnetic force computation is therefore still a matter of current research as evidenced by the numerous recent papers on this topic [1, 12, 6, 10, 13, 14, 15, 17].

The aim of this paper is to present implementation aspects and numerical results of the *eggshell* method. This method recently introduced [7, 8] is derived from an energy variation principle. The force is obtained by integrating the Maxwell stress tensor times the gradient of the FE shape functions over a thin layer of finite elements surrounding a device on which the force is computed. The performance of the *eggshell* method is compared with that of three classical force computation methods [18]: the Lorentz, the Maxwell Stress Tensor, and the Virtual Work method.

In this paper we restrict ourselves to two-dimensional magnetostatic field computations with perpendicular current and linear constitutive relations. Numerical results demonstrate that, with first order triangular finite elements, very fine meshes are needed to achieve results which satisfy our accuracy requirements. Second order isoparametric FEs appear to be more efficient to solve the discretized magnetic vector potential equation. Our numerical experiments were carried out with the FEMLAB simulation package [5], applying adaptive mesh refinement around the device on which the force is

computed. The implementation of the *eggshell* method requires a procedure for marking the elements forming the shell and a procedure for integrating over this shell. We wrote these procedures for FEMLAB by extracting information from the FEMLAB mesh data structures.

As test problems for our comparison, we consider the example model *Electromagnetic Forces on Parallel Current Carrying Wires* provided with the Electromagnetics Module [4], pages 71–72, and variants of this model. We doubled the number of wires and replaced one or both wires by a permanent magnet. For each problem we investigate the convergence of the computed force for a decreasing mesh size. The academic nature of this problem is such that simple checks on the computed forces are available.

This paper is structured as follows. In Section 2, the partial differential equation of the magnetic field (2D, perpendicular current) is summarized. Section 3 describes the FE discretization for linear and quadratic triangular elements. For the quadratic elements special attention is paid to elements with curvilinear edges. Classical force computation methods are discussed in Section 4. The theoretical background and implementation of the *eggshell* method is found in Section 5. Section 6 presents the numerical results of both the classical and the *eggshell* methods. Our conclusions are given in Section 7.

2 Perpendicular Current Magnetostatic Formulation

Let $\mathbf{J}$ denote the electric current density in the electrical conductors and $\mathbf{M}$ the magnetization associated with the permanent magnets. For isotropic media the magnetic reluctivity ν can be expressed as $\nu = \nu_0\,\nu_r$, where ν_0 is the magnetic reluctivity of empty space and ν_r the relative permeability. For stationary fields and geometries without moving parts, the double curl equation for the magnetic vector potential $\mathbf{A}$ can be written as

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

This equation is a system of coupled partial differential equations for the three components of $\mathbf{A}$. The magnetic induction $\mathbf{B}$ can be derived from $\mathbf{A}$ by

$$\mathbf{B} = \nabla \times \mathbf{A}\,. \tag{2}$$

The magnetic field $\mathbf{H}$ is related to $\mathbf{B}$ by a constitutive relation, that depends on the materials considered:

$$\mathbf{H} = \nu\,\mathbf{B} - \mathbf{M}\,. \tag{3}$$

For 2D planar geometries with perpendicular current, the vectors $\mathbf{J}$, $\mathbf{M}$, $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{H}$ simplify to

$$\mathbf{J} = (0,\,0,\,J_z)\,, \tag{4}$$
$$\mathbf{M} = (M_x(x,y),\,M_y(x,y),\,0)\,, \tag{5}$$
$$\mathbf{A} = (0,\,0,\,A_z(x,y))\,, \tag{6}$$
$$\mathbf{B} = (B_x(x,y),\,B_y(x,y),\,0)\,, \tag{7}$$
$$\mathbf{H} = (H_x(x,y),\,H_y(x,y),\,0)\,. \tag{8}$$

Given these expressions, the system of coupled PDEs (1) becomes a single scalar PDE for the vector potential component A_z

$$-\frac{\partial}{\partial x}\left(\nu\,\frac{\partial A_z}{\partial x} + M_y\right) - \frac{\partial}{\partial y}\left(\nu\,\frac{\partial A_z}{\partial y} - M_x\right) = J_z\,. \tag{9}$$

This is a diffusion equation with diffusion coefficient ν and source term $J_z - \frac{\partial M_x}{\partial y} + \frac{\partial M_y}{\partial x}$. In this paper, we consider homogeneous Dirichlet boundary conditions, i.e., $A_z = 0$. Having solved this PDE, the

magnetic induction can be computed by the 2D equivalent of (2)

$$B_x = \frac{\partial A_z}{\partial y} \text{ and } B_y = -\frac{\partial A_z}{\partial x}\,. \tag{10}$$

The magnetic field components H_x and H_y in turn are given by (3)

$$H_x = \nu\, B_x - M_x \text{ and } H_y = \nu\, B_y - M_y\,. \tag{11}$$

The FE discretization [16] of (9) requires casting this equation in weak formulation. To this end, we denote the domain on which the PDE is defined by Ω and its boundary by Γ. Furthermore, $L^2(\Omega)$ is the space of square integrable functions and $H^1(\Omega)$ its subspace of functions with square integrable first derivatives [11]. The functions of $H^1(\Omega)$ that vanish on Γ are denoted by $H_0^1(\Omega)$. The space $L^2(\Omega)$ has the scalar product

$$(u,\, v) = \int_\Omega u(\mathbf{x})\, v(\mathbf{x})\, d\mathbf{x}\,. \tag{12}$$

The gradient

$$\nabla u = (\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y})\,, \quad u \in H^1(\Omega) \tag{13}$$

belongs to $L^2(\Omega) \times L^2(\Omega)$. Denoting by $\cdot$ the Euclidean scalar product, the inner product on the latter space is defined as

$$(\underline{u},\, \underline{v}) = \int_\Omega \underline{u}(\mathbf{x}) \cdot \underline{v}(\mathbf{x})\, d\mathbf{x}\,. \tag{14}$$

Given the current density component J_z and the magnetization components M_x and M_y, we introduce on $H_0^1(\Omega)$ the linear form

$$\mathtt{F} : H_0^1(\Omega) \to \mathbb{R} : \mathtt{F}(v) = (J_z - \frac{\partial M_x}{\partial y} + \frac{\partial M_y}{\partial x},\, v)\,, \tag{15}$$

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

$$\mathtt{A} : H_0^1(\Omega) \times H_0^1(\Omega) \to \mathbb{R} : \mathtt{A}(w,\, v) = (\nu\, \nabla w,\, \nabla v)\,. \tag{16}$$

By multiplying both sides of (9) with $v \in H_0^1(\Omega)$, integrating over Ω and applying integration by parts, the following weak formulation of (9) is obtained

$$\text{find } u \in H_0^1(\Omega) \text{ such that } \mathtt{A}(u,\, v) = \mathtt{F}(v)\,, \qquad \forall\, v \in H_0^1(\Omega)\,. \tag{17}$$

Under suitable conditions on ν, J_z, M_x and M_y, this weak formulation has a unique solution, the *weak* solution of (9). Finally, choosing a finite dimensional subspace $X \subset H_0^1(\Omega)$ and solving

$$\text{find } u \in X \text{ such that } \mathtt{A}(u,\, v) = \mathtt{F}(v)\,, \qquad \forall\, v \in X\,, \tag{18}$$

yields the spatial discretization of (17).

3 FE Discretization

In the FE method the space X in (18) is a space of piecewise polynomials. Assuming Ω to be a polygonal domain, we denote by $\mathcal{T}_h$ a triangulation of Ω. The subscript h denotes the mesh width

of the triangulation. When Ω has smooth curved boundaries, the triangulation only resolves the boundary in the limit for $h \to 0$.

We denote by $\mathcal{P}_k$ $(k = 1, 2)$ the space of polynomials in the variables x and y with total degree less than or equal to k. With any triangulation $\mathcal{T}_h$, we associate the finite dimensional subspace

$$X_h = X_h^k := \{v_h \in \mathcal{C}^0(\overline{\Omega}) \,|\, v_{h|T} \in \mathcal{P}_k\,, \quad \forall\, T \in \mathcal{T}_h\}\,, \tag{19}$$

where $v_{h|T}$ is the restriction of v_h to T. For the construction of a basis for the subspace V_h

$$V_h := \{v_h \in X_h^k \,|\, v_h = 0 \text{ on } \Gamma\}, \tag{20}$$

we associate a set of FE nodes Ξ_h^k with the triangulation $\mathcal{T}_h$, defined by

$$\Xi_h = \begin{cases} \Xi_h^1 = \{\mathbf{a}_T^i \,|\, i \in \{1,2,3\}\,, \quad T \in \mathcal{T}_h\}\,, & \text{for first order elements} \\ \Xi_h^2 = \{\mathbf{a}_T^i \,|\, i \in \{1,\cdots,6\}\,, \quad T \in \mathcal{T}_h\}\,, & \text{for second order elements} \end{cases} \tag{21}$$

where $\mathbf{a}_T^i$, $i \in \{1,2,3\}$ and $\mathbf{a}_T^i$, $i \in \{4,5,6\}$ denote the vertices and the midpoints of the edges of triangle T, respectively.

In order to resolve curvilinear boundaries for second order elements, *isoparametric* elements are used (see Section 3.2.1). The midpoints of an edge having vertices on the boundary are then shifted to the boundary. In a global enumeration over all triangles we denote the FE nodes by $\mathbf{a}_j$ and introduce the following subset of Ξ_h

$$\Xi_h^0 = \{\mathbf{a}_j \in \Xi_h \,|\, \mathbf{a}_j \notin \Gamma\}\,. \tag{22}$$

To each node $\mathbf{a}_j \in \Xi_h^0$, we associate the function $N_j \in V_h$ such that

$$N_j(\mathbf{a}_\ell) = \delta_{j\ell}, \qquad \forall \ell : \mathbf{a}_\ell \in \Xi_h\,, \tag{23}$$

where $\delta_{j\ell}$ is the Kronecker delta. The set $\{N_j \,|\, \mathbf{a}_j \in \Xi_h^0\}$ forms a basis for V_h.

Equation (9) is discretized by taking subspace V_h for X in (18), and approximating the solution u by

$$u_h = \sum_{j\,:\,\mathbf{a}_j \in \Xi_h^0} s_j\, N_j\,. \tag{24}$$

The coefficients s_j represent the values of u_h at the FE nodes.

In order to be able to formulate details about the *eggshell* method, we need to repeat some notations and details that are familiar in FE theory.

