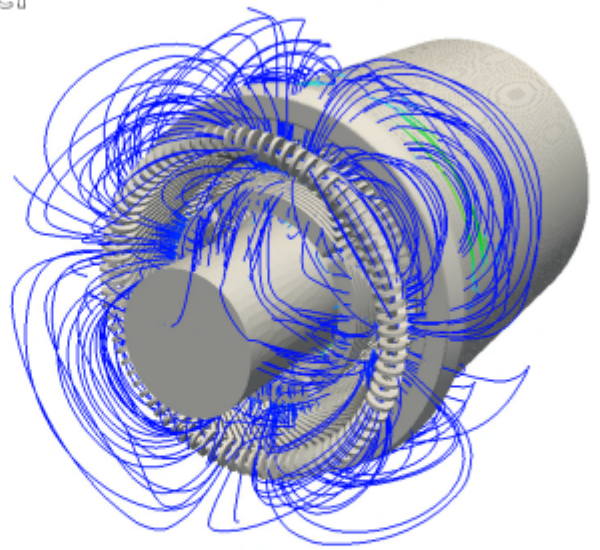
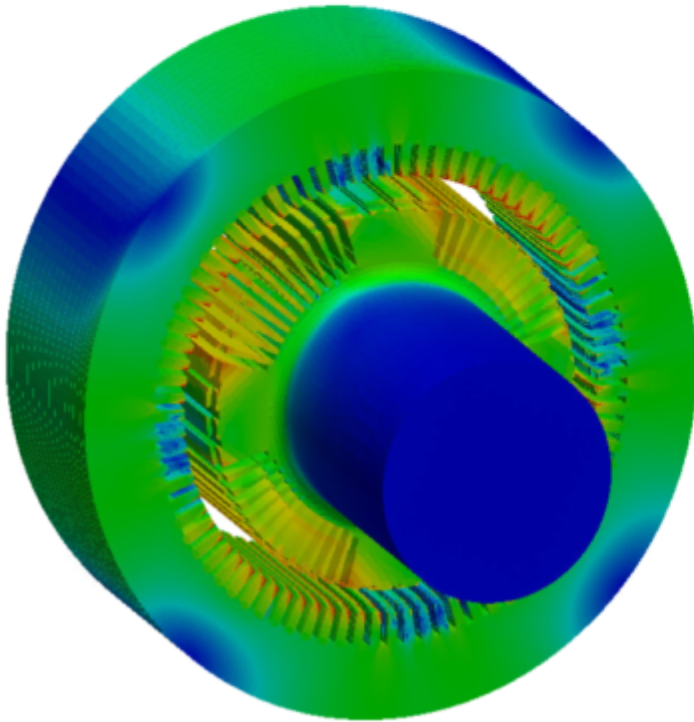


Principles of the time-based and multi-harmonic versions of code_Carmel

LAMEL

2022/12/01

code_Carmel



Contents

Foreword	xii
Introduction	xiii
Nomenclature	xv
 I Specific modelling of electromagnetism problems	 1
1 Formation of the equations	3
1.1 Definition of the problem	3
1.2 Maxwell's equations	4
1.3 Quasistatic states assumption	6
1.3.1 Electroquasistatic model	7
1.3.2 Magnetoquasistatic model	8
1.3.3 Choice of model	9
1.4 Partial differential equations in the continuous domain	9
1.4.1 Magnetodynamic problem	9
1.4.2 Magnetostatic problem	10
1.4.3 Electrokinetic problem	10
1.5 Electrical and magnetic constitutive relations of the media	11
1.5.1 Electrical conductivity	11
1.5.2 Magnetic permeability	12
1.5.2.1 Ferromagnetic materials	12
1.5.2.2 Magnets	13
1.6 Crossing conditions at media interfaces	13
1.7 Boundary conditions	14
 2 Formulation of potential equations	 17
2.1 Electrokinetic problem	17
2.1.1 Reminder of the equations	17
2.1.2 Magnetic formulation φ	18
2.1.3 Electrical formulation \mathbf{T}	19
2.2 Magnetostatic problem	20
2.2.1 Reminder of the equations	20
2.2.2 Vector magnetic potential formulation \mathbf{A}	20
2.2.3 Scalar magnetic potential formulation Ω	21
2.3 Magnetodynamic problem	22
2.3.1 Reminder of the equations	22
2.3.2 Electrical formulation $\mathbf{A} - \varphi$	22
2.3.3 Magnetic formulation $\mathbf{T} - \Omega$	23

3	Imposition of global quantities	25
3.1	Introduction of K and N fields	25
3.2	Introduction of the function α et du champ β	26
3.3	Electrokinetics	27
3.3.1	Vector electric potential formulation \mathbf{T}	27
3.3.1.1	Imposition of the current	27
3.3.1.2	Imposition of the voltage	28
3.3.2	Scalar electric potential formulation φ	28
3.3.2.1	Imposition of the voltage	28
3.3.2.2	Imposition of the current	29
3.3.3	Review of imposing overall values in electrokinetics	29
3.4	Magnetostatics	29
3.4.1	Formulation A	29
3.4.1.1	Imposition of a flux	29
3.4.1.2	Imposition of a magnetic potential difference	30
3.4.2	Formulation in Ω	31
3.4.2.1	Imposition of a flux	31
3.4.2.2	Imposition of a magnetic potential difference	31
3.4.3	Review of imposing overall values in magnetostatics	31
3.5	Magnetodynamics	32
3.5.1	Formulation A - φ	32
3.5.1.1	Imposition of a voltage in a wound conductor	32
3.5.1.2	Imposition of a flux and of a voltage in a solid conductor	33
3.5.1.3	Imposition of a magnetomotive force and an electric current in a solid conductor	33
3.5.2	Formulation T - Ω	34
3.5.2.1	Imposition of a magnetomotive force and an electric current in a solid conductor	34
3.5.2.2	Imposition of a flux and a voltage	34
4	Dealing with regions that are not simply connected	37
4.1	Electrokinetics	37
4.2	Magnetostatics	39
5	Weak form of the equations	41
5.1	Function spaces	41
5.1.1	Definitions	41
5.1.2	Property of continuous function spaces	42
5.1.3	Electromagnetic fields	43
5.1.4	Potential	43
5.2	Projection principles	44
5.3	Magnetodynamic problem	45
5.3.1	Formulation A - φ	45
5.3.1.1	Projection in space only	45
5.3.1.2	Projection in space and time	47
5.3.2	Formulation T- Ω	48
5.3.2.1	Projection in space only	48
5.3.2.2	Projection in space and time	50
5.4	Magnetostatic problem	51
5.4.1	Formulation A	51
5.4.1.1	Projection in space only	51
5.4.1.2	Projection in space and time	52
5.4.2	Formulation Ω	52
5.4.2.1	Projection in space only	52

5.4.2.2	Projection in space and time	52
5.5	Electrokinetic problem	53
5.5.1	Formulation φ	54
5.5.2	Formulation T	54
6	Coupling with external circuits	55
6.1	Breakdown of the source current	55
6.2	Circuit equation	56
6.2.1	Expression for the magnetic flux	56
6.2.2	Formulation of the electrical problem	56
6.2.3	Mesh current method	57
6.2.4	Method for calculating the tree of the electrical circuit	59
6.3	Coupling solid conductors in spectral version	60
II	Overview of space and time discretisation	63
7	Discretisation spaces	65
7.1	Interpolation spaces	65
7.1.1	Overview	65
7.1.2	Shape functions	65
7.1.2.1	Nodal function	65
7.1.2.2	Edge function	66
7.1.2.3	Facet function	67
7.1.2.4	Volume function	67
7.1.3	Discrete spaces	68
7.1.4	Potentials	68
7.2	Discrete differential operators	68
7.2.1	The discrete gradient G_{an}	69
7.2.2	The discrete curl R_{fa}	69
7.2.3	The discrete divergence D_{vf}	70
7.2.4	The dual mesh concept	70
7.2.5	Properties of the operators	71
7.3	Properties of interpolation spaces	71
7.4	Discretisation of fields and potentials	73
8	Source terms and global quantities	75
8.1	Introduction of a gauge (edge and facet trees)	75
8.1.1	Value of trees	75
8.1.2	Construction of a facet tree	77
8.2	Discretisation of K and N	80
8.2.1	Discretisation of N	81
8.2.2	Discretisation of K	83
8.3	Discretisation of α et β	83
8.4	Discretisation of the current density of a wound inductor	84
8.4.1	Introduction	84
8.4.2	Discretisation using a Whitney complex	84
8.4.2.1	Incidence matrix	85
8.4.2.2	Mass matrix	85
8.4.3	Use of the facet tree	86
8.4.4	Inversion of the co-tree matrix	86
8.4.5	Application to an elbow of circular cross-section	86
8.4.6	Conclusion	88
8.5	Imposition of a uniform current	88

8.5.1	Use of a guideline	88
8.5.2	Case of a constant cross-section	88
8.5.2.1	Description of the method	88
8.5.2.2	Illustration of the principle	90
8.5.2.3	Implementation of the academic facet tree	90
8.5.2.4	Obtaining the academic facet tree	91
8.5.2.5	Use of the academic facet tree	91
8.5.2.6	The minimisation method	92
8.5.2.6.1	The divergence matrix	92
8.5.2.6.2	The minimisation matrix	93
8.5.2.6.3	Taking account of boundary conditions	95
8.5.2.6.4	Inversion of the co-tree matrix	96
8.5.2.7	Calculation of the mass matrix and of right member	98
8.6	Case of non-constant cross-section	98
8.6.1	Geometry	99
8.6.2	Calculation of the current density	99
8.6.3	Use of the facet tree	100
8.6.4	Minimisation method	101
9	Discretisation of weak forms	103
9.1	Discrete function spaces	103
9.1.1	Approximation of $H^1(\mathcal{D})$	103
9.1.2	Discrete approximation of $H(\mathbf{rot}, \mathcal{D})$	104
9.1.3	Discrete approximation of $H(\mathbf{div}, \mathcal{D})$	105
9.1.4	Discrete approximation of $L^2(\mathcal{D})$	106
9.1.5	Taking account of <i>ad hoc</i> boundary conditions	106
9.2	Electrokinetic problem	107
9.2.1	Formulation φ with imposed voltage	107
9.2.2	Formulation φ with imposed current	108
9.2.3	Formulation \mathbf{T}	108
9.3	Magnetostatic problem	109
9.3.1	Projection in space only	109
9.3.1.1	Formulation \mathbf{A}	109
9.3.1.2	Formulation Ω	109
9.3.2	Projection in space and time	110
9.4	Magnetodynamic problem	110
9.4.1	Projection in space only	110
9.4.1.1	Formulation $\mathbf{A} - \varphi$	110
9.4.1.2	Formulation $\mathbf{T} - \Omega$	111
9.4.2	Projection in space and time	113
9.4.2.1	Differentiation in the spectral domain	114
9.4.2.2	Formulation $\mathbf{A} - \varphi$	114
9.4.2.3	Formulation $\mathbf{T} - \Omega$	117
9.5	Time discretisation	119
9.5.1	Weak form discretisation	119
9.5.2	Magnétodynamique	120
9.5.2.1	Formulation $\mathbf{A} - \varphi$	120
9.5.2.2	Formulation $\mathbf{T} - \Omega$	121
9.6	Equations with overall values	121
9.6.1	Case of an imposed voltage on a wound conductor	121
9.6.2	Case of a surface insulator	122
9.7	Resolution of discrete problems	123
9.7.1	Generic matrix notation	123
9.7.2	Time discretisation	123

9.7.2.1	Time discretisation of the magnetic equation	123
9.7.2.2	Time discretisation of the mechanical equation	123
9.7.2.3	Time discretisation of the generic problem	124
III	Construction of the matrix system	125
10	Implementation of finite element method	127
10.1	Finite elements used	127
10.2	Reference elements and shape functions used	128
10.2.1	Case of the tetrahedron	128
10.2.1.1	Finite element P_1 de classe H^1	128
10.2.1.2	Finite element of class \mathbf{H}_{rot}	129
10.2.1.3	Finite element of class \mathbf{H}_{div}	130
10.2.1.3.1	Case of the prism	131
10.2.1.3.2	Case of the hexahedron	134
10.2.2	Case of the pyramid	137
10.2.2.1	Nodal shape functions	138
10.2.2.2	Edge shape functions	139
10.2.2.3	Facet shape functions	140
10.2.2.3.1	Hiptmair approach	140
10.2.2.3.2	Whitney approach	141
10.2.2.3.3	Comparison of the two types of function	142
10.2.3	Transformation of the reference element into a real element (Calculating the integral)	142
10.2.4	Calculation of elementary integrals by the Gauss method	143
10.2.4.1	Case of triangles	143
10.2.4.2	Case of rectangles	143
10.2.4.3	Case of tetrahedra	144
10.2.4.4	Case of prisms	146
10.2.4.5	Case of hexahedra	146
10.2.4.6	Case of pyramids	148
11	Taking motion into account	149
11.1	General principle	149
11.2	Blocked step method	151
11.2.1	Mesh layout with the blocked step method	151
11.2.2	Finite element problem on \mathcal{D}_R et \mathcal{D}_S	151
11.2.3	Motion equation for \mathcal{D}_R et \mathcal{D}_S	152
11.2.4	Notation of the total system with the blocked step method	152
11.2.5	Conclusion	153
11.3	Overlapping method	153
11.3.1	Mesh layout with the overlapping method	154
11.3.2	Extension of nodal shape functions to \mathcal{D}_θ	154
11.3.3	Overlapping reference element	154
11.3.4	Dealing with edge unknowns	156
11.3.5	Notation of the total system with the overlapping method	156
11.4	Specific method for the spectral version	157
11.4.1	Principle of the blocked step	157
11.4.2	Spectral representation of motion	158
11.5	Kinematic coupling	160
11.5.1	Formation of the equation of the physical problem	160
11.5.2	Treatment	161
11.5.3	Weak coupling of the magnetic equation and mechanical equation	161