3.1 Linear Triangular Elements

Let $\widehat{T}_1$ denote the first order *reference* triangle with vertices $(\xi_1, \eta_1) = (0,0)$, $(\xi_2, \eta_2) = (1,0)$ and $(\xi_3, \eta_3) = (0,1)$ in the $\xi\eta$-plane, shown in Fig. 1. On this triangle $\widehat{T}_1$, the first order Lagrange basis functions $\widehat{N}_i$ have the following form

$$\begin{aligned} \widehat{N}_1(\xi,\eta) &= 1 - \xi - \eta, \\ \widehat{N}_2(\xi,\eta) &= \xi, \\ \widehat{N}_3(\xi,\eta) &= \eta. \end{aligned} \tag{25}$$

Let $(x_i, y_i), i \in \{1,2,3\}$ denote the coordinates of the FE nodes $\mathbf{a}_T^i$ of the *general* triangle $T_1 \in \mathcal{T}_h$ as shown in Fig. 1. The area of T_1 is given by

$$\triangle = \frac{1}{2}|(x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)|\,. \tag{26}$$

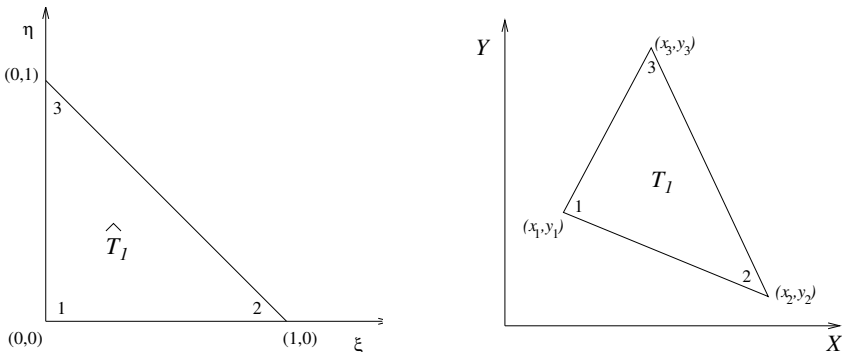


Figure 1: Linear triangular element: reference (*left*) and general (*right*) triangle

By a coordinate transformation from $\widehat{T}_1$ to T_1, the barycentric coordinates ζ_i associated with the vertex $\mathbf{a}_T^i$ can be expressed as

$$\zeta_i = \frac{1}{2\triangle}(a_i + b_i\, x + c_i\, y)\,, \quad i \in \{1,2,3\}, \tag{27}$$

where the values a_i, b_i and c_i, $i \in \{1,2,3\}$ denote (cf. [19], page 89)

$$\begin{array}{lllllllll} a_1 &=& x_2y_3 - x_3y_2, & b_1 &=& y_2 - y_3, & c_1 &=& x_3 - x_2, \\ a_2 &=& x_3y_1 - x_1y_3, & b_2 &=& y_3 - y_1, & c_2 &=& x_1 - x_3, \\ a_3 &=& x_1y_2 - x_2y_1, & b_3 &=& y_1 - y_2, & c_3 &=& x_2 - x_1. \end{array} \tag{28}$$

The first order basis function N_i associated with the vertex $\mathbf{a}_T^i$ can be expressed as $N_i = \zeta_i$, $i \in \{1,2,3\}$. The FE approximation for the magnetic vector potential component on T_1 can be written as

$$u_{h|T_1} = \sum_{i=1}^{3} s_i\, N_i\,. \tag{29}$$

By substituting (29) into (10), the discrete magnetic fluxes $B_{x,h}$ and $B_{y,h}$ on T_1 can be expressed as

$$B_{x,h|T_1} = \sum_{i=1}^{3} s_i \frac{\partial N_i}{\partial y} \text{ and } B_{y,h|T_1} = -\sum_{i=1}^{3} s_i \frac{\partial N_i}{\partial x}\,. \tag{30}$$

We remark that for first order elements $B_{x,h}$ and $B_{y,h}$ are constant per element.

3.2 Quadratic Triangular Elements

Let $\widehat{T}_2$ denote the second order *reference* triangle with vertices $(\xi_1,\eta_1) = (0,0)$, $(\xi_2,\eta_2) = (1,0)$ and $(\xi_3,\eta_3) = (0,1)$ and midpoints of the edges $(\xi_4,\eta_4) = (\frac{1}{2},0)$, $(\xi_5,\eta_5) = (\frac{1}{2},\frac{1}{2})$ and $(\xi_6,\eta_6) = (0,\frac{1}{2})$ in the $\xi\eta$-plane, as shown in Fig. 2. On triangle $\widehat{T}_2$, the second order Lagrange basis functions $\widehat{N}_i$ associated with the nodes (ξ_i,η_i), $i \in \{1,\cdots,6\}$ have the following form

$$\begin{array}{lll} \widehat{N}_1(\xi,\eta) &=& 2(\xi+\eta-1)(\xi+\eta-\frac{1}{2}) = 2\xi^2 + 2\eta^2 + 4\xi\eta - 3\xi - 3\eta + 1, \\ \widehat{N}_2(\xi,\eta) &=& 2\xi(\xi-\frac{1}{2}) = 2\xi^2 - \xi, \\ \widehat{N}_3(\xi,\eta) &=& 2\eta(\eta-\frac{1}{2}) = 2\eta^2 - \eta, \\ \widehat{N}_4(\xi,\eta) &=& -4\xi(\xi+\eta-1) = -4\xi^2 - 4\xi\eta + 4\xi, \\ \widehat{N}_5(\xi,\eta) &=& 4\xi\eta, \\ \widehat{N}_6(\xi,\eta) &=& -4\eta(\xi+\eta-1) = -4\eta^2 - 4\xi\eta + 4\eta. \end{array} \tag{31}$$

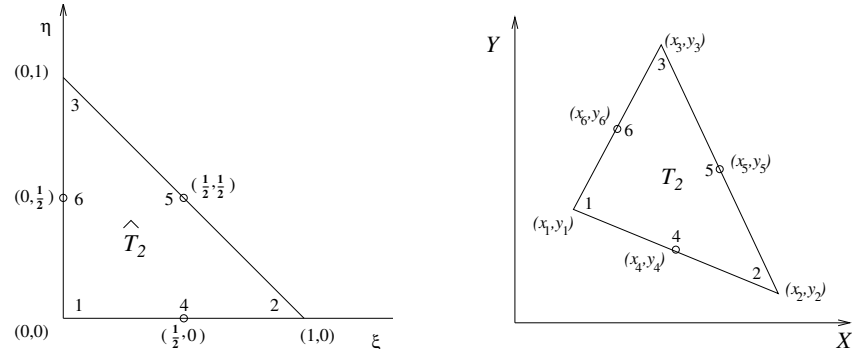


Figure 2: Quadratic triangular element: reference (*left*) and general (*right*) triangle

As in the previous subsection, the expressions for the basis functions on a triangle $T_2 \in \mathcal{T}_h$ can be obtained by a coordinate transformation from $\widehat{T}_2$ to T_2. Let (x_i, y_i), $i \in \{1, \cdots, 6\}$ denote the coordinates of the FE nodes $\mathbf{a}_T^i$ of a second order element. The second order basis functions N_i associated with the nodes $\mathbf{a}_T^i$ and their natural derivatives can be expressed by

$$N = \begin{bmatrix} N_1^e \\ N_2^e \\ N_3^e \\ N_4^e \\ N_5^e \\ N_6^e \end{bmatrix} = \begin{bmatrix} \zeta_1(2\zeta_1 - 1) \\ \zeta_2(2\zeta_2 - 1) \\ \zeta_3(2\zeta_3 - 1) \\ 4\zeta_1\zeta_2 \\ 4\zeta_2\zeta_3 \\ 4\zeta_3\zeta_1 \end{bmatrix}, \quad \begin{bmatrix} \frac{\partial N}{\partial \zeta_1} & \frac{\partial N}{\partial \zeta_2} & \frac{\partial N}{\partial \zeta_3} \end{bmatrix} = \begin{bmatrix} 4\zeta_1 - 1 & 0 & 0 \\ 0 & 4\zeta_2 - 1 & 0 \\ 0 & 0 & 4\zeta_3 - 1 \\ 4\zeta_2 & 4\zeta_1 & 0 \\ 0 & 4\zeta_3 & 4\zeta_2 \\ 4\zeta_3 & 0 & 4\zeta_1 \end{bmatrix}. \tag{32}$$

where the functions ζ_i are defined in (27). The discrete magnetic vector potential can be expressed as six-term equivalent of (29), and the discrete magnetic flux densities as six-term equivalent of (30).

3.2.1 Quadratic Triangular Elements with Curvilinear Edges

When a second order triangle $T \in \mathcal{T}_h$ has at least one curved edge, expressions for the Cartesian derivatives of the shape functions N_i need adjustment. These adjustments are detailed here for a 6-node isoparametric triangle as shown in Fig. 3. The element geometry is defined by the coordinates

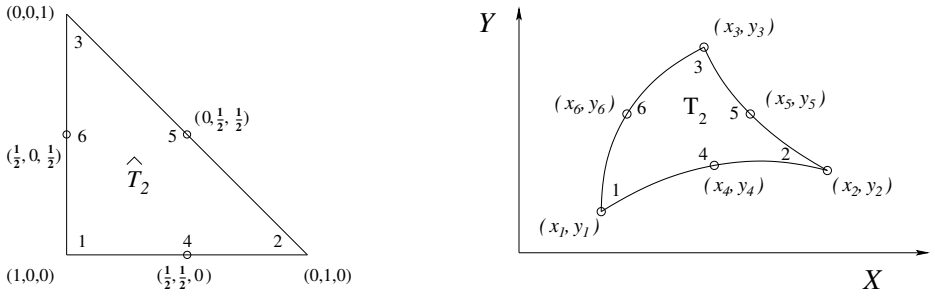


Figure 3: The 6-node quadratic isoparametric triangle: reference (*left*) and general (*right*) triangle

of the nodes $(x_i\,, y_i)$ $i \in \{1, \cdots, 6\}$. The corner nodes are numbered **1, 2, 3** in counterclockwise sense.

The side nodes are numbered **4, 5, 6** opposite to corners **3, 1, 2**, respectively. It is shown in ([2], Ch. 24), that the differential area element of T is given by

$$dT = Jd\zeta_1 d\zeta_2 d\zeta_3, \tag{33}$$

where the Jacobian J is one-half times the determinant of the Jacobian matrix **Jac**

$$\mathbf{Jac} = \begin{bmatrix} 1 & 1 & 1 \\ \sum_{i=1}^{6} x_i \frac{\partial N_i}{\partial \zeta_1} & \sum_{i=1}^{6} x_i \frac{\partial N_i}{\partial \zeta_2} & \sum_{i=1}^{6} x_i \frac{\partial N_i}{\partial \zeta_3} \\ \sum_{i=1}^{6} y_i \frac{\partial N_i}{\partial \zeta_1} & \sum_{i=1}^{6} y_i \frac{\partial N_i}{\partial \zeta_2} & \sum_{i=1}^{6} y_i \frac{\partial N_i}{\partial \zeta_3} \end{bmatrix}, \qquad J = \frac{1}{2}\det(\mathbf{Jac}). \tag{34}$$

J is a second order polynomial in ζ_1, ζ_2 and ζ_3. The side nodes may be arbitrarily located within the constraint that J remains positive. A scalar function w interpolated over a second order triangle satisfies $\frac{\partial w}{\partial x} = \sum_{i=1}^{6} w_i \frac{\partial N}{\partial x}$ (likewise for $\frac{\partial}{\partial y}$). Taking $w = 1, x, y$, one obtains the following system of linear equations

$$\mathbf{Jac} \begin{bmatrix} \frac{\partial \zeta_1}{\partial x} & \frac{\partial \zeta_1}{\partial y} \\ \frac{\partial \zeta_2}{\partial x} & \frac{\partial \zeta_2}{\partial y} \\ \frac{\partial \zeta_3}{\partial x} & \frac{\partial \zeta_3}{\partial y} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{35}$$

Solving this system yields

$$\begin{bmatrix} \frac{\partial \zeta_1}{\partial x} & \frac{\partial \zeta_1}{\partial y} \\ \frac{\partial \zeta_2}{\partial x} & \frac{\partial \zeta_2}{\partial y} \\ \frac{\partial \zeta_3}{\partial x} & \frac{\partial \zeta_3}{\partial y} \end{bmatrix} = \frac{1}{2J} \begin{bmatrix} J_{y23} & J_{x32} \\ J_{y31} & J_{x13} \\ J_{y12} & J_{x21} \end{bmatrix}, \tag{36}$$

in which $J_{xji} = \sum_{k=1}^{6} x_k (\frac{\partial N_k}{\partial \zeta_j} - \frac{\partial N_k}{\partial \zeta_i})$, $J_{yji} = \sum_{k=1}^{6} y_k (\frac{\partial N_k}{\partial \zeta_j} - \frac{\partial N_k}{\partial \zeta_i})$ and $J = \frac{1}{2}(J_{x21}J_{y31} - J_{y12}J_{x13})$. Taking the dot product of the natural-coordinate partials (32) with the node coordinates, the entries of the matrix in the right-hand side of (36) can be expressed as linear polynomials is ζ_1, ζ_2 and ζ_3

$$\begin{array}{lcll} J_{x21} = x_2 - x_1 & + & 4(\Delta x_4(\zeta_1 - \zeta_2) & + & (\Delta x_5 - \Delta x_6)\zeta_3), \\ J_{x32} = x_3 - x_2 & + & 4(\Delta x_5(\zeta_2 - \zeta_3) & + & (\Delta x_6 - \Delta x_4)\zeta_1), \\ J_{x13} = x_1 - x_3 & + & 4(\Delta x_6(\zeta_3 - \zeta_1) & + & (\Delta x_4 - \Delta x_5)\zeta_2), \\ J_{y12} = y_1 - y_2 & + & 4(\Delta y_4(\zeta_2 - \zeta_1) & + & (\Delta y_6 - \Delta y_5)\zeta_3), \\ J_{y23} = y_2 - y_3 & + & 4(\Delta y_5(\zeta_3 - \zeta_2) & + & (\Delta y_4 - \Delta y_6)\zeta_1), \\ J_{y31} = y_3 - y_1 & + & 4(\Delta y_6(\zeta_1 - \zeta_3) & + & (\Delta y_5 - \Delta y_4)\zeta_2), \end{array} \tag{37}$$

in which

$$\begin{array}{lll} \Delta x_4 = x_4 - \frac{1}{2}(x_1 + x_2), & \Delta x_5 = x_5 - \frac{1}{2}(x_2 + x_3), & \Delta x_6 = x_6 - \frac{1}{2}(x_3 + x_1), \\ \Delta y_4 = y_4 - \frac{1}{2}(y_1 + y_2), & \Delta y_5 = y_5 - \frac{1}{2}(y_2 + y_3), & \Delta y_6 = y_6 - \frac{1}{2}(y_3 + y_1). \end{array} \tag{38}$$

The Cartesian derivatives of the shape functions are now given by

$$\frac{\partial N}{\partial x} = \frac{1}{2J} \begin{bmatrix} (4\zeta_1 - 1)J_{y23} \\ (4\zeta_2 - 1)J_{y31} \\ (4\zeta_3 - 1)J_{y12} \\ 4(\zeta_2 J_{y23} + \zeta_1 J_{y31}) \\ 4(\zeta_3 J_{y31} + \zeta_2 J_{y12}) \\ 4(\zeta_1 J_{y12} + \zeta_3 J_{y23}) \end{bmatrix}, \quad \frac{\partial N}{\partial y} = \frac{1}{2J} \begin{bmatrix} (4\zeta_1 - 1)J_{x32} \\ (4\zeta_2 - 1)J_{x13} \\ (4\zeta_3 - 1)J_{x21} \\ 4(\zeta_2 J_{x32} + \zeta_1 J_{x13}) \\ 4(\zeta_3 J_{x13} + \zeta_2 J_{x21}) \\ 4(\zeta_1 J_{x21} + \zeta_3 J_{x32}) \end{bmatrix}. \tag{39}$$

Remark 3.1 *It is clear that the straight-sided triangle geometry, i.e., $\Delta x_4 = \Delta x_5 = \Delta x_6 = 0$, is a special case for which the J_{xji}'s and J_{yji}'s are independent of the ζ_i's, and the Jacobian* **Jac** *is constant.*

4 Classical Force Computation Methods

In this section we recall three classical methods for computing the magnetic force: the Lorentz method, the Maxwell Stress Tensor method and the Virtual Work method.

4.1 Lorentz Method

In Lorentz method the total magnetic force $\mathbf{F}$ on a current carrying device is computed by integrating the volume force density $\mathbf{J} \times \mathbf{B}$ over the volume V of the device

$$\mathbf{F} = \int_V \mathbf{J} \times \mathbf{B}\, dV\,. \tag{40}$$

In 2D perpendicular current computations the domain of integration reduces to a cross section which we denote by Ω_1. Substituting (4) and (7) for $\mathbf{J}$ and $\mathbf{B}$ into (40) we obtain for the x- and y-components of the total force

$$F_x = -\int_{\Omega_1} J_z\, B_y\, d\Omega_1 \text{ and } F_y = \int_{\Omega_1} J_z\, B_x\, d\Omega_1\,. \tag{41}$$

Remark 4.1 *Lorentz method can only be applied to electrical conductors with $\mu_r = 1$. The method does not allow computing the force on permanent magnets.*

4.2 Maxwell Stress Tensor Method

In the Maxwell Stress Tensor(MST) method the total force on a magnetizable device with unit outward normal $\mathbf{n}$ on its enclosing surface S is computed by integrating the product of the Maxwell stress tensor σ_{EM} and $\mathbf{n}$ over the enclosing surface

$$\mathbf{F} = \int_S \sigma_{EM}\, \mathbf{n}\, dS\,. \tag{42}$$

In this paper we always assume that the device on which we want to compute the force is surrounded by empty space. Then the Maxwell stress tensor is given by

$$\sigma_{EM} = -\frac{1}{2\mu_0}\,(\mathbf{B}\cdot\mathbf{B})\, I + \frac{1}{\mu_0}\mathbf{B}\,\mathbf{B}^T\,, \tag{43}$$

where I denotes the identity matrix.

In 2D perpendicular current computations the stress tensor simplifies to the 2×2 matrix

$$\sigma_{EM} = -\frac{1}{2\mu_0}\begin{pmatrix} B_x^2 + B_y^2 & 0 \\ 0 & B_x^2 + B_y^2 \end{pmatrix} + \frac{1}{\mu_0}\begin{pmatrix} B_x^2 & B_x\,B_y \\ B_x\,B_y & B_y^2 \end{pmatrix}, \tag{44}$$

while the domain of integration in (42) reduces to the boundary of the 2D cross section of the device denoted by Γ_1. By substituting

$$H_x = \nu_0\, B_x \text{ and } H_y = \nu_0\, B_y \tag{45}$$

in (44) we obtain the following expressions for the x- and y-components of the total force

$$F_x = \int_{\Gamma_1} \left[-\frac{1}{2}\, n_x\,(H_x\, B_x + H_y\, B_y) + B_x\,(n_x\, H_x + n_y\, H_y)\right] d\Gamma_1\,, \tag{46}$$

and

$$F_y = \int_{\Gamma_1} \left[-\frac{1}{2}\, n_y\,(H_x\, B_x + H_y\, B_y) + B_y\,(n_x\, H_x + n_y\, H_y)\right] d\Gamma_1\,. \tag{47}$$

4.3 Virtual Work Method

In the Virtual Work method the force is computed from the magnetic energy of the system. The magnetic energy W_{EM} of a system with volume V is given by

$$W_{EM} = \int_V \left(\int_0^B \mathbf{H} \cdot d\mathbf{B} \right) . \tag{48}$$

Under constant magnetic flux condition, the magnetic force is computed as

$$\mathbf{F}_\Phi = -\nabla W_{EM} . \tag{49}$$

Under constant current condition, the force is computed similarly, but with opposite sign.

5 Eggshell Method

5.1 Theoretical Background

In this section we give a description of the *eggshell* method for computing the force, omitting details that can be found in [7, 8, 9]. In [8] the authors derive an expression for the time-variation of the total energy of an electromechanical system Ω subject to a deformation. This variation is the sum of two terms: a term equal to the rate of change of electromagnetic energy stored in the system Ω and a term expressing the mechanical work power received by Ω. The latter is the time derivative of the work W_{EM} done by the electromagnetic forces. The deformation of Ω is described by a vector field $\mathbf{v}$ that attributes a velocity to each material particle in Ω. The spatial gradient of $\mathbf{v}$ is denoted by $\nabla\mathbf{v}$. In [8] the Maxwell stress tensor σ_{EM} is defined as the dual of $\nabla\mathbf{v}$ as follows

$$\frac{\partial W_{EM}}{\partial t} = \dot{W}_{EM} = \int_\Omega \sigma_{EM} \, : \, \nabla\mathbf{v} \, d\Omega , \tag{50}$$

where : denotes the tensor product, i.e., $T \, : \, R = \sum_{ij} T_{ij} \, R_{ij}$. Given an expression for the electromagnetic energy, the formalism proposed in [8] allows to derive an expression for the Maxwell stress tensor. Integration by parts allows to rewrite (50) as

$$\dot{W}_{EM} = -\int_\Omega (\operatorname{div} \sigma_{EM}) \cdot \mathbf{v} \, d\Omega + \int_\Gamma \mathbf{n} \, \sigma_{EM} \, \mathbf{v} \, d\Gamma , \tag{51}$$

where $\mathbf{n}$ denotes the outward normal at Γ. From this expression the authors derive an expression for the total magnetic force $\mathbf{F}$ on a rigid body Ω_1. The rigid body Ω_1 is assumed to be shifted an infinitesimal amount $\delta\mathbf{u}$. It is argued that for computing the time derivative of the work W_{EM}, it can be assumed that displacement causes only a small region S (called the *eggshell*) surrounding the body to deform. The (virtual) velocity field associated with that deformation and its gradient are

$$\mathbf{v} = \gamma \, \delta\dot{\mathbf{u}} , \qquad \nabla\mathbf{v} = \nabla\gamma \, \delta\dot{\mathbf{u}} , \tag{52}$$

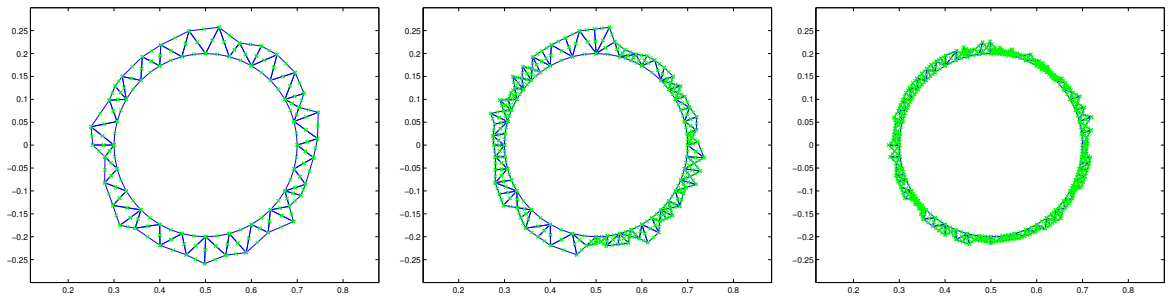
where γ is a smooth function whose value is 1 on the inner surface of the shell S touching Ω_1 and 0 on its outer surface. The gradient $\nabla\mathbf{v}$ is 0 outside the shell S. Using (50), one consequently obtains

$$\dot{W}_{EM} = -\mathbf{F} \cdot \delta\dot{\mathbf{u}} = \int_S \sigma_{EM} \, : \, \nabla\mathbf{v} \, dS , \tag{53}$$

where $\mathbf{F}$ is the *resultant force* on Ω_1. Finally, one obtains by using (52) and after factoring out $\delta\dot{\mathbf{u}}$

$$\mathbf{F} = \int_S \sigma_{EM} \, \nabla\gamma \, dS , \tag{54}$$

which is the *eggshell* formula for the force on the rigid body Ω_1. In the limit of an infinitely thin shell, one readily recovers expression (42).