12 Processing non-linearity	163
12.1 Fixed point	163
12.1.1 Description of the method	163
12.1.2 Approximate method and solutions of the fixed point	164
12.1.3 Study of convergence	164
12.1.4 Advantages and disadvantages	165
12.2 Newton-Raphson	165
12.2.1 Description of the method	166
12.2.2 Study of convergence	166
12.2.3 Magnetostatic example	166
12.2.4 Advantages and disadvantages	168
12.3 Solving non linear problem	168
12.3.1 Numerical resolution by the fixed point method	169
12.3.2 Numerical resolution by the Newton-Raphson method	170
12.3.3 Overall solution method	171
12.3.4 Magnetostatic matrix system	171
12.3.4.1 Fixed point method for the vector magnetic potential formulation	171
12.3.4.2 Newton's method for the vector magnetic potential formulation	172
12.3.4.3 Fixed point method for the scalar electric potential formulation	174
12.3.4.4 Newton's method for the scalar magnetic potential formulation	174
12.3.5 Magnetodynamic matrix system	174
12.3.5.1 Fixed point method for the vector magnetic potential formulation	174
12.3.5.2 Newton's method for the vector magnetic potential formulation	175
12.4 Solving non linear problem	176
13 Numbering the unknowns	177
13.1 General numbering principle	177
13.2 Numbering for the time-based version of code_Carmel	177
13.2.1 Electrokinetics	177
13.2.1.1 Formulation φ	177
13.2.1.2 Formulation \mathbf{T}	178
13.2.2 Magnetostatics	178
13.2.2.1 Formulation \mathbf{A}	178
13.2.2.2 Formulation Ω	178
13.2.3 Magnetodynamics	179
13.2.3.1 Formulation $\mathbf{A} - \phi$	179
13.2.3.2 Formulation $\mathbf{T} - \Omega$	179
13.2.4 Numbering for the spectral version of code_Carmel	180
13.3 Dealing with floating potentials	180
13.4 Dealing with boundary conditions	180
13.5 Dealing with periodicity conditions	180
14 Assembly	181
14.1 General assembly principle	181
14.2 Magnetodynamic overall matrix - Harmonic case	181
14.2.1 Vector magnetic potential formulation	181
14.2.1.1 Spectral approach to the scalable non-linear problem	182
14.2.1.1.1 Resolution using the Galerkin projection	183
14.2.1.1.2 Tensor notation	183
14.2.2 Scalar electric potential formulation	186
14.3 Electrokinetic overall matrix	187
14.3.1 Formulation φ with imposed voltage	188
14.3.2 Formulation φ with imposed current	188
14.3.3 Formulation \mathbf{T}	188

14.4	Magnetostatic overall matrix - Time-based case	189
14.4.1	Vector magnetic potential formulation	189
14.4.1.1	Linear magnetostatic vector magnetic potential	189
14.4.1.2	Non-linear magnetostatic vector magnetic potential	189
14.4.2	Scalar magnetic potential formulation	190
14.5	Magnetostatic overall matrix - Harmonic case	191
14.5.1	Vector magnetic potential formulation	191
14.5.2	Scalar magnetic potential formulation	191
14.6	Magnetodynamic overall matrix - Time-based case	191
14.6.1	Vector magnetic potential formulation	191
14.6.2	Scalar magnetic potential formulation	193
14.7	Processing overall values	193
14.7.1	Magnetodynamics	193
14.7.1.1	Imposing a voltage on a coiled conductor	193
14.8	Coupling with an external circuit	193
14.8.1	Breakdown of the source current	193
14.8.2	Circuit equation	194
14.8.3	Expression for the magnetic flux	194
14.8.4	Strong coupling of the magnetic equation with circuit equations	195
14.9	Dealing with domains that are not simply connected	196
IV	Resolution of the matrix system	197
15	Resolution of the linear system	199
15.1	Overview of linear systems	199
15.1.1	Calculation costs for field physics simulations	199
15.1.2	Two families of methods to resolve a linear system	201
15.1.3	Solutions offered by Code_Carmel	201
15.2	Conjugate gradient (CG) type iterative methods	202
15.2.1	Principle	202
15.2.1.1	Positioning of the problem	202
15.2.1.2	Steepest Descent	204
15.2.1.3	Principle of the conjugate gradient	205
15.2.1.4	Conjugate gradient algorithm	206
15.2.2	Preconditioned conjugate gradient (PCG)	207
15.2.2.1	Principes	207
15.2.2.2	PCG algorithm	209
15.2.2.3	PCG in code_Carmel	210
15.2.3	Range of preconditioners available in code_Carmel	211
15.2.3.1	Jacobi preconditioner	212
15.2.3.2	Crout preconditioner	213
15.2.3.3	MUMPS preconditioner	214
15.3	Direct methods	216
15.3.1	Principle	216
15.3.1.1	Factorisation	216
15.3.1.2	Down- up	216
15.3.2	The various approaches	217
15.3.3	Main steps	219
15.3.4	Main difficulties	220
15.3.5	The MUMPS product	221
15.3.5.1	History	221
15.3.5.2	Main characteristics of MUMPS	223
15.3.5.3	Advantages and specific features	224

15.3.5.3.1	Pivoting	224
15.3.5.3.2	Iterative refinement	225
15.3.5.3.3	Reliability of the calculations	226
15.3.5.3.4	Memory management	228
15.3.5.3.5	Management of singular matrices	229
15.3.6	Implementing MUMPS in code_Carmel	229
15.3.6.1	Version compatibility and copyright	229
15.3.6.2	Ergonomic choices	230
15.3.6.3	Code_Carmel parameters to use MUMPS	231
15.3.6.4	MUMPS warnings and error reporting	232
15.4	Organisation of calculations with MUMPS	232
15.4.1	Initialisation	232
15.4.2	Filling	233
15.4.3	Calculation steps	233
15.4.4	Cleaning	234
V	Post-processing	235
16	Force calculations	237
16.1	Maxwell stress tensor method	237
16.1.1	Principle	237
16.1.2	Discretisation	238
16.2	Virtual work method	240
16.2.1	Principle	240
16.2.2	Discretisation	240
16.2.2.1	Local derivative of the magnetic energy	240
16.2.2.2	Local derivative of the magnetic co-energy	241
16.2.2.3	Derivative of the Jacobian matrix	242
17	Calculating local magnetic flux	245
17.1	Introduction	245
17.2	Presentation of the problem	245
17.3	Case of formulation \mathbf{A}	245
17.4	Case of formulation Ω	246
17.4.1	First approach	246
17.4.2	Second approach	247
17.4.3	Third approach	248
18	Calculation of iron losses	249
18.1	Magnetic materials	249
18.1.1	Magnetic values	249
18.1.2	Classification of magnetic materials	251
18.1.2.1	Diamagnetism	251
18.1.2.2	Paramagnetism	251
18.1.2.3	Ferromagnetism	251
18.1.3	Configuration in magnetic domains	252
18.1.3.1	Weiss domains	252
18.1.3.1.1	Anisotropy energy	252
18.1.3.1.2	Magnetostatic energy	253
18.1.3.2	Bloch walls	254
18.1.4	Magnetisation process - First magnetisation curve	254
18.2	Magnetic losses	255
18.2.1	Hysteresis losses	256

18.2.2	Induced current losses	256
18.2.3	Anomalous losses	258
18.2.4	Rotational field losses	259
18.3	Description of the iron loss calculation procedure	259
19	Exploratory points	263
19.1	Search method	263
19.1.1	Nodal function method	264
19.1.2	Jacobian matrix method	264
19.1.3	Barycentric coordinate method	265
19.2	Tetrahedra	266
19.2.1	Nodal function method	267
19.2.2	Jacobian matrix method	267
19.2.3	Barycentric coordinates method	267
19.2.4	Proof of the equivalence of the last two methods	268
19.3	Prisms	269
19.3.1	Nodal function method	269
19.3.2	Jacobian matrix method	271
19.3.3	Barycentric coordinates method	272
VI	Appendixes	285
A	Reference documents	287
B	The quasi steady-state approximation (QSSA)	289
B.1	Analysis of time constants	289
C	U.w gauge condition	291
D	Incorporation of Overlapping elements into code_Carmel	293
D.1	Presentation of the Overlapping element	293
D.1.1	Reference element	293
D.1.2	Nodal shape functions	295
D.1.3	Edge shape functions	295
D.1.4	Gauss points	296
E	Taking account of non-linearity	299
E.1	Non-linear constitutive relation	299
E.2	Calculation of the Jacobian	299
E.3	Breakdown of operators into linear and non-linear parts	301
F	Discrete model from incidence matrices	303
F.1	Discrete differential operators	303
F.1.1	Node-edge incidence	303
F.1.2	Edge-facet incidence	304
F.1.3	Facet-element incidence	305
F.1.4	Properties	305
F.2	Dual mesh	306
F.2.1	Définitions	306
F.2.2	Properties	308
F.3	Discrete Maxwell's equations	308
F.4	Discretisation of the constitutive relations	309
F.5	Discrete formulations	310
F.5.1	Current density discretisation	311

F.5.2	Magnetodynamic problem	313
F.5.2.1	Electrical formulation A - φ	314
F.5.2.2	Electrical formulation T - Ω	314
F.5.3	Magnetostatic problem	315
F.5.4	Electrokinetic problem	315
F.6	Time discretisation	316
G	Determination of fields of given curl or divergence	317
G.1	Edge tree	317
G.2	Facet tree	319
H	Formulation A - φ	323
I	Finding the element containing a point in code_Carmel	325
J	Libraies of linear algebra	327
J.1	Expression of needs	327
J.1.1	Management of loss of control	327
J.1.2	A wide range of linear algebra libraries	328
J.2	Annex: Theoretical supplements	329
J.2.1	Krylov spaces	329
J.2.2	Orthogonality	330
J.2.3	Convergence	331
J.2.4	Computation and memory costs	332
J.3	Annex: Non-linear resolution strategies	332
J.3.1	Construction of the preconditioner	332
K	MUMPS copyright	335
L	Moving from real element to reference element	337
L.1	Case of the tetrahedron	337
L.2	Nodal approximation function	337
L.3	Edge approximation functions	338
L.4	Transformation of derivatives	338
L.5	Transformation of integrals	338
M	Add-ins for force and torque calculation	339
M.1	Maxwell stress tensor	339
M.1.1	General case	339
M.1.2	Two-dimensional case	340
M.2	Virtual work method	340
M.2.1	Derivative of the magnetic energy (vector potential formulation	340
M.2.2	Derivative of the magnetic co-energy (scalar potential formulation	341
M.2.3	Calculation of the derivative of matrix \mathbf{J}'	342
M.2.4	Two-dimensional case	342
N	Development using orthogonal polynomials	343
N.1	Généralités	343
N.2	Legendre polynomials	344
N.3	Chebyshev polynomials	345
N.4	Development using the Fourier basis	346
O	Kronecker product	349
P	Hadamard product	351

<i>CONTENTS</i>	xi
Q Modified Gauss quadrature	353
Index	355

Foreword

Controlling the operating behaviour of electrical machines – transformers, alternators, motors, etc. – is a major concern for EDF and for electrical machine manufacturers and operators in general.

These devices must meet precise specifications in normal operation when they are first commissioned. But the equipment changes over its lifetime and the operating constraints can change (the Grid Code in 2017, for example). Thus, it is often useful to be able to assess their behaviour under abnormal conditions (new specifications) or exceptional conditions (faults).

These concerns apply to a very wide range of “equipment”:

- cylindrical rotor generators;
- salient pole rotor generators;
- induction motors;
- transformers;
- diagnostic instruments;
- electromagnetic compatibility;
- effects of the magnetic field on the human body, etc.