(a) Shell for initial mesh

(b) Shell after one adaptive refinement step

(c) Shell after two adaptive refinement steps

Figure 4: Illustration of the *eggshell* S surrounding a circular cross section Ω_1 on three increasingly refined meshes

5.2 Implementation of Eggshell Method

The implementation of the *eggshell* method requires procedures for marking the elements forming the shell and for integrating over this shell. We wrote these procedures as MATLAB functions, extracting information from the FEMLAB `mesh` data structures. Isoparametric discretizations for those elements that touch the boundary of Ω_1 are taken into account.

5.2.1 Construction of the shell

The starting point for our implementation consists of identifying those triangles $T \in \mathcal{T}_h$ whose union defines the shell S. In Fig. 4 the shell around a circular cross section Ω_1 is shown on a sequence of three adaptively constructed meshes. The finer the mesh, the thinner the *eggshell* becomes. The construction of the shell is a two step process. In the first step, the FE nodes $\mathbf{a}_T^i \in \Xi_h$ lying on the boundary of Ω_1 are collected in a list. In a second step, all triangles $T \in \mathcal{T}_h$ having at least on node in the boundary list and lying in the exterior of Ω_1 are marked. These steps require retrieving information from the FEMLAB `mesh` structure. For details, we refer to [3] (from page 3-42 to page 3-55). At the end of the two-step process, the $(x,\, y)$-coordinates and the global numbers of the FE nodes of the triangles in the shell are available.

5.2.2 Computation of the derivatives of the FE shape functions

Having the coordinates of the FE nodes at our disposal, the $(x,\, y)$-derivatives of the FE shape functions can be constructed using the expressions (39). On S these derivatives are used to represent both the discrete magnetic flux components $B_{x,h}$ and $B_{y,h}$ and the function γ introduced in (52).

For second order elements, the expressions (39) are polynomials in $\zeta_i, i \in \{1, 2, 3\}$. To represent these polynomials, a class for polynomials `polynom3` has been introduced. This class consists of a vector holding the polynomial coefficients and a set of operations on this representation. For example, if `e` is a vector of length 9, then $p =$ `polynom3(e)` represents the polynomial

$$p = e_0 + e_1\zeta_1 + e_2\zeta_2 + e_3\zeta_3 + e_4{\zeta_1}^2 + e_5\,\zeta_1\zeta_2 + e_6\zeta_1\zeta_3 + e_7{\zeta_2}^2 + e_8\zeta_2\zeta_3 + e_9{\zeta_3}^2. \tag{55}$$

The operations in the class allow to evaluate a polynomial in a point, to make linear combinations of polynomials and to multiply polynomials. For curvilinear second order elements, the shape function derivatives (39) must be divided by the polynomial $2J$. Division of two polynomials is not implemented. Instead, division is delayed until quadrature over the shell is performed.

The code **NxNy**, shown in Fig. 5, calculates the Jacobian J and the $(x,\, y)$-derivatives of the shape functions for a single shell element. Properly speaking, `Nx` and `Ny` correspond to $2\,J\,\frac{\partial N}{\partial x}$ and $2\,J\,\frac{\partial N}{\partial y}$, respectively. The output values `J`, `Nx` and `Ny` can be constant, first or second order polynomials in ζ_1, ζ_2, and ζ_3, as commented in Fig. 5.

5.2.3 Assembling the stress tensor

The $(x,\, y)$-components of the discrete magnetic flux on an element T in the shell can be constructed as linear combinations of the shape function derivatives corresponding to (30),

$$B_{x,h|T} = \sum_{i=1}^{6} s_i \frac{\partial N_i}{\partial y} \quad \text{and} \quad B_{y,h|T} = -\sum_{i=1}^{6} s_i \frac{\partial N_i}{\partial x}. \tag{56}$$

The coefficients s_i in this expansion are retrieved from the discrete vector potential computed by FEMLAB. The global numbering of the FE nodes in the shell is used to this end.

The magnetic stress tensor σ_{EM}, computed using expression (44) becomes a 2×2 matrix with polynomials in ζ_i as entries. The implementation **Stress** of σ_{EM} for a single shell element is given at the bottom of Fig. 8.

5.2.4 Integration of $\sigma_{EM}\nabla\gamma$ over the *eggshell* using Gaussian quadrature

For integrating $\sigma_{EM}\nabla\gamma$ over the shell, Gaussian quadrature is used. For isoparametric discretizations, the Jacobian (34) is a function of the barycentric coordinates ζ_1, ζ_2 and ζ_3 and can thus not be factored out of the integration rules. We consider the following five Gaussian quadrature rules ([2], Ch. 24):

- the *one point rule*
  When first order discretizations are used, $\sigma_{EM}\nabla\gamma$ is element-wise constant. In this case, the one-point rule

  $$\int_{\Omega^e} F(\zeta_1, \zeta_2, \zeta_3)\, d\Omega^e \approx J(\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}) F(\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}), \tag{57}$$

  is exact. For second order discretizations higher order rules are required.

- two *three point rules*
  We will use two three-point rules, referred to as `rule=3` and `rule=-3`. Both rules have weights equal to $\frac{1}{3}$, but differ in the position of the quadrature points, as indicated in Fig. 6 (b and c). The `rule=-3` or midpoint rule e.g. is given by

  $$\int_{\Omega^e} F(\zeta_1, \zeta_2, \zeta_3)\, d\Omega^e \approx \frac{1}{3} J(\tfrac{1}{2}, \tfrac{1}{2}, 0)\, F(\tfrac{1}{2}, \tfrac{1}{2}, 0) + \frac{1}{3} J(0, \tfrac{1}{2}, \tfrac{1}{2})\, F(0, \tfrac{1}{2}, \tfrac{1}{2}) + \frac{1}{3} J(\tfrac{1}{2}, 0, \tfrac{1}{2})\, F(\tfrac{1}{2}, 0, \tfrac{1}{2}). \tag{58}$$

- one *six* and one *seven point rule*
  The quadrature points of these Gaussian quadrature rules are shown in Fig. 6 (d and e). Their weights are given in Fig. 7.

Fig. 7 displays the implementation of the Gaussian rules in **TriGaussRuleInfo**. The barycentric coordinates of the quadrature points and the corresponding weights are given in the arrays `Gauss.`ζ and `Gauss.weight`, respectively.

Due to the definition of the function γ in (52), only nodes lying on the boundary of rigid body Ω_1 contribute to the integral (54). For a 6-node triangle, as shown in Fig. 3, this implies that when two vertices lie on the boundary, the midpoint node contributes as well. The input parameter `index`

```
function [J,Nx,Ny] = NxNy(node,order)
% The function NXNY computes the Jacobian determinant J, and the Cartesian derivatives of the shape functions
% multiplied by twice this determinant.

% Note:
% If order==1 J,J×Nx, J×Ny are constants
% If order==2 and non-curved triangles, J is constant, and 2J×Nx, 2J×Ny are polynomials of degree 1,
%                  for curved triangles J, 2J×Nx, 2J×Ny are polynomials of degree 2 in ξ1, ξ2 and ξ3

  if (order==1)
    Jx21=node(1,2)-node(1,1); Jx32=node(1,3)-node(1,2); Jx13=node(1,1)-node(1,3);
    Jy12=node(2,1)-node(2,2); Jy23=node(2,2)-node(2,3); Jy31=node(2,3)-node(2,1);

    % dN/dx = 1/(2*J) Nx{1:3}; dN/dy = 1/(2*J) Ny{1:3}
    Nx=cell(1,3); Nx{1} =Jy23; Nx{2} =Jy31; Nx{3} =Jy12;
    Ny=cell(1,3); Ny{1} =Jx32; Ny{2} =Jx13; Ny{3} =Jx21;
  elseif (order==2)
    dx(4:6)=4*[node(1,4)-0.5*(node(1,1)+node(1,2)),node(1,5)-0.5*(node(1,2)+node(1,3)), ...
               node(1,6)-0.5*(node(1,3)+node(1,1))];
    Jx21=polynom3([node(1,2)-node(1,1),dx(4),-dx(4),(dx(5)-dx(6))]);
    Jx32=polynom3([node(1,3)-node(1,2),(dx(6)-dx(4)),dx(5),-dx(5)]);
    Jx13=polynom3([node(1,1)-node(1,3),-dx(6),(dx(4)-dx(5)),dx(6)]);

    dy(4:6)=4*[node(2,4)-0.5*(node(2,1)+node(2,2)),node(2,5)-0.5*(node(2,2)+node(2,3)), ...
               node(2,6)-0.5*(node(2,3)+node(2,1))];
    Jy12=polynom3([node(2,1)-node(2,2),-dy(4),dy(4),(dy(6)-dy(5))]);
    Jy23=polynom3([node(2,2)-node(2,3),(dy(4)-dy(6)),-dy(5),dy(5)]);
    Jy31=polynom3([node(2,3)-node(2,1),dy(6),(dy(5)-dy(4)),-dy(6)]);

    % dN/dx = 1/(2*J) Nx{1:6}
    Nx=cell(1,6); Nx{1} =polynom3([-1,4,0,0])*Jy23; Nx{2} =polynom3([-1,0,4,0])*Jy31;
    Nx{3} =polynom3([-1,0,0,4])*Jy12;
    Nx{4} =polynom3([ 0,0,4,0])*Jy23+polynom3([ 0,4,0,0])*Jy31;
    Nx{5} =polynom3([ 0,0,0,4])*Jy31+polynom3([ 0,0,4,0])*Jy12;
    Nx{6} =polynom3([ 0,4,0,0])*Jy12+polynom3([ 0,0,0,4])*Jy23;

    % dN/dy = 1/(2*J) Ny{1:6}
    Ny=cell(1,6); Ny{1} =polynom3([-1,4,0,0])*Jx32; Ny{2} =polynom3([-1,0,4,0])*Jx13;
    Ny{3} =polynom3([-1,0,0,4])*Jx21;
    Ny{4} =polynom3([ 0,0,4,0])*Jx32+polynom3([ 0,4,0,0])*Jx13;
    Ny{5} =polynom3([ 0,0,0,4])*Jx13+polynom3([ 0,0,4,0])*Jx21;
    Ny{6} =polynom3([ 0,4,0,0])*Jx21+polynom3([ 0,0,0,4])*Jx32;
  end

  % Determinant
  J=0.5*(Jx21*Jy31-Jy12*Jx13);
```