For a long time, functional analysis was essentially based on tests and calculations applied to simple geometries. Today, in addition, electromagnetic modelling provides a powerful means of investigation to better understand the problems encountered. The modelling approach first consists in defining a set of equations to locally describe the electromagnetic field. These are based on Maxwell’s equations coupled with laws describing the behaviour of materials. These equations are then formatted so that proven techniques can be applied to solve them. Finally, the results are processed so that they can be expressed in terms of familiar electrical engineering variables.

The Lille Laboratory of Electrical Engineering and Power Electronics (L2EP) and the ERMES department (formerly THEMIS) of EDF R&D are jointly developing code_Carmel. This is a software package for three-dimensional calculation of electromagnetic fields based on the finite element method. It is particularly suited to the study of electrical machines under transient conditions (in its time-based version) or in the steady state (in its multi-harmonic version).

Hence, more particularly, this document aims to summarise the spectral approaches dedicated to the specific resolution of transient electromagnetism problems, with motion and with random parameters. In particular, it formalises general expressions for the spectral representation of the time dimension by considering not only a harmonic basis (suited to periodic problems) but also a polynomial basis (for the processing of non-periodic variables). In addition, it allows for general application of the Spectral Stochastic Finite Element Method (SSFEM) to take into account uncertainties in the constitutive relations for linear magnetoharmonic problems with motion.

This document provides a detailed presentation of the equations processed by the code_Carmel software and how to solve them using a finite element method. This is not a textbook on electromagnetism or the finite element method. It assumes basic knowledge of electromagnetic phenomena and numerical methods in general.

Some notational conventions should be specified by way of introduction. A vector field is set in bold type. For example, \mathbf{B} represents the flux density vector field throughout the domain under study.

Introduction

To study the internal behaviour of the electromagnetic structure of an electrical device, we have used numerical modelling [Vérité et al 2007]. The modelling consists in establishing a mathematical structure that describes the physical phenomena. The mathematical model is formed of Maxwell's equations, which include Ampère's circuital law, Faraday's law, and Gauss's laws for magnetism and electricity, associated with the constitutive relations of the various media and the boundary conditions.

The resolution of such a model consists in identifying changes in the magnetic and electric fields in space and over time. The finite element method is generally used to model complex systems. Space and time discretisation of the domain under study is thus carried out. The magnetic and electric fields are thus represented on the mesh elements. To this end, the University of Science and Technology of Lille has designed code_Carmel.

EDF R&D wished to better master its tools for the calculation of electromagnetic fields. The code_Carmel software package was chosen and it was decided to jointly develop it in partnership with the University of Lille 1 within the Electrical Equipment Modelling Laboratory (LAMEL).

The reference methods for solving a magnetodynamic problem are step-by-step integration methods over time. They are robust and easy to implement. Nevertheless, their high precision is obtained at the cost of calculation times that can be very long, thus reducing their scope of application.

They are all the more time-consuming given that the values of interest are calculated over several periods to reach the steady state. The principle of spectral approaches consists in representing the operator(s) of the physical system as a linear combination of predefined functions (for which the operator(s) are easy to calculate). An approximation of the solution sought is thus constructed on the basis of carefully chosen functions of finite and relatively small size. Spectral methods (or multi-harmonic methods) are suitable tools for this purpose.

This document has chiefly been drafted based on the existing bibliography within LAMEL (the list is given in Annex A). This bibliography is supplemented by more specific references where necessary.

This document describes the operating principles of code_Carmel software in its time-based and multi-harmonic versions (restrictions to one version or the other are indicated in the text). It describes the electromagnetic equations used, their discretisation to enable use of the finite element method, and the methods used to solve the mathematical problems involved.

With a nomenclature to standardise notation of the symbols used, the document has been divided into five parts:

1. The specific modelling of the equations to be solved (the physical equations and consideration of global variables and/or motion, etc.) ;
2. Time and space discretisation;
3. Construction of the matrix system;
4. Solution of the linear and/or non-linear problem;
5. Specific use of results such as exploratory points or iron losses.

Nomenclature

Notation related to the continuous domain

$\mathbf{a} \cdot \mathbf{b}$	Scalar product of vectors \mathbf{a} et \mathbf{b} (contracted product)
$\mathbf{a} \times \mathbf{b}$	Vector product of vectors \mathbf{a} et \mathbf{b}
$\mathbf{H}(\mathbf{rot}, \mathcal{D})$	Function space whose curl belongs to $\mathbf{L}^2(\mathcal{D})$
$\mathbf{H}_0(\mathbf{rot}, \mathcal{D})$	Function space $\mathbf{H}(\mathbf{rot}, \mathcal{D})$ satisfying a homogeneous Dirichlet boundary condition on the tangential component
$\mathbf{L}^2(\mathcal{D})$	Square-summable vector function space defined on \mathcal{D}
div	Divergence operator
Γ	Domain boundary \mathcal{D} ($\partial\mathcal{D}$)
Γ_B	Domain boundary \mathcal{D} where conditions are imposed of the form $\mathbf{n} \cdot \mathbf{B} = 0$
Γ_c	Conductive domain boundary \mathcal{D}_c ($\partial\mathcal{D}_c$)
Γ_E	Domain boundary \mathcal{D} where conditions are imposed of the form $\mathbf{n} \wedge \mathbf{E} = 0$
Γ_H	Domain boundary \mathcal{D} where conditions are imposed of the form $\mathbf{n} \wedge \mathbf{H} = 0$
Γ_i^s	Boundary of inductor i
Γ_J	Domain boundary \mathcal{D} where conditions are imposed of the form $\mathbf{n} \cdot \mathbf{B} = 0$
$\langle \cdot \cdot \rangle_{\mathcal{D}}$	Scalar product on $\mathcal{D} : (\mathbf{x}, \mathbf{y}) \mapsto \langle \mathbf{x} \mathbf{y} \rangle_{\mathcal{D}} = \int_{\mathcal{D}} \mathbf{x} \cdot \mathbf{y}$
\mathbf{dl}	Unit vector at a tangent to a curve
grad	Gradient operator
\mathbf{n}	Unit vector normal to a surface
rot	Curl operator
\mathcal{D}	Space domain under study
\mathcal{D}_s^i	Wound or bar type inductor (where the source current is imposed)
\mathcal{D}_c	Conductive domain
\mathcal{D}_s	Source domain (all windings or bars) $\mathcal{D}_s = \cup_i \mathcal{D}_s^i$
\mathcal{D}_{nc}	Non-conductive or insulating domain ($\mathcal{D} \setminus \mathcal{D}_c$)

\mathcal{T}	Time domain under study
$H(\text{div}, \mathcal{D})$	Function space $L^2(\mathcal{D})$ whose divergence belongs to $L^2(\mathcal{D})$
$H^1(\mathcal{D})$	Sobolev space of scalar functions whose derivative belongs to $L^2(\mathcal{D})$
$H_0^1(\mathcal{D})$	Sobolev space of functions belonging to $H^1(\mathcal{D})$ satisfying a homogeneous Dirichlet boundary condition
$H_0(\text{div}, \mathcal{D})$	Function space $H(\text{div}, \mathcal{D})$ satisfying a homogeneous Dirichlet boundary conditions on the normal component
$L^2(\mathcal{D})$	Square-summable scalar function space defined on \mathcal{D}
$ \cdot $	Absolute value
$\ \cdot\ _2$	Euclidean norm
$\ \cdot\ _{L^2(\mathcal{D})}$	Norm $L^2(\mathcal{D})$ induced by the scalar product $\langle \cdot \cdot \rangle_{\mathcal{D}} : \mathbf{x} \mapsto \langle \mathbf{x} \mathbf{x} \rangle_{\mathcal{D}}^{1/2}$
$H_{0,x}(\mathbf{grad}, \mathcal{D})$	Function space $H(\mathbf{grad}, \mathcal{D})$ satisfying a homogeneous Dirichlet boundary condition for the value of the function on Γ_x
S_i	Cross-section of inductor i

Notation related to the discrete domain

Γ_h	Boundary of \mathcal{D}_h ($\partial\mathcal{D}_h$)
\mathbf{w}_i^1	Vector interpolation function associated with edge 'i'
\mathbf{w}_i^2	Vector interpolation function associated with facet 'i'
\mathcal{D}_h	Discretised domain under study (set of volume elements)
\mathcal{E}_h	Set of edges
\mathcal{F}_h	Set of faces
\mathcal{N}_h	Set of nodes
\mathcal{W}^0	Space of dimension n_0 of vectors containing all values at the nodes
\mathcal{W}^1	Space of dimension n_1 of vectors containing all circulation values on the edges
\mathcal{W}^2	Space of dimension n_2 of vectors containing all flux values across the facets
\mathcal{W}^3	Space of dimension n_3 of vectors containing all values associated with the elements
$\underline{\mathbf{D}}$	Matrix $n_3 \times n_2$ of element-facet incidence
$\underline{\mathbf{G}}$	Matrix $n_1 \times n_0$ of edge-node incidence
$\underline{\mathbf{M}}$	Mass matrix
$\underline{\mathbf{R}}$	Matrix $n_2 \times n_1$ of facet-edge incidence
n_0	Number of nodes in mesh M
n_1	Number of edges in mesh M

n_2	Number of facets in mesh M
n_3	Number of elements in mesh M
W^0	Scalar function space of dimension n_0 generated by node interpolation functions
w_i^0	Scalar interpolation function associated with node 'i'
W^1	Vector function space of dimension n_1 generated by edge interpolation functions
W^2	Vector function space of dimension n_2 generated by facet interpolation functions
W^3	Scalar function space of dimension n_3 generated by element interpolation functions

Electromagnetic fields

$\mathbf{B}(\mathbf{x}, t)$	Magnetic flux density (T)
$\mathbf{D}(\mathbf{x}, t)$	Electric induction (C/m^2)
$\mathbf{E}(\mathbf{x}, t)$	Electric field (V/m)
$\mathbf{H}(\mathbf{x}, t)$	Magnetic field (A/m)
$\mathbf{J}(\mathbf{x}, t)$	Current density (A/m^2)
$\mathbf{J}_{ind}(\mathbf{x}, t)$	Induced current density (A/m^2)
$\rho(\mathbf{x}, t)$	Electric charge density (C/m^3)
ρ_i	Charge in element 'i' (C)
$\underline{\rho}$	Vector (1xn3) containing all charges ρ_i
\underline{b}	Vector (1xn2) containing all fluxes b_i
\underline{d}	Vector (1xn2) containing all fluxes d_i
\underline{e}	Vector (1xn1) containing all circulations e_i
\underline{h}	Vector (1xn1) containing all circulations h_i
\underline{j}	Vector (1xn2) containing all fluxes j_i
b_i	Flux of the magnetic flux density vector through facet 'i' (Wb)
d_i	Electric induction flux through facet 'i' (C)
e_i	Circulation of the electric field along edge 'i' (V)
h_i	Circulation of the magnetic field along edge 'i' (A)
j_i	Current density flux through facet 'i' (current through facet 'i' in A)

Source fields

α_i	Value of the function α at node 'i'
------------	--------------------------------------------

α	Source scalar function
$\mathbf{H}_s(\mathbf{x}, t)$	Source field (A/m)
$\mathbf{K}(\mathbf{x}, t)$	Normalised source field (A)
$\mathbf{N}(\mathbf{x}, t)$	Normalised source field such that $\mathbf{rot}\mathbf{K} = \mathbf{N}$ (A/m ²)
$\mathbf{J}_s(\mathbf{x}, t)$	Source current density (A/m ²)
$\underline{\alpha}$	Vector (1xn0) of values of α at node 'i'
\underline{h}	Vector (1xn1) containing all circulations h_{si}
\underline{k}	Vector (1xn1) containing all circulations k_i
\underline{n}	Vector (1xn2) containing all circulations n_i
h_{si}	Circulation of the source field along edge 'i' (A)
k_i	Circulation of the normalised source field along edge 'i'
n_i	Flux of normalised source field N of coils through facet 'i' (A)

Potentials

$\mathbf{A}(\mathbf{x}, t)$	Vector magnetic potential (Wb/m)
$\mathbf{A}_h(\mathbf{x}, t)$	Finite Element approximation of the vector magnetic potential (Wb/m)
$\mathbf{T}(\mathbf{x}, t)$	Vector electric potential (A/m)
$\mathbf{T}_h(\mathbf{x}, t)$	Finite Element approximation of the vector electric potential (A/m)
$\Omega(\mathbf{x}, t)$	Scalar magnetic potential
$\Omega_h(\mathbf{x}, t)$	Finite Element approximation of the scalar magnetic potential
Ω_i	Value of the scalar magnetic potential at node 'i'
$\underline{\Omega}$	Vector (1xn0) containing all node values Ω_i
$\underline{\varphi}$	Vector (1xn0) containing all node values φ_i
\underline{a}	Vector (1xn1) containing all circulations a_i
\underline{t}	Vector (1xn1) containing all circulations k_i
$\varphi(\mathbf{x}, t)$	Scalar electric potential (V)
$\varphi_h(\mathbf{x}, t)$	Finite Element approximation of the scalar electric potential (V)
φ_i	Value of the scalar electric potential at node 'i' (V)
a_i	Circulation of the vector magnetic potential along edge 'i' (Wb)
t_i	Circulation of the vector electric potential along edge 'i' (A)