Figure 5: The function **NxNy** computes the Jacobian J and the derivatives of the shape functions $2J\frac{\partial N}{\partial x}$ and $2J\frac{\partial N}{\partial y}$

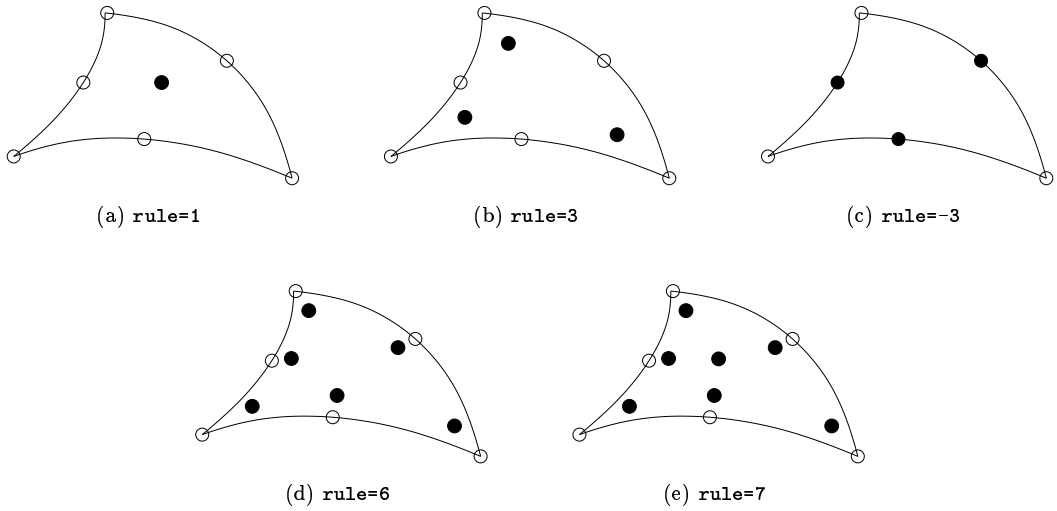


Figure 6: Gaussian quadrature points (dark circles) for the curved 6-node triangles

of function **IntegralElement** in Fig. 8 contains a list of indices corresponding with the boundary vertices. The input parameter `order` indicates whether first or second order elements are used. In case of first order elements (`order` = 1), the *one point* rule is applied. In the `j`-loop for second order elements, (cf. line 28–31 in Fig. 8), the gradient of a shape function centered on the boundary is evaluated in the quadrature point $\zeta^{(m)} = (\zeta_1^{(m)}, \zeta_2^{(m)}, \zeta_3^{(m)}), m \in \{1, \cdots, M\}$, where M is the number of quadrature points. Note that within this loop the division by $2J$, evaluated in $\zeta^{(m)}$, is made as well. According to (54), this gradient is multiplied by the stress tensor σ_{EM}, evaluated in $\zeta^{(m)}$ (line 21–22 and 33–34 of Fig. 8). Next, the product is divided by $(2J(\zeta^{(m)}))^2$ and scaled by the Gauss weights. Finally, the output parameter `Integral` corresponds to the contribution to the (x, y)-components of the magnetic force for that particular triangle element, according to the quadrature rule.

6 Numerical Results

In this section, we compare the performance of the Lorentz, the MST and the Virtual Work method (described in Section 4) and the *eggshell* method (described in Section 5). For this purpose, we examine the model problem of the electromagnetic forces on parallel current carrying wires from the Electromagnetics Module of FEMLAB (see [4], page 71) and three variations on this model. In all models, the vector potential equation has been solved using first and second order isoparametric elements on adaptively constructed meshes of triangles. Dirichlet boundary conditions are set on the boundary of the computational domain. Only in the first model problem an analytical expression for the force is available. However, we can verify our computational results of all four models by either symmetry arguments, or, by checking that the resultant force on the different parts of the model vanishes.

In FEMLAB, the **postint** function has been used to compute both the surface integrals (41) in the Lorentz method and the line integrals (46) and (47) in the MST method. The Virtual Work method has been implemented in the FEMLAB function **cemforce**, which supports linear elements only.

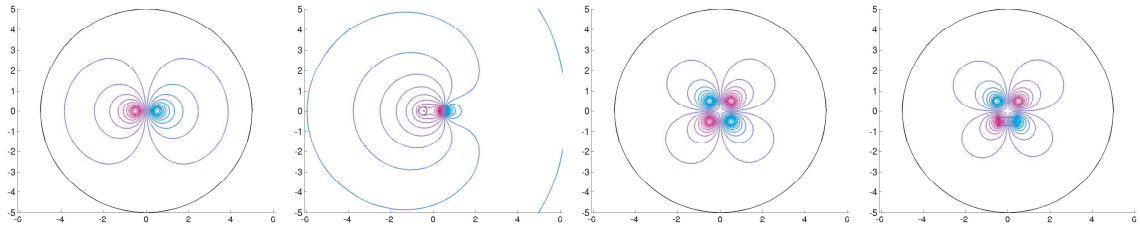
In the following four subsections, results of the computed forces versus the number of the FE degrees of freedom (DOF) are plotted in two columns: the left and right column show the results obtained using first and second order elements, respectively. In the legend of the figures **S1** (`shape=1`)

```
function [Gauss]=TriGaussRuleInfo(rule)
% Function TRIGAUSSRULEINFO gives number of Gauss points, the degree of the Gauss rule,
% the weights and coordinates of the Gauss quadrature rules.

  if (rule==1)
    Gauss.M=1; Gauss.degree=1; Gauss.weight(1) = 1.00000000000000;
    Gauss.ζ =[[0.33333333333333 0.33333333333333 0.33333333333333]];
  elseif (rule==-3)
    Gauss.M=3; Gauss.degree=2; Gauss.weight(1:3) = 0.33333333333333;
    Gauss.ζ =[[0.50000000000000 0.50000000000000 0.00000000000000]; ...
              [0.00000000000000 0.50000000000000 0.50000000000000]; ...
              [0.50000000000000 0.00000000000000 0.50000000000000]];
  elseif (rule==3)
    Gauss.M=rule; Gauss.degree=2; Gauss.weight(1:3) = 0.33333333333333;
    Gauss.ζ =[[0.666666666666667 0.166666666666667 0.166666666666667]; ...
              [0.166666666666667 0.666666666666667 0.166666666666667]; ...
              [0.166666666666667 0.166666666666667 0.666666666666667]];
  elseif(rule==6)
    Gauss.M=rule; Gauss.degree=4;
    Gauss.weight(1:3) = 0.109951743655322; Gauss.weight(4:6) = 0.223381589678011;
    Gauss.ζ =[[0.816847572980459 0.091576213509771 0.091576213509771]; ...
              [0.091576213509771 0.816847572980459 0.091576213509771]; ...
              [0.091576213509771 0.091576213509771 0.816847572980459]; ...
              [0.108103018168070 0.445948490915965 0.445948490915965]; ...
              [0.445948490915965 0.108103018168070 0.445948490915965]; ...
              [0.445948490915965 0.445948490915965 0.108103018168070]];
  elseif(rule==7)
    Gauss.M=rule; Gauss.degree=5; Gauss.weight(1) = 0.225000000000000;
    Gauss.weight(2:4) = 0.125939180544827; Gauss.weight(5:7) = 0.132394152788506;
    Gauss.ζ =[[0.333333333333333 0.333333333333333 0.333333333333333]; ...
              [0.797426985353087 0.101286507323456 0.101286507323456]; ...
              [0.101286507323456 0.797426985353087 0.101286507323456]; ...
              [0.101286507323456 0.101286507323456 0.797426985353087]; ...
              [0.059715871789770 0.470142064105115 0.470142064105115]; ...
              [0.470142064105115 0.059715871789770 0.470142064105115]; ...
              [0.470142064105115 0.470142064105115 0.059715871789770]];
  end
```

Figure 7: The function **TriGaussRuleInfo** sets up Gaussian quadrature on a triangle

```
function[Integral]=IntegralElement(Gauss,Node,index,order,s,μ0);
% INTEGRALELEMENT computes the (x,y)-components of the total force on a single element
% given by its barycentric coordinates of the boundary nodes in the array ζ.

  [J,Nx,Ny]=NxNy(Node,order);
  ζ=Gauss.ζ; weight=Gauss.weight; M=Gauss.M;

  Stress=ComputeStress(s,μ0,Nx,Ny);

  if (order==1)
    % For first order the Nx and Ny values can be computed directly, since the Jacobian does not depend on ζ^(m)
    for j=1:3
      Nx{j}=Nx{j}/(2*J); Ny{j}=Ny{j}/(2*J);
    end
    Gradient=[0;0];
    for j=index
      % Gradient=sum (over j) of [dN(j)/dx dN(j)/dy]^T
      Gradient=Gradient + [Nx{j};Ny{j}];
    end
    % funm = Stress × Gradient
    funm = [double(Stress.c11) * Gradient(1) + double(Stress.c12) * Gradient(2);...
            double(Stress.c21) * Gradient(1) + double(Stress.c22) * Gradient(2)];
    Integral=J*funm; % weight = 1
  else
    Integral=[0;0];
    for m=1:M
      Gradient=[0;0]; Jm=J(ζ(m,1:3));
      for j=index
        % Gradient=sum (over j) of [dN(j)/dx dN(j)/dy]^T evaluated in ζ^(m)
        Gradient=Gradient + [Nx{j}(ζ(m,:));Ny{j}(ζ(m,:))]/(2*Jm);
      end
      % funm = Stress (evaluated in ζ^(m)) × Gradient
      funm = [Stress.c11(ζ(m,:)) * Gradient(1) + Stress.c12(ζ(m,:)) * Gradient(2);...
              Stress.c21(ζ(m,:)) * Gradient(1) + Stress.c22(ζ(m,:)) * Gradient(2)]/(4*Jm^2);
      Integral = Integral + weight(m)*Jm*funm;
    end
  end

function Stress=ComputeStress(s,μ0,Nx,Ny);
% COMPUTESTRESS assembles the stress tensor σ_EM times (2 J)^2 in a structure array with fields of polynomials

  Bx = polynom3(0.0); By = Bx;
  for j=1:length(Nx)
    Bx = Bx + s(j) * Ny{j}; By = By − s(j) * Nx{j};
  end

  c11 = 1./(2*μ0) * (By*By − Bx*Bx); c12 = −1./μ0 * Bx*By;
  Stress = struct('c11',c11,'c12',c12,'c21',c12,'c22',−1. * c11);
```

Figure 8: The function **IntegralElement** computes the contribution of a single triangular element to the total magnetic force. The function **ComputeStress** computes the stress tensor times $(2J)^2$ for first and second order triangles in free space



(a) `Model 1` (b) `Model 2 (zoomed)` (c) `Model 3` (d) `Model 4`

Figure 9: Flux line plot for model 1 up to model 4

refers to first order elements, while **S2** (`shape=2`) refers to second order elements. In ordering the methods, the MST method is treated differently. Numerical results show that the MST method with first order triangles requires very fine meshes in order to obtain good results. We therefore report on the MST method for quadratic elements only. To make the difference in convergence for the different methods more clear, we chose to insert the results of the MST method for quadratic elements in the left column instead of the right one. As discussed in Section 5.2.4, we always use the `rule=1` for the first order *eggshell* method; for the second order *eggshell* method numerical results are given for the Gauss rules `rule=3`, `rule=-3` and `rule=6`. It is observed that Gauss rule `rule=7` does not give more accurate results than Gauss rule `rule=6`. According to Remark 4.1, no Lorentz method results in model problems with a permanent magnets are shown.

6.1 First Model Problem

We consider a configuration of two parallel current carrying wires of infinite length placed in empty space. The wires have a circular cross section of equal radius and their centers are placed on the x-axis at equal distance from the origin. The current is equal to $1\,A$ and $-1\,A$ in the left and right wire, respectively. The current density is constant over the wires and they are assumed to be non-permeable. The empty space surrounding the wires is modelled by a circle with a radius that is large compared to the radius of the wires. Details of the first model problem are given by

radius of wires (r)	$20\,cm$
center of left/right wire	$(\pm 0.5\,m, 0.0\,m)$
current density (J_0)	$1/\pi r^2\; A/m^2$
current density (J_z) in left/right wire	$\pm J_0$
radius of computational domain	$5\,m$.

The flux line plot and the results of the force computation for this model can be found in Fig. 9(a) and in Fig. 10, respectively.

The x-component of the force on the left/right wire F_x can be shown analytically to be approximately equal to $\mp 1.92 \cdot 10^{-7}\,N$, whereas $F_y = 0.0\,N$. Fig. 10(a-b) and (c-d) depict the computed F_x and F_y values, respectively. An additional check on the computed force values is the verification that the sum of F_x on the left and right wire equals zero. In Fig. 10(e-f) these sums are plotted. From Fig. 10(a),(c) and (e), showing the results of the Lorentz, the Virtual Work, the *eggshell* and the MST methods, we conclude that for first order elements the methods converge equally fast to the correct value. We remark that for these methods the initial mesh is not sufficiently fine to obtain a good approximation to the analytical solution. At least four refinement steps are necessary to obtain the first two significant digits of the solution. Although Fig. 10(a) shows that the MST method give more accurate results on coarse meshes, Fig. 10(c) and (e) demonstrate that the aforementioned methods outperform the MST method on finer meshes even for linear discretizations.

By comparing the scale of the y-axis in the left and right column of Fig. 10, we observe that for second order elements the initial mesh already provides two significant digits of the solution. From Fig. 10(d) and (f), a faster convergence can be seen for the Lorentz method. Fig. 10(d) shows that F_y computed by the Lorentz method is smaller than $10^{-15}\,N$ on the finest mesh. For the *eggshell* method it remains at least two orders of magnitude larger depending on the Gaussian rule considered. Fig. 10(f) shows that on the finest mesh the difference between the Lorentz and *eggshell* method (`rule=3`) is smaller. By studying this model problem only, it is difficult to draw conclusions concerning the different Gaussian rules. It is, however, clear that `rule=-3` gives the least satisfactory results.

By comparing the left and right column, we may conclude that our second order *eggshell* implementation outperforms previously existing implementations of the Virtual Work and the MST methods for the force computation on a conductor.

6.2 Second Model Problem

In the second model problem we replace the right wire in the first model problem by a permanent magnet magnetized in the y-direction. The geometrical and physical properties of the left wire remain unchanged. The magnet has a circular cross section with the same radius as the wire and is assumed to be non-permeable. The value for the pre-magnetization $M_y = 10 A/m$ is chosen such that the F_x value is comparable to that in the first model problem. Numerical results make clear that, in order to obtain accurate force results, it is necessary to increase the radius of the computational domain. Details of the second model problem are given by

radius wire and magnet	$20\,cm$
center of wire/magnet	$(\mp 0.5\,m, 0.0\,m)$
current density (J_0)	$1/\pi r^2\,A/m^2$
current density in wire (J_z)	J_0
pre-magnetization (M_x, M_y)	$(0\,A/m, 10\,A/m)$
radius of computational domain	$40\,m$.

The flux line plot and the results of the force computation for this model can be found in Fig. 9(b) and in Fig. 11, respectively.

Fig. 11(a-b) and (c-d) show the computed F_x and F_y values on the magnet. Both the F_y values and the sum of the F_x values on the wire and the magnet should vanish in the limit for small meshes. We conclude from Fig. 11(a) and (c) that, for the computation of the force on the magnet, the Virtual Work and the *eggshell* methods perform equally well for first order elements, whereas the MST method converges slower.

On the coarsest mesh, the second order *eggshell* method with `rule=3` and `rule=6` gives two significant digits of the solution of F_x on the magnet and it converges almost monotonically for smaller mesh widths as shown in Fig. 11(b). With `rule=-3` the *eggshell* method provides a good solution on the coarsest grid, but gives a larger error on the once refined mesh. For the other methods, the initial mesh appears to be not fine enough as illustrated by Fig. 11(a). Fig. 11(d) shows that for the computation of F_y on the magnet, the *eggshell* method (`rule=6`) comes close to a value of $6 \cdot 10^{-13}\,N$. With the rules `rule=3` and `rule=-3` the values obtained on the finest mesh remain one and two orders of magnitude larger. The values of the sum of the F_x components computed by the second order *eggshell* method on the finest mesh lie between $10^{-10}\,N$ and $3 \cdot 10^{-10}\,N$ depending on the Gaussian rules employed. This value is larger than for the previous model problem, and is still reasonably large compared to the value of $-2.5 \cdot 10^{-7}\,N$ for the force on the magnet.

By comparing the left and right column in Fig. 11, we may conclude that the *eggshell* method for second order elements outperforms previously existing implementations of the Virtual Work and the MST methods for the force computation on a permanent magnet.

6.3 Third Model Problem

In the third model problem we consider a configuration of four current carrying wires with circular cross section and equal radius. The wires are placed such that their centers form a square with its center in the origin. The top right and bottom left conductors carry a current of $1\,A$ while the top left and bottom right conductors carry $-1A$. Details of the third model problem are given by

radius of wires (r)	$20\,cm$
center of wire in first/second quadrant	$(\pm 0.5\,m, 0.5\,m)$
center of wire in third/fourth quadrant	$(\mp 0.5\,m, -0.5\,m)$
current density (J_0)	$1/\pi r^2\,A/m^2$
current density in first/third wire (J_z)	J_0
current density in second/fourth wire (J_z)	$-J_0$
radius of computational domain	$5\,m$.

The flux line plot and the results of the force computation for this model can be found in Fig. 9(c) and in Fig. 12, respectively.

Fig. 12(a-b) and (c-d) show the computed F_x values on the wire in the first and second quadrant, respectively. The sum of the force on the four wires should converge to zero. The x-component of this sum is shown in Fig. 12(e-f).

This model problem corroborates the conclusions drawn from the first model problem. Again, for this problem the Lorentz method is the method of choice. The *eggshell* method, however, outperforms previously existing Virtual Work and MST implementations. Fig. 12(b) and (d) show that the approximation error in the force obtained on coarse meshes with the Gaussian rule `rule=-3` is larger than the one obtained by the other two Gaussian rules. This effect can also be observed in Fig. 10(b) and in Fig. 11(b). The present results confirm the now expected result that the Gaussian rule `rule=6` gives the most reliable results.

6.4 Fourth Model Problem

In the fourth model problem, we replace the dipole formed by the two bottom wires in the third model problem by a permanent magnet with a rectangular cross section magnetized in the y-direction. The value of the pre-magnetization is chosen such that the value of F_x on the wires is comparable in size to that in the third model problem. Details of the fourth model problem are given by

radius of wires (r)	$20\,cm$
center of wire in first/second quadrant	$(\pm 0.5\,m, 0.5\,m)$
current density (J_0)	$1/\pi r^2\,A/m^2$
current density wire in first/second quadrant (J_z)	$\pm J_0$
size of magnet	$0.9\,m \times 0.4\,m$
center of magnet	$(0\,m, -0.5\,m)$
pre-magnetization (M_x, M_y)	$(0\,A/m, 2.5\,A/m)$
radius of computational domain	$5\,m$.

The flux line plot and the results of the force computation for this model can be found in Fig. 9(d) and in Fig. 13, respectively.

Fig. 13(a-b), showing the F_x values on the wire in the first quadrant, closely resembles Fig. 12(a-b), showing these values in the third model. Similarly, Fig. 13(c-d) with F_x values on the wire in the second quadrant are good copies of both Fig. 10(a-b) and Fig. 12(c-d). Although the F_x values differ in magnitude, we observe a comparable convergence behavior for each of the methods. The results of the Virtual Work and the first order *eggshell* methods on a given mesh are similar. The initial mesh is too coarse for both methods, but sufficiently fine for the second order MST method. For the second order *eggshell* method, the Gaussian rule `rule=-3` results in the poorest performance compared to the rules `rule=3` and `rule=6`. For the first model, we compared the results of *eggshell*

(`rule=3` and `rule=6`) with those obtained by the Lorentz method, and we concluded that for the F_x values both methods are performing equally well. For the fourth model problem, the Lorentz method cannot compute the force on the magnet, and, therefore, we find the *eggshell* method (`rule=3` and `rule=6`) to be the best choice for this kind of problems.

In Fig. 13(e-f) the F_y values on the permanent magnet are plotted instead of the F_x values as in Fig. 11(a-b). As the x-components of the force in the wires neutralize each other, the force in the x-direction is close to zero. The convergences behavior of F_y on the magnet is a good equivalent of those of the F_x shown in Fig. 11(a-b). This model problem thus gives a nice summary of the previous models. In Fig. 13(g) the sum of F_y remains above 10^{-10}, whereas in Fig. 13(h) it drops below this value, an indication that second order methods should be recommended in practise.

7 Conclusions

The aim of this work was to investigate the efficiency and the accuracy of the recently introduced *eggshell* method for magnetic force computations. We did so by comparing the performance of the method with the Lorentz, the Virtual Work and the MST methods. As model problems we considered four configurations of conductors and permanent magnets immersed in two-dimensional perpendicular current stationary magnetic fields. The models were discretized by first and second order isoparametric triangular finite elements on adaptively constructed meshes of triangles. All models were constructed in such a way that several checks on the computed solution are available.

Our conclusions can be summarized as follows. In all model problems, the MST method was the slowest to converge. Therefore, we do not favor the use of this method. For computing the force on conductors, the Lorentz method is the method of choice. For this application, the Lorentz, Virtual Work and *eggshell* methods on first order discretizations converge equally fast to the solution. The former method, however, is conceptually simpler and performs better than the other methods for second order discretizations. For computing the force on permanent magnets (for which the Lorentz method no longer applies), the Virtual Work and *eggshell* methods converge equally fast on first order discretizations. On second order discretizations, we obtained promising results for the *eggshell* method, whereas no implementation of the Virtual Work method was available in the simulation code used.

Both the finite element field simulation and the force computation by the classical methods was performed in the FEMLAB simulation environment. Our implementation of the *eggshell* method, as described in Section 5.2, was included in the same simulation environment. The second order isoparametric discretization extends the magnetic force computation capabilities of the FEMLAB package.