Overall values and circuit coupling

ϕ	Flux across a surface (Wb)
ξ	Magnetic potential difference (A)
a_{cir}	Number of branches in the circuit
b_{cir}	Number of independent loops in the circuit
J_{cir}	Fictitious current in a loop of the external circuit
KM	Branch–mesh incidence matrix
n_{cir}	Number of nodes in the circuit
U_C	Capacitive dipole voltage vector
U_L	Inductive dipole voltage vector
U_R	Resistive dipole voltage vector
U_S	Source voltage vector
I	Electric current (A)
V	Electric potential difference (V)

Constitutive relations

$B_r(\mathbf{x}, t)$	Remanent flux density (T)
$H_c(\mathbf{x}, t)$	Coercive field (A/m)
μ	Permeability ($H.m^{-1}$)
μ_0	Vacuum permeability ($4\pi 10^{-7} H.m^{-1}$)
μ_a	Magnetic permeability of a magnet ($H.m^{-1}$)
μ_r	Relative permeability of a medium
σ	Electrical conductivity ($\Omega^{-1}.m^{-1}$)
$\underline{\mathbf{b}}_r$	Vector (1xn2) containing all fluxes b_{ri}
$\underline{\mathbf{h}}$	Vector (1xn1) containing all circulations h_{ci}
ε	Permittivité
ε_0	Vacuum permittivity
ε_r	Relative permittivity of a medium
b_{ri}	Flux of the remanent flux density through facet 'i' (Wb)
h_{ci}	Circulation of the coercive field along edge 'i' (A)

Other physical variables

λ	Wavelength (m)
\mathbf{x}	position
ω	Angular frequency ($rad.s^{-1}$)
f	frequency (Hz)
r_i	Resistance of wound inductor 'i'
T	Study duration in s
t	time (s)

Finite elements

w_i^1	i ^{ème} basic function conforming with $\mathbf{H}(\mathbf{rot}, \mathcal{D})$
w_i^2	i ^{ème} basic function conforming with $H(\mathbf{div}, \mathcal{D})$
Jac	Jacobian matrix
u, v, w	Coordinates of a point in the reference element coordinate system
w_i^0	i ^{ème} basic function P^1 conforming with $H^1(\mathcal{D})$
x, y, z	Coordinates of a point in space in the Cartesian coordinate system (O, i,j,k)
A	total number of edges
a	global edge number
E	total number of elements
e	global element number
F	total number of facets
f	global facet number
K	A geometric element of the mesh
N	total number of nodes
n	global node number

Others

Alpha	Displacement step ratio
-------	-------------------------

Part I

Specific modelling of electromagnetism problems

Chapter 1

Formation of the equations

Summary

This chapter defines the problems studied by specifying the mathematical equations governing these models (magnetodynamic, magnetostatic and electrokinetic). Details are also given of the sub-domains relevant to the model, the conditions for crossing from one sub-domain to another and the domain boundary conditions.

1.1 Definition of the problem

In the following, we consider an electrotechnical system (see Figure 1.1) composed of air, ferro-magnetic materials and/or conductors and magnetic field sources (wound or non-wound inductors, and/or permanent magnets)¹.

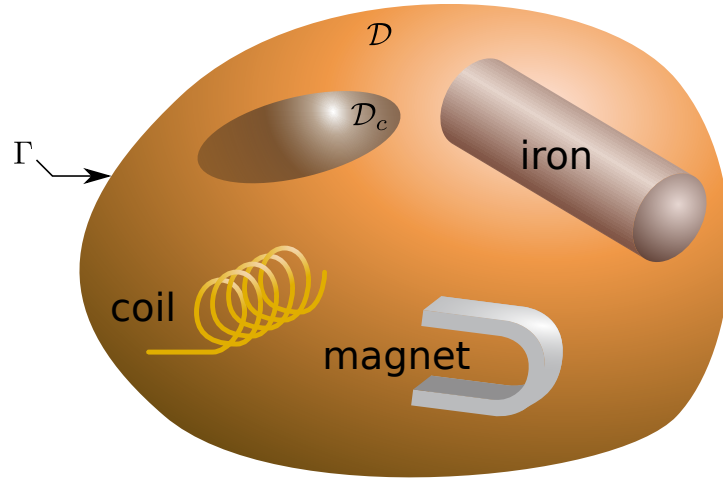


Figure 1.1: Schematic breakdown of the domain under study \mathcal{D}

The entire system forms the domain under study \mathcal{D}^2 with boundary Γ . It is composed of:

- the conducting media, of domain \mathcal{D}_c with boundary Γ_c . This is the domain where eddy currents are created;
- a non-conducting medium \mathcal{D}_{nc} .

¹A wound inductor is a domain in which the current is imposed and is uniform across the domain

² \mathcal{D} is an open set of \mathbb{R}^3

Domain \mathcal{D}_{nc} is made up of, for example:

- sources: wound or solid inductors carrying a current distribution \mathbf{J}_s , permanent magnets (if they are assumed to be non-conductive);
- ferromagnetic materials;
- air (of magnetic permeability μ_0).

The sources (regions of the domain where a source current density \mathbf{J}_s is imposed) define a sub-domain \mathcal{D}_s ³.

Remark 1.1.1 *If the conductivity in the ferromagnetic materials, magnets and coils is not neglected, the corresponding media are to be included in domain \mathcal{D}_c .*

Remark 1.1.2 *The conductors, magnets and ferromagnetic materials may be in contact with each other.*

Remark 1.1.3 *If the system under study has geometric symmetries or periodicity, it is possible to reduce the domain under study \mathcal{D} to only part of the system.*

Remark 1.1.4 *Boundary Γ may coincide with the boundary of a medium other than air.*

Boundary Γ is divided into two portions Γ_B and Γ_H to impose domain boundary conditions (see paragraph 1.7)⁴. As a reminder, Γ_B is the boundary of domain \mathcal{D} where conditions of the form $\mathbf{B} \cdot \mathbf{n} = 0$ are imposed; Γ_H is the boundary of domain \mathcal{D} where conditions of the form $\mathbf{n} \wedge \mathbf{H} = 0$ are imposed; \mathbf{n} is the normal vector leaving the given boundary.

The electromagnetic phenomena are investigated over a time interval \mathcal{T} of between 0 and T seconds:

$$\mathcal{T} = [0, T] \tag{1.1}$$

1.2 Maxwell's equations

The electromagnetic field is defined by four vector fields:

- $\mathbf{D}(\mathbf{x}, t)$: electric induction (C/m²) ;
- $\mathbf{E}(\mathbf{x}, t)$: electric field (V/m) ;
- $\mathbf{H}(\mathbf{x}, t)$: magnetic field (A/m) ;
- $\mathbf{B}(\mathbf{x}, t)$: magnetic flux density (T) ;

These vector fields depend on:

- t: time (s);
- \mathbf{x} : given position.

³Sub-domains \mathcal{D}_c , \mathcal{D}_{nc} and \mathcal{D}_s are included in \mathcal{D} .

⁴We will have: $\Gamma_B \cup \Gamma_H = \Gamma$ and $\Gamma_B \cap \Gamma_H = \emptyset$

Space and time distributions of the magnetic and electric fields are obtained from Maxwell's equations [Bossavit 1993], [Durand 1968], [Fournet 1985], [P  rez et al 1990]. They are thus written:

$$\mathbf{rot} \mathbf{H}(\mathbf{x}, t) = \mathbf{J}(\mathbf{x}, t) + \frac{\partial \mathbf{D}(\mathbf{x}, t)}{\partial t} \quad (\text{Maxwell-Amp  re law}) \quad (1.2)$$

$$\mathbf{rot} \mathbf{E}(\mathbf{x}, t) = - \frac{\partial \mathbf{B}(\mathbf{x}, t)}{\partial t} \quad (\text{Maxwell-Faraday law}) \quad (1.3)$$

$$\text{div} \mathbf{B}(\mathbf{x}, t) = 0 \quad (\text{Gauss's magnetic law}) \quad (1.4)$$

$$\text{div} \mathbf{D}(\mathbf{x}, t) = \rho(\mathbf{x}, t) \quad (\text{Gauss's electric law}) \quad (1.5)$$

with the addition of the four vector fields defined above:

- $\rho(\mathbf{x}, t)$: electric charge density (C/m³) ;
- $\mathbf{J}(\mathbf{x}, t)$: current density (A/m²) ;

Remark 1.2.1 *In this system of equations, 1.2 and 1.3 express the coupling between the electrical and magnetic values.*

Finally, the electric charge conservation equation is added:

$$\text{div} \mathbf{J}(\mathbf{x}, t) + \frac{\partial \rho(\mathbf{x}, t)}{\partial t} = 0 \quad (1.6)$$

The latter equation is implicitly contained in 1.2, 1.3, 1.4 , and 1.5.

The current density \mathbf{J} can be broken down into two terms: \mathbf{J}_s in the case where the inductor is wound, the current density is assumed to be uniform and known, and \mathbf{J}_{ind} in the case of a conductive domain where the current density is unknown.

$$\mathbf{J}(\mathbf{x}, t) = \mathbf{J}_{ind}(\mathbf{x}, t) + \mathbf{J}_s(\mathbf{x}, t) \quad (1.7)$$

This system is supplemented by constitutive relations, depending on the materials modelled.

$$\begin{aligned} \mathbf{J}_{ind} &= \mathcal{S}(\mathbf{E}(\mathbf{x}, t)) \\ \mathbf{H}(\mathbf{x}, t) &= \mathcal{K}(\mathbf{B}(\mathbf{x}, t)) \end{aligned} \quad (1.8)$$

In general, the induced current density is a function of the electric field. The magnetic field is a function of the magnetic flux density. These relations will be detailed in paragraph 1.5.

Verification of the system of equations 1.2, 1.3, 1.4 and 1.5⁵ implies the following continuity conditions when an interface crosses between two media, characterised by its normal \mathbf{n} :

$$\mathbf{E} \times \mathbf{n} = \mathbf{0} \quad (1.9)$$

$$\mathbf{H} \times \mathbf{n} = \mathbf{0} \quad (1.10)$$

$$\mathbf{B} \cdot \mathbf{n} = 0 \quad (1.11)$$

$$\mathbf{D} \cdot \mathbf{n} = 0 \quad (1.12)$$

These crossing conditions will be analysed in paragraph 1.6.

⁵in the sense of the distributions

To be correctly formulated, the problem defined by equations 1.2, 1.3, , 1.4 and 1.5 must be accompanied by domain boundary conditions. Conventionally, we write:

$$(\mathbf{n} \times \mathbf{H})|_{\Gamma_H} = \mathbf{H}^\Gamma \Leftrightarrow (\mathbf{J} \cdot \mathbf{n})|_{\Gamma_H} = \mathbf{J}^\Gamma \quad (1.13)$$

$$(\mathbf{n} \times \mathbf{E})|_{\Gamma_E} = \mathbf{E}^\Gamma \Leftrightarrow (\mathbf{B} \cdot \mathbf{n})|_{\Gamma_E} = \mathbf{B}^\Gamma \quad (1.14)$$

These boundary conditions will be discussed in paragraph 1.7.

Remark 1.2.2 *In time-based code_Carmel, the values of these boundary conditions are zero.*

1.3 Quasistatic states assumption

Solution of Maxwell's equations, as presented in the preceding paragraph, leads to “retarded potential solutions” [Pérez et al 1990]. This signifies that there is a delay between the electromagnetic field at a given point in space and the sources that gave rise to it (see Annex B).

The quasistatic approximation is based on the assumption that the characteristic time scale for changes in the sources (e.g. their period) is very much longer than the time scale for propagation. This can be demonstrated by a dimensional analysis of Maxwell's equations [Cahouet 1992].

Understanding this approach first requires a definition of the concept of the characteristic time of a system [Montier 2018]. This characterises the rate of change of a physical value over time. In other words, it represents the order of magnitude of the time required for a system subjected to disturbance to reach equilibrium. In this presentation, we are interested in so-called low frequency problems, and more particularly quasistatic states, valid when the characteristic time of the system studied τ is very long compared with the propagation time of light in the medium $\tau_{em} = l/c$, where l represents the characteristic length of the system and $c = (\varepsilon\mu)^{-\frac{1}{2}}$ the speed of light in the medium (ε being its electric permittivity and μ its magnetic permeability). By defining the speed of change of the system by $v = l/\tau$, the previous proposition also signifies that we have $v \ll c$.