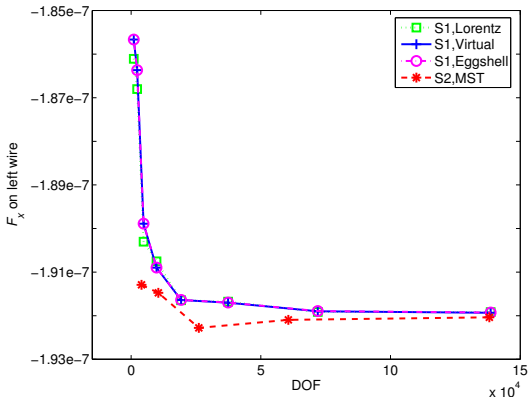
Acknowledegement

The authors would like to thank F. Henrotte for insightful discussions on the *eggshell* method, the FEMLAB support team for valuable suggestions on implementations issues and P. W. Hemker for his helpful advice during the course of this research. This research is supported by the Dutch Ministry of Economical Affairs within the project IOP-EMVT 02201.

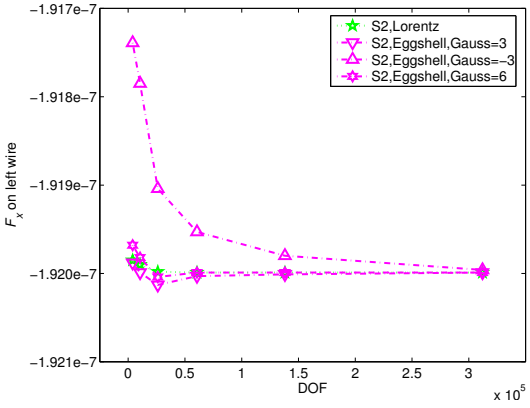
References

[1] A. Bossavit. Edge-Element Computation of the Force Field in Deformable Bodies. *IEEE Trans. Magn.*, 28(2):1263–1266, 1992.

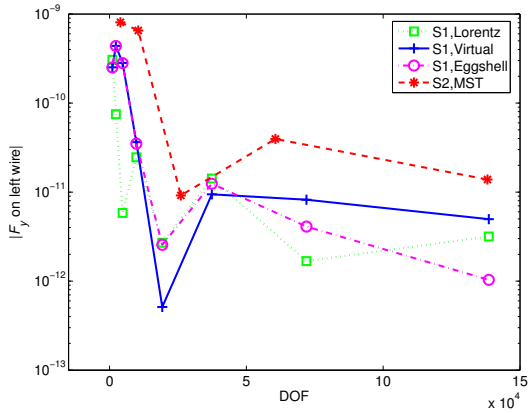
[2] A.C. Felippa. *Advanced Finite Element Methods (ASEN 5007).* Department of Aerospace Engineering Sciences, University of Colorado at Boulder, 2004. http://titan.colorado.edu/courses.d/IFEM.d.



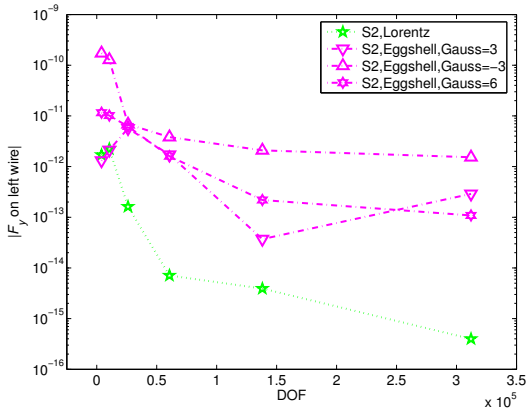
(a) F_x on left wire



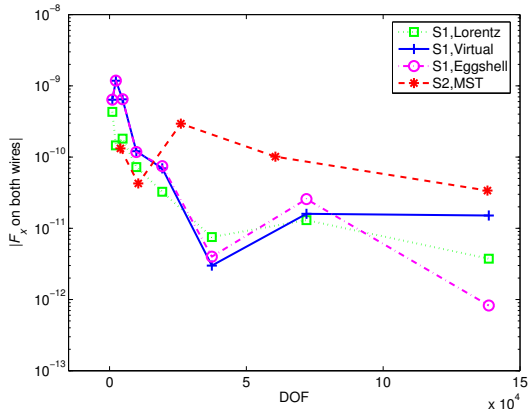
(b) F_x on left wire



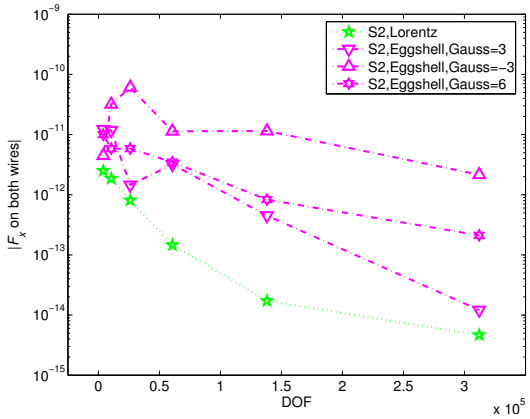
(c) $|F_y|$ on left wire



(d) $|F_y|$ on left wire

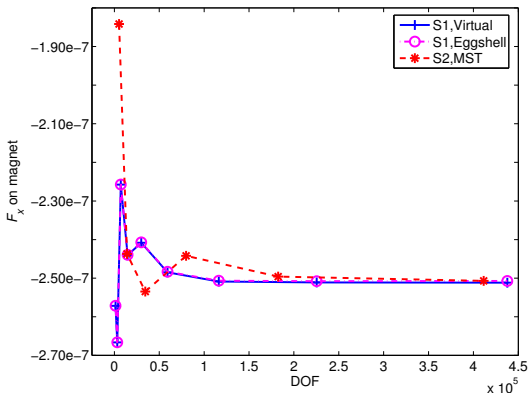


(e) $|\sum_{i=1}^{2} F_{x,i}|$

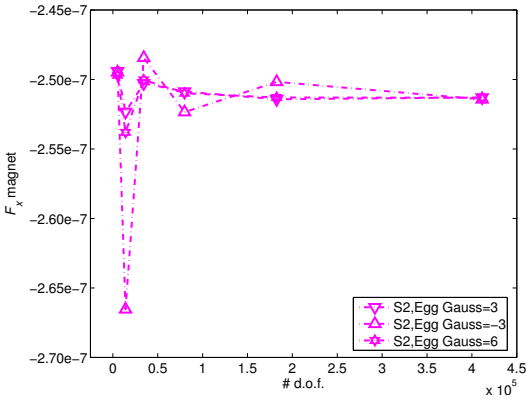


(f) $|\sum_{i=1}^{2} F_{x,i}|$

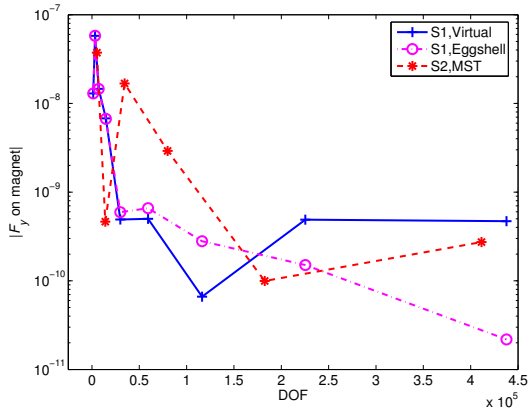
Figure 10: *model 1*: electromagnetic forces on two parallel current carrying wires



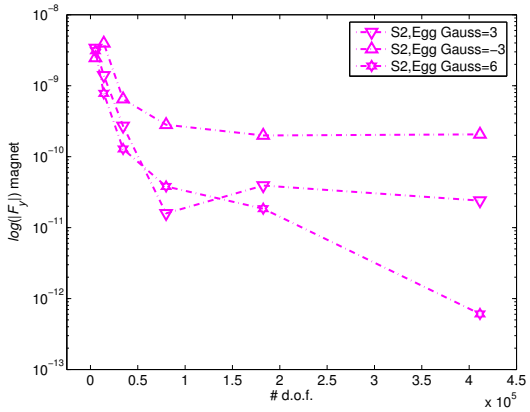
(a) F_x on magnet



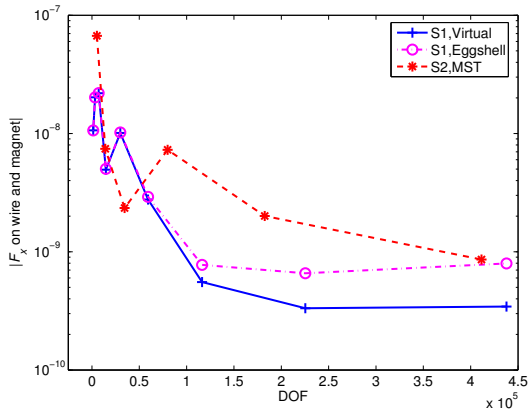
(b) F_x on magnet



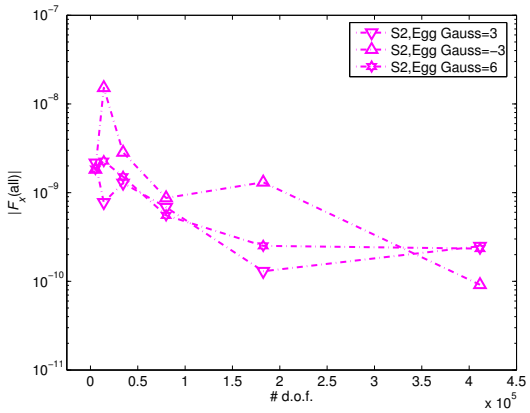
(c) $|F_y|$ on magnet



(d) $|F_y|$ on magnet

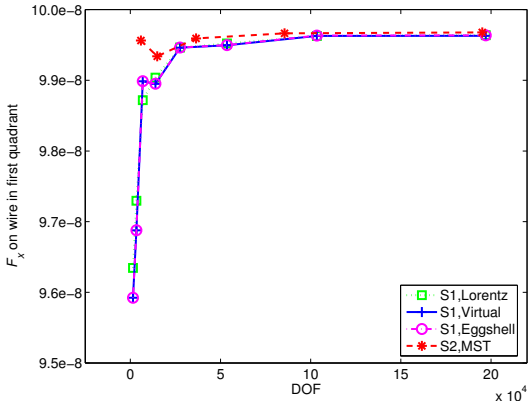


(e) $|\sum_{i=1}^{2} F_{x,i}|$

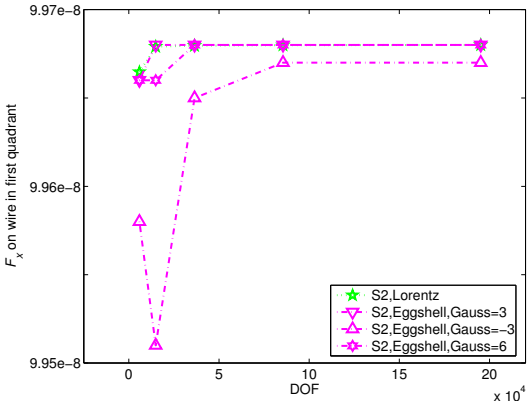


(f) $|\sum_{i=1}^{2} F_{x,i}|$

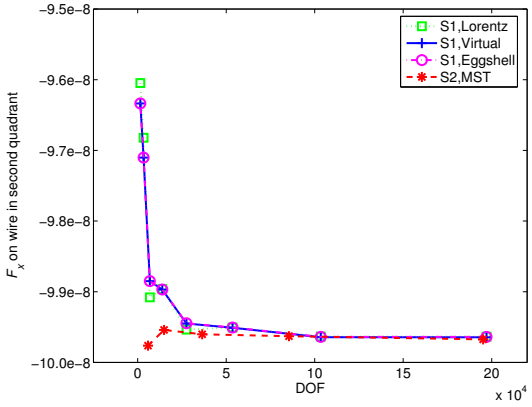
Figure 11: *model 2* : electromagnetic forces on one wire and one permanent magnet, magnetized in the y-direction