In this case, it can be considered that a disturbance is transmitted instantaneously throughout the domain, thus making it possible to neglect propagation phenomena. This is called the non-relativistic limit of the model. In practice, this approximation is valid for electrotechnical devices with response frequencies of up to a few hundred kHz.

However, simple dimensional analysis shows that under these assumptions, the Maxwell-Ampère and Maxwell-Faraday equations are not compatible. Indeed, if $|\ast|$ is the order of magnitude of quantity \ast , the Maxwell-Faraday equation gives:

$$\frac{|\mathbf{E}|}{l} \simeq \frac{|\mathbf{B}|}{\tau} \Rightarrow |\mathbf{E}| \simeq v|\mathbf{B}| \quad (1.15)$$

with $v = l/\tau$.

Similarly, dimensional analysis of the Maxwell-Ampère equation without source current gives:

$$\frac{|\mathbf{H}|}{l} \simeq \frac{|\mathbf{D}|}{\tau} \quad (1.16)$$

By adding the linear constitutive relations, this gives:

$$\frac{|\mathbf{B}|}{\mu l} \simeq \frac{\varepsilon|\mathbf{E}|}{\tau} \Rightarrow |\mathbf{E}| \simeq \frac{c^2}{v} |\mathbf{B}| \quad (1.17)$$

Hence the two Maxwell-Ampère and Maxwell-Faraday equations lead to two scale factors between $|\mathbf{E}|$ and $|\mathbf{B}|$, namely v and $\frac{c^2}{v}$. They become compatible in the relativistic case where $v \simeq c$. Hence, the Maxwell-Ampère and Maxwell-Faraday equations produce incompatible scale factors, and one of the two must thus be partially neglected. This choice will be made with particular regard to the order of magnitude of the current source \mathbf{J} and charge source ρ . The two resulting models are the **electroquasistatic** and the **magnetoquasistatic**.

1.3.1 Electroquasistatic model

In the **electroquasistatic** model, the variation in the electric induction field produces a magnetic field, while a fluctuation in the magnetic induction field does not induce an electric field. The Maxwell-Faraday equation is thus no longer valid and is replaced by:

$$\mathbf{rot} \mathbf{E} = \mathbf{0}$$

Dimensional analysis shows that this model is valid when:

$$|\mathbf{J}| \ll |\rho| c \quad (1.18)$$

Hence, Maxwell's equations within the limit of the electric quasistatic approximation are (the values are given a subscript "e" to indicate the electroquasistatic model):

$$\mathbf{rot} \mathbf{H}_e = \mathbf{J}_e + \frac{\partial \mathbf{D}_e}{\partial t} \quad (1.19)$$

$$\mathbf{rot} \mathbf{E}_e = \mathbf{0} \quad (1.20)$$

$$\mathbf{div} \mathbf{B}_e = 0 \quad (1.21)$$

$$\mathbf{div} \mathbf{D}_e = \rho_e \quad (1.22)$$

while that for the charge conservation remains identical:

$$\frac{\partial \rho_e}{\partial t} + \mathbf{div} \mathbf{J}_e = 0 \quad (1.23)$$

In this set of equations, the magnetic induction field \mathbf{B}_e and the magnetic field no longer \mathbf{H}_e appear as source terms (right-hand side). The electric and magnetic equations are thus decoupled. Hence, it is only necessary to solve equations 1.20, 1.22 and 1.23 to find \mathbf{E}_e and \mathbf{D}_e :

$$\begin{aligned} \mathbf{rot} \mathbf{E}_e &= 0 \\ \mathbf{div} \mathbf{D}_e &= \rho_e \\ \frac{\partial \rho_e}{\partial t} + \mathbf{div} \mathbf{J}_e &= 0 \end{aligned}$$

and to reconstruct *a posteriori* the magnetic unknowns \mathbf{B}_e and \mathbf{H}_e using:

$$\begin{aligned} \mathbf{rot} \mathbf{H}_e &= \mathbf{J}_e + \frac{\partial \mathbf{D}_e}{\partial t} \\ \mathbf{div} \mathbf{B}_e &= 0 \end{aligned}$$

1.3.2 Magnetoquasistatic model

Conversely, the **magnetoquasistatic** model is valid when:

$$|\mathbf{J}| \gg |\rho|c \quad (1.24)$$

This makes it possible to neglect the displacement currents $\frac{\partial \mathbf{D}}{\partial t}$ in the Maxwell-Ampère equation. Physically, this approximation implies that a variation in the magnetic induction field produces an electric field while a fluctuation in the electric induction field has no effect on the magnetic field. Within the limit of the magnetic quasistatic approximation, Maxwell's equations become (the values are given the subscript “m” to indicate the magnetoquasistatic model):

$$\mathbf{rot} \mathbf{H}_m = \mathbf{J}_m \quad (1.25)$$

$$\mathbf{rot} \mathbf{E}_m = -\frac{\partial \mathbf{B}_m}{\partial t} \quad (1.26)$$

$$\mathbf{div} \mathbf{B}_m = 0 \quad (1.27)$$

$$\mathbf{div} \mathbf{D}_m = \rho_m \quad (1.28)$$

By assuming $|\mathbf{J}| \gg |\rho|c$, the charge conservation equation is modified and only allows stationary currents:

$$\mathbf{div} \mathbf{J}_m = 0 \quad (1.29)$$

At first glance, the electric unknowns no longer appear as source terms in Maxwell's equations. By analogy with the electroquasistatic model, the electric and magnetic equations can be decoupled (finding \mathbf{B}_m and \mathbf{H}_m from 1.25, 1.27 and 1.29, then reconstructing \mathbf{E}_m and \mathbf{D}_m using 1.26 and 1.28).

However, a problem arises when the system contains a conductive domain in which induced currents are generated. Indeed, the source term in the Maxwell-Ampère equation depends on \mathbf{E} according to the constitutive relation. The four equations are thus coupled in \mathcal{D}_c and it is then question of solving them simultaneously. In summary, the electric and magnetic equations for the magnetoquasistatic model can be decoupled in $\mathcal{D} \setminus \mathcal{D}_c$ and must be considered simultaneously in \mathcal{D}_c .

In the non-conductive domain $\mathcal{D} \setminus \mathcal{D}_c$, equations 1.25, 1.27 and 1.29 are solved initially in order to find \mathbf{B}_m and \mathbf{H}_m :

$$\begin{aligned} \mathbf{rot} \mathbf{H}_m &= \mathbf{J}_m \\ \mathbf{div} \mathbf{B}_m &= 0 \\ \mathbf{div} \mathbf{J}_m &= 0 \end{aligned}$$

before reconstructing fields \mathbf{E}_m and \mathbf{D}_m using 1.26 and 1.28:

$$\begin{aligned} \mathbf{rot} \mathbf{E}_m &= -\frac{\partial \mathbf{B}_m}{\partial t} \\ \mathbf{div} \mathbf{D}_m &= \rho_m \end{aligned}$$

In the conductive domain \mathcal{D}_c , we can find \mathbf{B}_m , \mathbf{H}_m , \mathbf{E}_m and \mathbf{D}_m by simultaneously solving:

$$\begin{aligned} \mathbf{rot} \mathbf{H}_m &= \mathbf{J}_m \\ \mathbf{rot} \mathbf{E}_m &= -\frac{\partial \mathbf{B}_m}{\partial t} \\ \mathbf{div} \mathbf{B}_m &= 0 \\ \mathbf{div} \mathbf{D}_m &= \rho_m \\ \mathbf{div} \mathbf{J}_m &= 0 \end{aligned}$$

Remark 1.3.1 With the constitutive relations for linear and isotropic homogeneous permittivity and conductivity in the conductive domain, the magnetoquasistatic model requires that ρ_m should be zero in \mathcal{D}_c . Indeed, the expression 1.25 implies that $\text{div } \mathbf{J}_m = \sigma_m \text{div } \mathbf{E}_m = 0$ in the conductive domain (as $\text{div } (\text{rot } \mathbf{H}_m) = 0$). Yet $\rho_m = \text{div } \mathbf{E}_m / \varepsilon_m$, hence $\rho_m = 0$ in \mathcal{D}_c . In this case, the Maxwell-Gauss and charge conservation equations become equivalent at $\text{div } \mathbf{E}_m = 0$ and only one of the two need be considered.

1.3.3 Choice of model

In practice, electrotechnical devices have mainly inductive effects with the displacement currents $\frac{\partial \mathbf{D}}{\partial t}$ often negligible. Hence, the **magnetoquasistatic** model is particularly suited to typical electrotechnical applications, and will be used in the remainder of this presentation.

Remark 1.3.2 Reference may be made to [Rapetti, Rousseau 2011] for a more mathematical justification, where the characteristic times of the various electromagnetic phenomena are compared.

From a terminological point of view, two classes of problem can be distinguished within the **magnequasistatic** model:

- the **magnetostatic** problem, when the system does not contain conductive sub-domains ($\mathcal{D}_c = \emptyset$). In this case, the behaviour of the system is *statique* from a *magnétique* point of view: the main equations of the problem 1.25, 1.27 and 1.29 no longer contain time derivative terms. The reader will note that the problem is not purely static, given the term $\frac{\partial \mathbf{B}_m}{\partial t}$ in the Maxwell-Faraday equation 1.26. However, the dynamic is restricted to the electric equations, which are solved *a posteriori* once \mathbf{B}_m and \mathbf{H}_m have been found.
- the **magnetodynamic** problem, if a conductive sub-domain is present in the domain under study ($\mathcal{D}_c \neq \emptyset$). As seen above, all equations in the conductive domain must be taken into account, in particular the Maxwell-Faraday equation, which introduces the term *dynamique* $\frac{\partial \mathbf{B}_m}{\partial t}$

For the sake of clarity, the subscript m will be abandoned in the remainder of this presentation, though the quantities \mathbf{B} , \mathbf{H} , \mathbf{E} , \mathbf{D} , \mathbf{J} and ρ will nevertheless refer to those resulting from the **magnetoquasistatic** model.

1.4 Partial differential equations in the continuous domain

In the types of problem dealt with by code_Carmel, the space and time distributions of the electric fields \mathbf{E} and \mathbf{J} and the magnetic fields \mathbf{B} and \mathbf{H} are sought throughout the domain \mathcal{D} and in a time interval $[0, T]$ (denoted \mathcal{T}).

The current time-based version of code_Carmel is limited to cases magnetodynamics and static electromagnetism (magnetostatic and electrokinetic).

Remark 1.4.1 The spectral version of code_Carmel does not deal with electrokinetics. It allows magnetodynamic and magnetostatic modelling.

1.4.1 Magnetodynamic problem

Given the quasistatic state assumptions, the equations of an electromagnetic problem in the quasistatic state are:

$$\mathbf{rot} \mathbf{H}(\mathbf{x}, t) = \mathbf{J}(\mathbf{x}, t) \quad (1.25)$$

$$\mathbf{rot} \mathbf{E}(\mathbf{x}, t) = -\frac{\partial \mathbf{B}(\mathbf{x}, t)}{\partial t} \quad (1.3)$$

$$\mathbf{div} \mathbf{B}(\mathbf{x}, t) = 0 \quad (1.4)$$

$$\mathbf{div} \mathbf{D}(\mathbf{x}, t) = \rho(\mathbf{x}, t) \quad (1.5)$$

with:

$$\mathbf{div} \mathbf{J}(\mathbf{x}, t) = 0 \quad (1.29)$$

1.4.2 Magnetostatic problem

It is assumed that the phenomena are time invariant. Equations involving magnetic or electric terms are decoupled. Maxwell's equations are then written for the magnetic phenomena:

$$\mathbf{div} \mathbf{B}(\mathbf{x}) = 0 \quad (1.30)$$

$$\mathbf{rot} \mathbf{H}(\mathbf{x}) = \mathbf{J}_s(\mathbf{x}) \quad (1.31)$$

From these equations it can be deduced:

$$\mathbf{div} \mathbf{J}(\mathbf{x}) = 0 \quad (1.32)$$

Remark 1.4.2 *In the special case where motion is involved, the phenomena are no longer time invariant. The assumption then becomes “absence of eddy currents”. In this case, the equations are more accurately written:*

$$\mathbf{div} \mathbf{B}(\mathbf{x}, t) = 0 \quad (1.33)$$

$$\mathbf{rot} \mathbf{H}(\mathbf{x}, t) = \mathbf{J}_s(\mathbf{x}, t) \quad (1.34)$$

and we still have:

$$\mathbf{div} \mathbf{J}(\mathbf{x}, t) = 0 \quad (1.35)$$

1.4.3 Electrokinetic problem

If the domain under study is restricted to the conductive domain \mathcal{D}_c , the equations to be solved are limited to:

$$\mathbf{rot} \mathbf{E}(\mathbf{x}) = 0 \quad (1.36)$$

$$\mathbf{div} \mathbf{J}(\mathbf{x}) = 0 \quad (1.32)$$

Remark 1.4.3 *The spectral version of code_Carmel does not deal with electrokinetic problems.*

1.5 Electrical and magnetic constitutive relations of the media

The electrical and magnetic behaviour of the various media in the domain under study are taken into account by the constitutive relations. These link together the various magnetic and electric fields. These relations involve not only the fields themselves but also variables such as temperature or mechanical stress. These variables will be assumed to be constant in the following. Thus, the relations then strictly depend only on the position considered, the time, and possibly the electromagnetic fields.

In general, in an electrotechnical problem, a physical property characterises a sub-domain of the space domain \mathcal{D} .

1.5.1 Electrical conductivity

The electrical conductivity is generally assumed to be constant for each sub-domain of the space domain \mathcal{D} . A relation of the following form is then obtained for each sub-domain considered:

$$\mathbf{J}(\mathbf{x}, t) = \sigma(\mathbf{x}, \mathbf{E}) \mathbf{E}(\mathbf{x}, t) \quad (1.37)$$

with σ the electrical conductivity in the sub-domain ($\Omega^{-1}.\text{m}^{-1}$).

Remark 1.5.1 *This parameter can be a tensor of $\mathbb{R}^{3 \times 3}$.*

It may be necessary to define an anisotropic conductivity, e.g. to model a thin lamination stack in the form of a single medium or to limit parasitic currents. This is possible since version 1.13.2 of the time-based code, in a vector form that allows definition of the conductivity along the 3 Cartesian axes Ox, Oy and Oz, and also in a tensor form.

The vector form can be used, for example, for Fe-Si laminations in the Oxy plane and stacked in the Oz direction. If we define that the conductivity is 100 times lower along Oz due to the insulators, the conductivity of this medium will be defined as follows: 5.0d7, 5.0d7, 5.0d5. To cover all possible cases, the conductivity can be defined by a 3x3 tensor, its diagonal being equivalent to the vector conductivity.

Remark 1.5.2 *Warning! Zero values for the anisotropic conductivity can only be used in formulation $\mathbf{A} - \varphi$. Aberrant results may be obtained in formulation $\mathbf{T} - \Omega$, as this uses the inverse of conductivity, namely resistivity, which would become partly infinite. The code cannot work with such values.*

Remark 1.5.3 *It should also be noted that excessively high anisotropy values can make it more difficult to obtain results, due to poor numerical conditioning of the system to be solved.*

In the conductive domain \mathcal{D}_c , all fields can be defined. On the other hand, in non-conductive areas ($\sigma = 0$) where the induced currents are zero (only the wound inductor currents \mathbf{J}_s are present), the electric field \mathbf{E} cannot be defined⁶. In these areas, it is thus necessary to solve a magnetostatic problem with partial differential equations of the form:

$$\text{rot } \mathbf{H}(\mathbf{x}, t) = \mathbf{J}_s \quad (1.38)$$

$$\text{div } \mathbf{B}(\mathbf{x}, t) = 0 \quad (1.4)$$

⁶As a result, the electric field \mathbf{E} cannot be uniquely defined as there is then an infinite number of fields \mathbf{E} verifying the constitutive relation 1.37 and the partial differential equation 1.3 (If a vector \mathbf{X} is a solution of 1.3 then any field \mathbf{E} such that $\mathbf{E} = \mathbf{X} + \mathbf{grad } \varphi$ with φ a scalar function of finite value, is also a solution of 1.3 and is further a solution of 1.37 because since $\sigma = 0$, we have $\mathbf{J}_{\text{ind}} = \mathbf{0}$ at any point of \mathcal{D})

1.5.2 Magnetic permeability

For magnetic behaviour, if the material is not ferromagnetic, the model is linear and of the following form, for each sub-domain of the space domain under study \mathcal{D} :

$$\mathbf{B}(\mathbf{x}, t) = \mu_0 \mu_r(\mathbf{x}, \mathbf{H}) \mathbf{H}(\mathbf{x}, t) + \mathbf{B}_r \quad (1.39)$$

with:

- μ_0 the magnetic permeability of air;
- μ_r the relative magnetic permeability (this parameter can be a tensor of $\mathbb{R}^{3 \times 3}$);
- \mathbf{B}_r the remanent magnetic flux density.

1.5.2.1 Ferromagnetic materials

For ferromagnetic materials, relatively complex models can be used that take into account the phenomenon of hysteresis [Johnson 1987]. However, their introduction into numerical models leads to an increase in computation time that may be acceptable in 2D, but is now completely unacceptable in 3D. Hence, for soft ferromagnetic materials, it is preferable to use a relation of the form:

$$\mathbf{B}(\mathbf{x}, t) = \mu_0 \mu_r(\|\mathbf{H}\|_2^2) \mathbf{H}(\mathbf{x}, t) \quad (1.40)$$

with a function μ_r that can be taken from:

- a law such as [Marrocco 1977] (time-based version of code_Carmel):

$$H = \frac{B}{\mu_0} \frac{c B^{2\alpha} + \tau \epsilon}{B^{2\alpha} + \tau} \quad (1.41)$$

where:

- μ_0 is the magnetic permeability of a vacuum;
- α, c, τ and ϵ are characteristic variables of the given magnetic material.
- a spline expression of the constitutive relation $B(H)$ (spectral and time-based versions of code_Carmel);
- from Fröhlich's equation. This anhysteretic model [Swift et al, 2001] was developed by Fröhlich in 1881. The relationship between the flux density and the magnetic field can be presented in two ways:

$$\mu(\|H\|_2) = \frac{\alpha}{1 + \alpha\beta\|H\|_2} \quad (1.42)$$

or

$$\nu(\|B\|_2) = \frac{1}{\alpha - \beta\|B\|_2} \quad (1.43)$$

with $\alpha = \mu_m$ the maximum permeability and $\beta = \frac{\alpha}{\|B_{\text{sat}}\|_2}$ where $\|B_{\text{sat}}\|_2$ is the value of the magnetic flux density at saturation.

Remark 1.5.4 *The spectral version of code_Carmel only has the form with first-order splines (piecewise linear interpolation).*

The magnetic constitutive relation may be rewritten in the form:

$$\mathbf{H} = \nu(\mathbf{B}) \mathbf{B} \quad (1.44)$$

to simplify the calculus of variations. Thus, the value ν (expressed in m.H^{-1}) denotes the magnetic reluctivity and is defined such that:

$$\nu(\mathbf{B}) = (\mu(\mathbf{H}))^{-1} \quad (1.45)$$

1.5.2.2 Magnets

In the case of hard materials (permanent magnets), the phenomenon of remanence is introduced [Chavanne 1988], and a law of the following form is obtained for each sub-domain that includes a permanent magnet:

$$\mathbf{B}(\mathbf{x}, t) = \mu_a \mathbf{H}(\mathbf{x}, t) + \mathbf{B}_r \quad (1.46)$$

with:

- \mathbf{B}_r the remanent magnetic flux density;
- μ_a the magnetic permeability of the magnet, which is assumed to be constant and close to the permeability of air.

1.6 Crossing conditions at media interfaces

The relations described above are valid at points in space where the properties (ε , μ , σ) of the materials are continuous. This is no longer the case at interfaces between different materials (Figure 1.2).

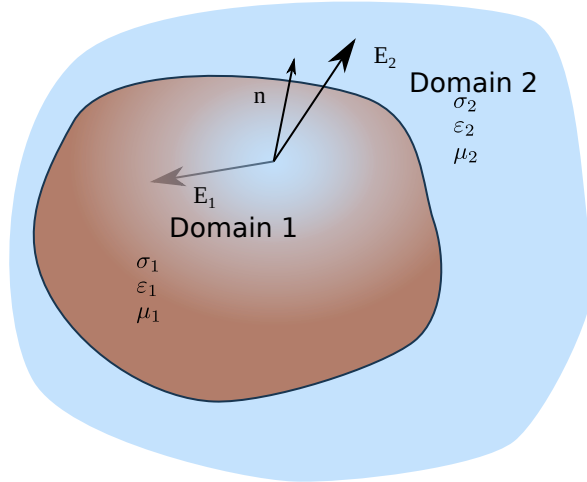


Figure 1.2: Crossing conditions

If we denote \mathbf{E}_1 , \mathbf{E}_2 , \mathbf{D}_1 , \mathbf{D}_2 , \mathbf{B}_1 , \mathbf{B}_2 , \mathbf{H}_1 , \mathbf{H}_2 , the values calculated with equations 1.2 to 1.5 on either side of the interface and \mathbf{n} , a normal to this interface, the relations between the values in the two domains are:

$$\mathbf{n} \times (\mathbf{E}_1 - \mathbf{E}_2) = \mathbf{0} \quad (1.47)$$

$$\mathbf{n} \cdot (\mathbf{B}_1 - \mathbf{B}_2) = 0 \quad (1.48)$$

$$\mathbf{n} \times (\mathbf{H}_2 - \mathbf{H}_1) = \mathbf{J}_{\text{surf}} \quad (1.49)$$

$$\mathbf{n} \cdot (\mathbf{D}_2 - \mathbf{D}_1) = \gamma \quad (1.50)$$

where γ is a surface charge density at the interface and \mathbf{J}_{surf} a surface current density.

In the context of code_Carmel, both these values are zero:

$$\begin{aligned} \gamma &= 0 \\ \mathbf{J}_{\text{surf}} &= \mathbf{0} \end{aligned} \quad (1.51)$$

1.7 Boundary conditions

Solving the system composed of Maxwell's equations and the constitutive relations (see paragraph 1.5) allows an infinite number of solutions. As a result, so that the problem is properly formulated mathematically and to ensure a unique solution, initial conditions are added for the time domain $[0, T]$ and boundary conditions are imposed for the space domain \mathcal{D} .

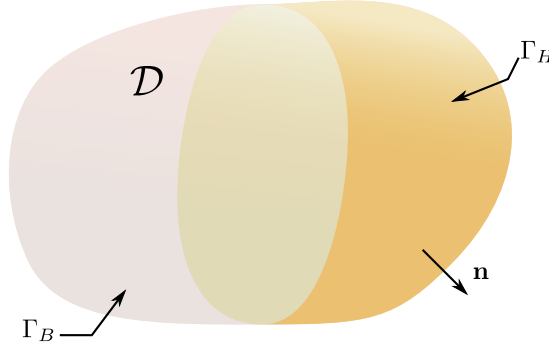


Figure 1.3: Boundary conditions

It is assumed that \mathcal{D} contains the source domain \mathcal{D}_s and the conductive domain \mathcal{D}_c which is a \mathbb{R}^3 bounded open set, simply connected, with connected Lipschitz boundary Γ_c . Finally, it is assumed that domains \mathcal{D}_c and \mathcal{D}_s are strictly disjoint ($\mathcal{D}_c \cap \mathcal{D}_s = \emptyset$) and strictly included in \mathcal{D} ($\mathcal{D}_s \cap \Gamma = \mathcal{D}_c \cap \Gamma = \emptyset$).

The boundary Γ of domain \mathcal{D} is broken down into two complementary parts denoted Γ_H and Γ_B such that $\Gamma_H \cap \Gamma_B = \emptyset$ and $\Gamma_H \cup \Gamma_B = \Gamma$ (see Figure 1.3). At the boundary Γ_H , boundary conditions are imposed of the form (perfect magnetic conductor):

$$\mathbf{H}(\mathbf{x}, t) \times \mathbf{n}|_{\Gamma_H} = \mathbf{0} \quad (1.52)$$

This condition makes it possible to consider a medium of infinite magnetic permeability on the other side of the domain, forcing the magnetic field to cross boundary Γ_H .

From equation 1.25 it is deduced that (electric wall):

$$\mathbf{J}(\mathbf{x}, t) \cdot \mathbf{n}|_{\Gamma_H} = 0 \quad (1.53)$$

At boundary Γ_B , in general, boundary conditions are imposed depending on the nature of the medium in contact with Γ_B . If the medium is conductive, we impose (perfect electrical conductor):

$$\mathbf{E}(\mathbf{x}, t) \times \mathbf{n}|_{\Gamma_B} = \mathbf{0} \quad (1.54)$$

from equation 1.3 it is deduced that (magnetic wall):

$$\mathbf{B}(\mathbf{x}, t) \cdot \mathbf{n}|_{\Gamma_B} = 0 \quad (1.55)$$

On the other hand, if the medium is non-conductive, boundary conditions are imposed only on \mathbf{B} and not on \mathbf{E} , as \mathbf{E} is not defined in the non-conductive areas [Golovanov et al 1998]. In this case, the only condition that can be imposed on \mathbf{E} is that its tangential component is written $\mathbf{E}_t = \mathbf{n} \times \mathbf{grad} \varphi$ with φ a scalar electric potential.