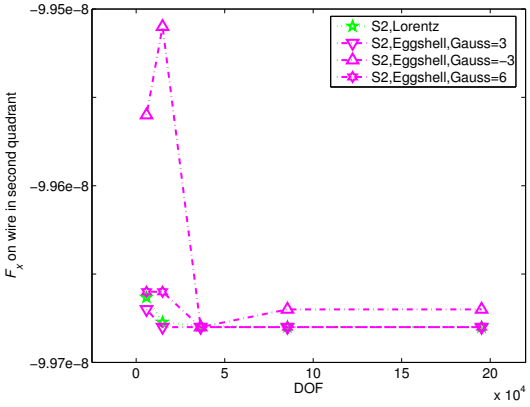
(a) F_x on wire in first quadrant



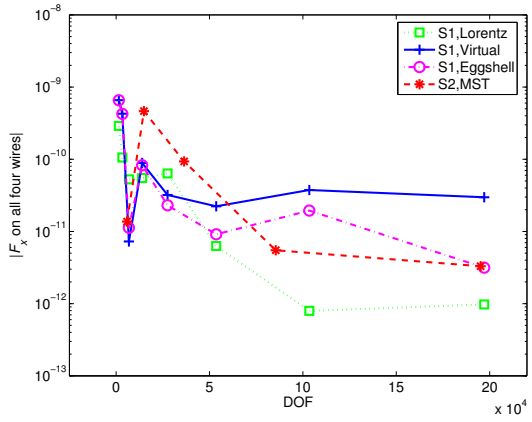
(b) F_x on wire in first quadrant



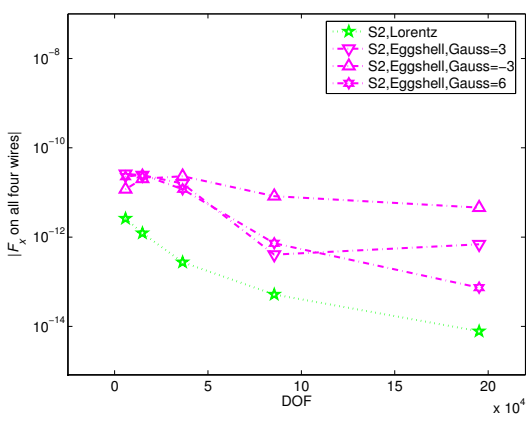
(c) F_x on wire in second quadrant



(d) F_x on wire in second quadrant

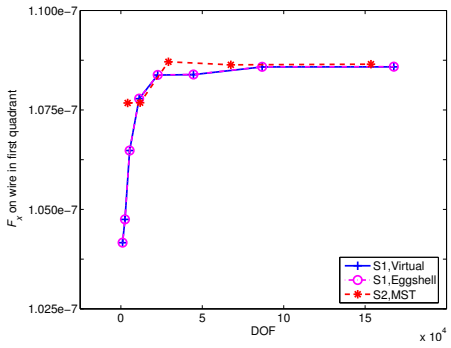


(e) $|\sum_{i=1}^{4} F_{x,i}|$

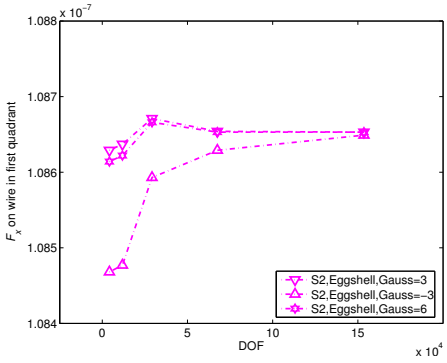


(f) $|\sum_{i=1}^{4} F_{x,i}|$

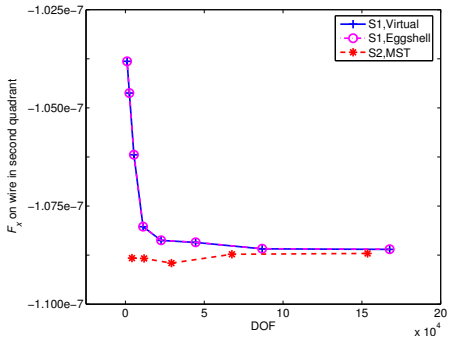
Figure 12: *model 3*: four parallel current carrying wires. The wires are placed such that their centers form a square centered around the origin



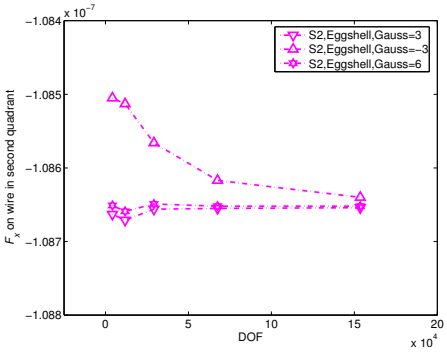
(a) F_x on wire in first quadrant



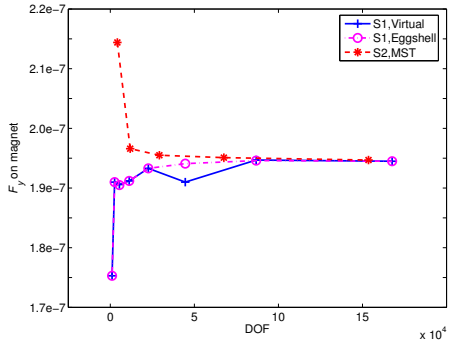
(b) F_x on wire in first quadrant



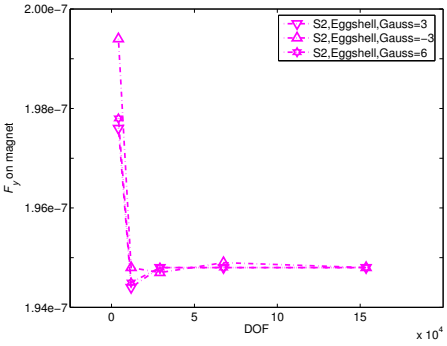
(c) F_x on wire in second quadrant



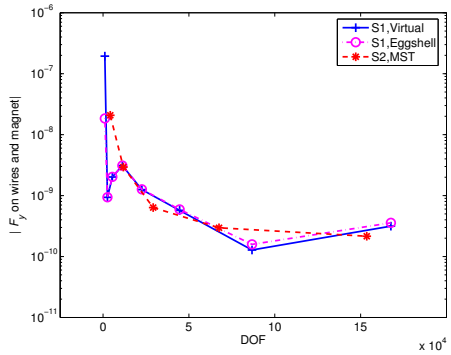
(d) F_x on wire in second quadrant



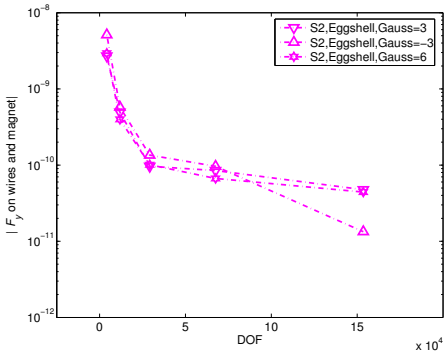
(e) F_y on magnet



(f) F_y on magnet



(g) $|\sum_{i=1}^{3} F_{y,i}|$



(h) $|\sum_{i=1}^{3} F_{y,i}|$

Figure 13: *model 4*: two parallel current carrying wires and a permanent magnet. The permanent magnet has a rectangular cross section and is magnetized in the y-direction

[3] FEMLAB. *FEMLAB, Version 2.2, Reference Manual*, November 2001.

[4] FEMLAB. *FEMLAB, Version 3.0, Electromagnetics Module, Model Library*, January 2004.

[5] FEMLAB. *FEMLAB, Version 3.0, Users Guide*, January 2004.

[6] G. Henneberger, Ph. K. Sattler, and D. Shen. Force Calculation with Analytical Accuracy in the Finite Element Based Computational Magnetostatics. *IEEE Trans. Magn.*, 27(5):4254–4247, 1991.

[7] F. Henrotte, G. Deliège, and K. Hameyer. The Eggshell Approach for the Computation of Electromagnetic Forces in 2D and 3D. *COMPEL*, 23:996–1005, 2004.

[8] F. Henrotte and K. Hameyer. Computation of Electromagnetic Force Densities: Maxwell Stress Tensor vs. Virtual Work Principle. *J. Comput. Appl. Math.*, 168:235–243, 2004.

[9] F. Henrotte, H Vande Sande, G. Deliège, and K. Hameyer. Electromagnetic Force Density in a Ferromagnetic Material. *IEEE Trans. Magn.*, 40:553–556, 2004.

[10] K. Komeza, A. Pelikant, J. Tegopoulos, and S. Wiak. Comparative Computation of Forces and Torques of Electromagnetic Devices by Means of Different Formulae. *IEEE Trans. Magn.*, 30(5):3475–3478, 1994.

[11] J. L. Lions and E. Magenes. *Nonhomogeneous Boundary Value Problems and Applications.* Springer-Verlag, Berlin, 1972.

[12] L.H. De Medeiros, G. Reyne, and G. Meunier. Comparison of Global Force Calculation on Permanent Magnets. *IEEE Trans. Magn.*, 34(5):3560–3563, 1998.

[13] W. Müller. Comparison of Different Methods of Force Calculation. *IEEE Trans. Magn.*, 26(2):1058–1061, 1990.

[14] Z. Ren. Comparison of Different Force Calculation Methods in 3D Finite Element Modelling. *IEEE Trans. Magn.*, 30(5):3471–3474, 1994.

[15] G. Reyne, Sabonnadière, J. L. Coulomb, and P. Brissonneau. A Survey of the Main Aspects of Magnetic Forces and Mechanical Behaviour of Ferromagnetic Materials Under Magnetisation. *IEEE Trans. Magn.*, 23(5):3765–3767, 1987.

[16] P. P. Sylvester and R. L. Ferrari. *Finite Elements for Electrical Engineers.* Cambridge University Press, New York, third edition, 1996.

[17] J. P. Webb. An Estimator for Force Errors in Finite-Element Analysis. *IEEE Trans. Magn.*, 39(3):1428–1431, 2003.

[18] H. H. Woodson and J. R. Melcher. *Electromechanical Dynamics I, II and III.* John Wiley, 1968.

[19] O.C. Zienkiewics and R. L. Taylor. *The Finite Element Method: Volume 1 The Basics.* Butterworth-Heinemann, fifth edition, 2000.