Finally, the conditions on Γ_c should be described. Indeed, this boundary is strictly included in the domain under study and thus should not bear any boundary conditions. However, with the magnetoquasistatic model, the electrical unknown is only taken into account inside the conductive domain. A boundary condition should be added for Γ_c . This is obtained for Γ_c from the tangential continuity equation of the magnetic field for zero surface currents:

$$[\mathbf{H} \times \mathbf{n}]_{\Gamma_c} = 0 \quad (1.56)$$

Then, considering the divergence of this equation and vector equality:

$$\text{div}(\mathbf{U} \times \mathbf{n}) = \mathbf{rot} \mathbf{U} \cdot \mathbf{n}$$

we have:

$$[\mathbf{rot} \mathbf{H} \cdot \mathbf{n}]_{\Gamma_c} = 0 \quad (1.57)$$

Finally, from the Maxwell-Ampère equation:

$$[(\mathbf{J}_s + \sigma \mathbf{E}) \cdot \mathbf{n}]_{\Gamma_c} = 0 \quad (1.58)$$

Now, \mathbf{J}_s is zero on Γ_c because there is no wound conductor in \mathcal{D}_c or in contact with \mathcal{D}_c , and also because: $\sigma = 0$ in $\mathcal{D} \setminus \mathcal{D}_c$. The boundary condition on \mathbf{E} along the length of Γ_c is thus obtained:

$$\sigma \mathbf{E} \cdot \mathbf{n}|_{\Gamma_c} = 0 \quad (1.59)$$

Remark 1.7.1 *The preceding equation is also written $\mathbf{J} \cdot \mathbf{n}|_{\Gamma_c} = 0$ which makes it possible to conserve \mathbf{J} between $\mathcal{D} \setminus \mathcal{D}_c$ where \mathbf{J} is zero, and \mathcal{D}_c where \mathbf{J} is defined and non-zero.*

Remark 1.7.2 *The case where the conductive domain \mathcal{D}_c and the source domain \mathcal{D}_s touch boundary Γ is thus not taken into account here. However, the model can easily be adapted to deal with this case.*

Remark 1.7.3 *In the case of the spectral version of code_Carmel,, the boundary conditions may not be zero. This point will be dealt with later.*

Chapter 2

Formulation of potential equations in code_Carmel

Summary

Maxwell's equations and associated constitutive relations can be solved by considering the fields as unknowns [Bossavit 2003], [Daveau, Rioux-Damidaou 1999], [Dular 1994], [Ren et al 1990]. Nevertheless, in code_Carmel, it is preferred to express the magnetic and electric fields as a function of potentials, which may be scalar or vector. The electrokinetic, magnetostatic and magnetodynamic equations are described here as a function of potentials, following the presentation of [Le Menach 1999] .

2.1 Electrokinetic problem

In the case of an electrokinetic problem (see Figure 2.1), the current density distribution is sought in a conductive material subjected, for example, to an electric potential difference [Le Menach 1999], [Korecki 2009].

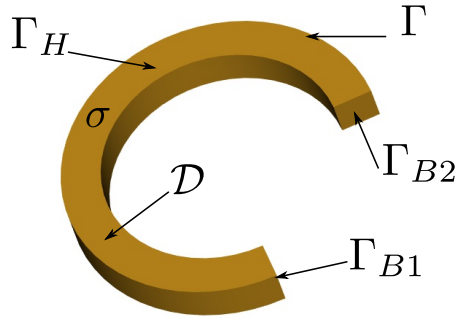


Figure 2.1: Typical electrokinetic problem

In the example in Figure 2.1, the current is sought in the portion of the ring of electrical conductivity σ from the fixed electric potentials on Γ_{B1} and Γ_{B2} .

2.1.1 Reminder of the equations

If the sources are of the continuous type (time-invariable), an electrokinetic problem can be solved to obtain the steady state of the electrical values in the conductive domain. In this case, the

system of equations to be solved is written:

$$\mathbf{rot} \mathbf{E}(\mathbf{x}) = \mathbf{0} \text{ avec } \mathbf{E} \times \mathbf{n}|_{\Gamma_B} = \mathbf{0} \quad (2.1)$$

$$\operatorname{div} \mathbf{J}_{\text{ind}}(\mathbf{x}) = 0 \text{ avec } \mathbf{J}_{\text{ind}} \cdot \mathbf{n}|_{\Gamma_H} = 0 \quad (2.2)$$

$$\mathbf{J}_{\text{ind}}(\mathbf{x}) = \sigma \mathbf{E}(\mathbf{x}) \quad (2.3)$$

where σ is constant for each sub-domain.

The distributions of \mathbf{E} and \mathbf{J}_{ind} are sought in the entire domain under study, and their changes are independent of time. Two potential formulations can be used to solve this type of problem.

Remark 2.1.1 *These formulations are not available in the spectral version of code_Carmel.*

2.1.2 Magnetic formulation φ

Given that the field has zero curl (see equation 2.1) and given that domain \mathcal{D} is simply connected and Γ connected, it is expressed as a function of a scalar electric potential φ_e such that:

$$\mathbf{E} = -\mathbf{grad} \varphi_e \quad (2.4)$$

By expressing the current density \mathbf{J}_{ind} as a function of the scalar potential φ_e and the electrical constitutive relation 1.37 (or, for the formulation φ , as the vector magnetic potential is no longer introduced into equation 2.32), the equation is solved:

$$\operatorname{div} \sigma \mathbf{grad} \varphi_e(\mathbf{x}) = 0 \text{ avec } \mathbf{E} = -\mathbf{grad} \varphi_e \quad (2.5)$$

In the case of Figure 2.1, the surface of the conductor is broken down into four parts Γ_B , Γ_H (with $\Gamma = \Gamma_B \cup \Gamma_H$), Γ_{B1} and Γ_{B2} (with $\Gamma_B = \Gamma_{B1} \cup \Gamma_{B2}$). The boundary conditions 1.54 are then written:

$$\varphi_e|_{\Gamma_{B1}} = \varphi_1 \quad \text{et} \quad \varphi_e|_{\Gamma_{B2}} = \varphi_2 \quad (2.6)$$

It will be noted that $\varphi_{12} = \varphi_1 - \varphi_2$ represents the potential difference imposed on the conductor. Since the scalar potential is defined as a constant, we can arbitrarily choose $\varphi_2 = 0$ and $\varphi_1 = \varphi_{12}$. A source scalar potential φ_s defined as follows (see Figure 2.2) is now introduced:

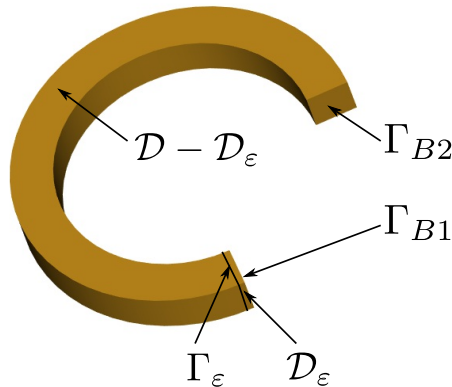


Figure 2.2: Typical electrokinetic problem and imposition of a scalar potential

$$\varphi_s|_{\Gamma_{B2}} = 0 ; \varphi_s|_{\Gamma_{B1}} = \varphi_{12} \text{ et } \varphi_s = 0 \text{ sur } \mathcal{D} - \mathcal{D}_\varepsilon \quad (2.7)$$

and φ_s varies linearly over the thickness ε , which is arbitrary. Potential φ_e can thus be written by introducing a potential φ ¹:

$$\varphi_e = \varphi + \varphi_s \text{ with } \varphi|_{\Gamma_B} = 0 \quad (2.8)$$

The initial problem is thus reduced to the following equivalent:

$$\operatorname{div} \sigma \operatorname{grad} \varphi(\mathbf{x}) = -\operatorname{div} \sigma \operatorname{grad} \varphi_s(\mathbf{x}) \quad (2.9)$$

with:

- $\mathbf{E} = -\operatorname{grad} (\varphi + \varphi_s)$;
- $\operatorname{div} \mathbf{J}_{ind} = 0$;
- $\mathbf{J}_{ind} = -\sigma \operatorname{grad} (\varphi + \varphi_s)$

2.1.3 Electrical formulation \mathbf{T}

Given that the current density \mathbf{J}_{ind} is, according to equation 1.32, a zero divergence vector field, it derives from a vector electric potential \mathbf{T}_e such that:

$$\mathbf{J}_{ind} = \operatorname{rot} \mathbf{T}_e \quad (2.10)$$

In addition, the current density \mathbf{J}_{ind} is normal to surfaces Γ_{B1} and Γ_{B2} . As a result, the flux of \mathbf{J}_{ind} through these surfaces, which represents the current intensity I_0 , can serve as a source term for the vector electric potential problem. Indeed, if we denote \mathbf{J}_0 a current density distributed uniformly in the conductor of cross-section S_c and \mathbf{n} the normal to S_c , this gives:

$$I_0 = \mathbf{J}_0 \cdot \mathbf{n} S_c \quad (2.11)$$

The current density \mathbf{J}_{ind} can then be written as the superposition of \mathbf{J}_0 plus a current density \mathbf{J}_m , thus giving [Biro et al 1993]:

$$\mathbf{J}_{ind} = \mathbf{J}_0 + \mathbf{J}_m \quad (2.12)$$

By definition, \mathbf{J}_0 is zero divergence so we can write:

$$\mathbf{J}_0 = \operatorname{rot} \mathbf{H}_s \quad (2.13)$$

where \mathbf{H}_s represents a source magnetic field². It will be noted that \mathbf{H}_s it is not unique and that there are an infinite number of source fields such that their curl gives current density \mathbf{J}_0 .

From equations 2.10, 2.12 and 2.13 it is deduced that \mathbf{J}_m is zero divergence. \mathbf{J}_m can thus also be expressed as a vector electric potential \mathbf{T} . Hence, this gives:

$$\mathbf{J}_{ind} = \operatorname{rot} (\mathbf{T} + \mathbf{H}_s) \quad (2.14)$$

Given that the normal component of \mathbf{J}_m is zero on Γ_H , \mathbf{T} will be taken such that:

$$\mathbf{T} \times \mathbf{n}|_{\Gamma_H} = \mathbf{0} \quad (2.15)$$

In fact, contrary to \mathbf{T}_e defined in 2.10, the circulation of \mathbf{T} on a curve C of Γ , surrounding the conductor, is equal to zero since the flux of \mathbf{J}_m is zero across any section³.

However, the circulation of \mathbf{H}_s on this contour C is equal to I_0 .

¹As indicated in paragraph 5.1.4, $\varphi \in H_{0,B}(\operatorname{grad}, \mathcal{D})$

²belonging to $\mathbf{H}(\operatorname{rot}, \mathcal{D})$, as shown below

³According to equations 2.14 and 2.15, the vector potential \mathbf{T} belongs to $\mathbf{H}_{0,H}(\operatorname{rot}, \mathcal{D})$.

In the same way as for the scalar electric potential formulation, the initial problem is equivalent to solving:

$$\mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{T}(\mathbf{x}) = -\mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \quad (2.16)$$

with:

$$\mathbf{J}_{\text{ind}} = \mathbf{rot} (\mathbf{T} + \mathbf{H}_s)$$

However, it will be noted that the vector potential \mathbf{T} is defined at a specific gradient. Equation 2.16 thus allows for an infinite number of solutions. To ensure a unique solution, we must impose a gauge condition. There are several, notably the Coulomb gauge [Durand 1968], [Fournet 1985]:

$$\text{div} \mathbf{T} = 0$$

Another gauge consists in imposing the scalar product $\mathbf{U} \cdot \mathbf{w} = \mathbf{f}(r)$ [Albanese, Rubinacci 2000]. (see Annex C).

2.2 Magnetostatic problem

2.2.1 Reminder of the equations

When the problem does not involve induced currents, we are required to solve the magnetostatic equations, which are written:

$$\mathbf{rot} \mathbf{H}(\mathbf{x}) = \mathbf{J}_s(\mathbf{x}) \text{ avec } \mathbf{H} \times \mathbf{n}|_{\Gamma_H} = \mathbf{0} \quad (2.17)$$

$$\text{div} \mathbf{B}(\mathbf{x}) = 0 \text{ avec } \mathbf{B} \cdot \mathbf{n}|_{\Gamma_B} = 0 \quad (2.18)$$

$$\mathbf{B}(\mathbf{x}) = \mu \mathbf{H}(\mathbf{x}) + \mathbf{B}_r \quad (2.19)$$

where μ is constant for each sub-domain.

Two potential formulations may be used.

Remark 2.2.1 *Coupling between the potential equations defined in the conductive and non-conductive domain occurs naturally if formulations of the same type are used, such as the formulation $\mathbf{A} - \varphi$ ($\mathbf{T} - \Omega$ respectively) for the conductive domain and the formulation \mathbf{A} (Ω respectively) for the non-conductive domain. It suffices to satisfy certain continuity conditions for the potentials [Boualem 1997] [Dular 1994].*

2.2.2 Vector magnetic potential formulation \mathbf{A}

Given that the magnetic flux density is zero divergence, according to equation 1.4, a vector magnetic potential, denoted \mathbf{A} , can be introduced such that:

$$\mathbf{B}(\mathbf{x}) = \mathbf{rot} \mathbf{A}(\mathbf{x}) \quad (2.20)$$

The normal component of \mathbf{B} being zero on Γ_B , the boundary conditions for the vector potential are as follows:

$$\mathbf{A} \times \mathbf{n}|_{\Gamma_B} = \mathbf{K}_A \quad (2.21)$$

with \mathbf{A} defined in the entire domain \mathcal{D} ⁴

⁴According to equations 2.20 and 2.21, the vector potential \mathbf{A} belongs to $\mathbf{H}_{0,b}(\mathbf{rot}, \mathcal{D})$.

As with the scalar potential formulation, the source term is generally the current density. Without changing the general appearance of the problem, we will take \mathbf{K}_A as being equal to zero on Γ_B . However, if it is desired to impose a flux, a source vector potential \mathbf{A} can be superimposed on the vector potential \mathbf{A}_s over all or part of domain \mathcal{D} (see paragraph 2.1.3).

This expression of the magnetic flux density (equation 2.20) leads to a new equation for the magnetic field:

$$\mathbf{H} = \frac{1}{\mu} \text{rot } \mathbf{A} - \frac{1}{\mu} \mathbf{B}_r$$

Using 2.17 the equation to be solved is given by:

$$\text{rot} \frac{1}{\mu} \text{rot} \mathbf{A}(\mathbf{x}) = \mathbf{J}_s(\mathbf{x}) + \frac{1}{\mu} \text{rot } \mathbf{B}_r \quad (2.22)$$

where μ is constant for each sub-domain.

It has been shown (from the equations in paragraph 5.1.1) that there are an infinite number of solutions for the vector potential \mathbf{A} . A unique solution can be obtained, as for the vector electric potential (see paragraph 2.1.3), by imposing a gauge condition [Albanese, Rubinacci 2000].

Remark 2.2.2 *In the spectral version of code_Carmel, no gauge is applied.*

2.2.3 Scalar magnetic potential formulation Ω

To take account of the inductors, where the current density \mathbf{J}_s is known, we introduce, as in the electrokinetic case, a source magnetic field \mathbf{H}_s defined by equation 2.13.

$$\mathbf{J}_s = \text{rot } \mathbf{H}_s$$

Given that the domain is simply connected, it is possible to introduce a scalar magnetic potential Ω such that:

$$\mathbf{H}(\mathbf{x}) = \mathbf{H}_s(\mathbf{x}) - \text{grad } \Omega(\mathbf{x}) \quad (2.23)$$

In contrast to electrokinetic problems, we can have \mathbf{H}_s such that $\mathbf{H}_s \times \mathbf{n} = \mathbf{0}$ on Γ_H . Indeed, the boundary Γ_H is not entirely in contact with the inductors. With the boundary conditions on \mathbf{H} , we have:

$$-\text{grad} \Omega \times \mathbf{n}|_{\Gamma_H} = \mathbf{0} \text{ d'où } \Omega|_{\Gamma_H} = K_\Omega \quad (2.24)$$

where K_Ω is a constant. It is possible to introduce a magnetic potential difference between two disjoint surfaces of Γ_H by adding a source scalar potential as proposed in the electrokinetic case. In magnetostatic applications processed using code_Carmel, only inductors will be considered as magnetic field sources. Consequently, in the following, we will take K_Ω as being equal to zero⁵

The equation to be solved is deduced from 1.4 and 2.23 such that:

$$\text{div } \mu \text{grad } \Omega(\mathbf{x}) = \text{div } \mu \mathbf{H}_s(\mathbf{x}) + \text{div } \mathbf{B}_r \quad (2.25)$$

where μ is constant for each sub-domain.

⁵Under these conditions, the scalar potential Ω thus belongs to sub-space $\mathbf{H}_{0,h}(\text{grad}, \mathcal{D})$.

2.3 Magnetodynamic problem

2.3.1 Reminder of the equations

We consider a domain \mathcal{D} containing a conductive domain \mathcal{D}_c , assumed to be contractible, and wound inductors.

To simplify the mathematics, we will limit the electromagnetic field sources to a single inductor but extension to several inductors is quite possible, as shown in studies carried out with code `_Carmel`. Finally, it is also possible to apply an electric potential difference to the terminals of the conductive domain or to impose a current there.

For a wound inductor, a source magnetic field is defined, denoted \mathbf{H}_s , such that $\text{rot}\mathbf{H}_s = \mathbf{J}_s$ with $\mathbf{H}_s \times \mathbf{n} = \mathbf{0}$ on Γ_H .

Remark 2.3.1 *It will be noted that \mathbf{H}_s is not unique and that there are an infinite number of source fields such that their curl is equal to the current density flowing through the wound inductor.*

Unless there are specific constraints on the field, it can be defined for the entire domain \mathcal{D} . Under these conditions, the local form of Ampère's circuital law is written:

$$\text{rot}\mathbf{H}(\mathbf{x}, t) = \mathbf{J}_{\text{ind}}(\mathbf{x}, t) + \mathbf{J}_s(\mathbf{x}, t) \quad (2.26)$$

with:

$$\mathbf{J}_{\text{ind}}(\mathbf{x}, t) = \sigma \mathbf{E}(\mathbf{x}, t) \quad (2.27)$$

where:

- \mathbf{J}_{ind} represents the induced current density in the conductive domain \mathcal{D}_c ;
- σ is the electrical conductivity, constant for each sub-domain of the conductive domain \mathcal{D}_c .

In addition:

$$\text{rot}\mathbf{H}_s(\mathbf{x}, t) = \mathbf{J}_s(\mathbf{x}, t) \quad (2.28)$$

Two potential formulations can be introduced: the electrical formulation and the magnetic formulation. These formulations are defined only in the conductive domain \mathcal{D}_c (the term \mathbf{J}_s is thus zero, though \mathbf{H}_s is not necessarily zero).

2.3.2 Electrical formulation $\mathbf{A} - \varphi$

Given that the magnetic flux density is zero divergence, according to equation 1.4, a vector magnetic potential, denoted \mathbf{A} , can be introduced such that:

$$\mathbf{B}(\mathbf{x}, t) = \text{rot}\mathbf{A}(\mathbf{x}, t) \text{ avec } \mathbf{A} \times \mathbf{n}|_{\Gamma_B} = \mathbf{0} \quad (2.29)$$

with \mathbf{A} defined in the entire domain \mathcal{D} .

By using equation 1.3 and according to equation 2.29, the field \mathbf{E} can be expressed as a function of the vector potential defined at a specific gradient. We thus have:

$$\mathbf{E}(\mathbf{x}, t) = -\frac{\partial \mathbf{A}(\mathbf{x}, t)}{\partial t} - \text{grad}(\varphi(\mathbf{x}, t) + \varphi_s) \quad (2.30)$$

where φ is the scalar electric potential defined in paragraph 2.1.2. If we consider only short-circuit conductors, the potential φ_s is zero⁶.

By replacing the magnetic field \mathbf{H} and the current density \mathbf{J}_{ind} by their expressions as a function of \mathbf{A} and φ , and taking into account constitutive relation 1.46, the local form of Ampère's circuital law 1.25 and the current density conservation law 1.29 are written:

$$\mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{A}(\mathbf{x}, t) + \sigma \left(\frac{\partial \mathbf{A}(\mathbf{x}, t)}{\partial t} + \mathbf{grad} \varphi(\mathbf{x}, t) \right) = \mathbf{J}_s(\mathbf{x}, t) + \frac{1}{\mu} \mathbf{rot} \mathbf{B}_r \quad (2.31)$$

$$\text{div} \sigma \left(\frac{\partial \mathbf{A}(\mathbf{x}, t)}{\partial t} + \mathbf{grad} \varphi(\mathbf{x}, t) \right) = 0 \quad (2.32)$$

where μ and σ are constant for each sub-domain.

An infinite number of vectors \mathbf{A} can be defined such that their curl is equal to the magnetic flux density. To ensure the uniqueness of this potential, a gauge condition is introduced such as the Coulomb gauge $\text{div} \mathbf{A} = 0$ or a shape condition $\mathbf{A} \cdot \mathbf{W} = 0$ with \mathbf{W} a vector field whose field lines do not form loops and are such that they connect all the points of the domain [Albanese, Rubinacci 2000], [Kettunen et al 1999]⁷.

Remark 2.3.2 *In the spectral version of code_Carmel, no gauge is applied.*

2.3.3 Magnetic formulation $\mathbf{T} - \Omega$

The formulation $\mathbf{T} - \Omega$ in the spectral version of code_Carmel is limited to linear problems with source fields of the wound type.

In the case of a magnetic formulation, field \mathbf{H} is expressed as a function of potentials and field \mathbf{H}_s . Given that the induced current density is zero divergence, a vector electric potential, denoted \mathbf{T} , can be introduced such that:

$$\mathbf{J}_{\text{ind}}(\mathbf{x}, t) = \mathbf{rot} \mathbf{T}(\mathbf{x}, t) \quad (2.33)$$

with \mathbf{T} defined in the conductive domain.

Given that the conductive domain is assumed to be contractible, we can then take $\mathbf{T} = \mathbf{0}$ outside the conductive domain and impose $\mathbf{T} \times \mathbf{n} = \mathbf{0}$ on boundary Γ_c of \mathcal{D}_c .

Given that $\mathbf{rot} \mathbf{H}(\mathbf{x}, t) = \mathbf{J}_{\text{ind}}(\mathbf{x}, t) + \mathbf{J}_s(\mathbf{x}, t)$, this gives:

$$\mathbf{rot} (\mathbf{H}(\mathbf{x}, t) - \mathbf{H}_s(\mathbf{x}, t) - \mathbf{T}(\mathbf{x}, t)) = \mathbf{0} \quad (2.34)$$

Field \mathbf{H} can thus be expressed as a function of vector potential \mathbf{T} and field \mathbf{H}_s defined at a specific gradient. We thus have:

$$\mathbf{H}(\mathbf{x}, t) = \mathbf{H}_s(\mathbf{x}, t) + \mathbf{T}(\mathbf{x}, t) - \mathbf{grad} \Omega(\mathbf{x}, t) \text{ avec } \mathbf{T} \times \mathbf{n}|_{\Gamma_c} = \mathbf{0} \text{ et } \Omega|_{\Gamma_H} = 0 \quad (2.35)$$

with Ω the scalar magnetic potential defined in the entire domain.

By introducing equations 2.33 and 2.35 into the local form of Faraday's law 1.3 and the magnetic induction conservation law 1.4, the system to be solved is written in the form:

⁶We will see that φ belongs to $\mathbf{H}_{0,B}(\mathbf{grad}, \mathcal{D})$ and \mathbf{A} belongs to $\mathbf{H}_{0,B}(\mathbf{rot}, \mathcal{D})$.

⁷A specific gauge can sometimes be chosen that sets $\varphi = 0$. This is the formulation \mathbf{A}^* . This formulation is not currently available in code_Carmel

$$\mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{T}(\mathbf{x}, t) + \frac{\partial}{\partial t} \mu (\mathbf{T}(\mathbf{x}, t) - \mathbf{grad} \Omega(\mathbf{x}, t)) = -\mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s(\mathbf{x}, t) - \frac{\partial}{\partial t} (\mu \mathbf{H}_s(\mathbf{x}, t) + \mathbf{B}_r) \quad (2.36)$$

$$\operatorname{div} \mu (\mathbf{T}(\mathbf{x}, t) - \mathbf{grad} \Omega(\mathbf{x}, t)) = -\operatorname{div} (\mu \mathbf{H}_s(\mathbf{x}, t) + \mathbf{B}_r) \quad (2.37)$$

where μ and σ are constant for each sub-domain.

As with the formulation **A** - φ , a gauge condition must be applied to vector potential \mathbf{T} to ensure uniqueness. This gauge is defined only in the conductive domain \mathcal{D}_c .

Remark 2.3.3 *In the spectral version of code_Carmel, no gauge is applied.*

Chapter 3

Computation and imposition of global electromagnetic quantities

Summary

The preceding chapters have provided the local equations to be solved for magnetodynamic, electrokinetic and magnetostatic problems. The sources are naturally current densities. However, in practice it is useful to impose data consisting of other electrical values. Hence, in addition to these sets of equations, overall equations can be added (often in integral forms). This chapter introduces these complementary expressions and a way to introduce them [Korecki 2009].

3.1 Introduction of K and N fields

Consider the system, included in domain \mathcal{D} of boundary Γ , shown in Figure 3.1). It is made up of an inductor (\mathcal{D}_s^i boundary Γ_s^i) and the domain \mathcal{D}_{nc} .

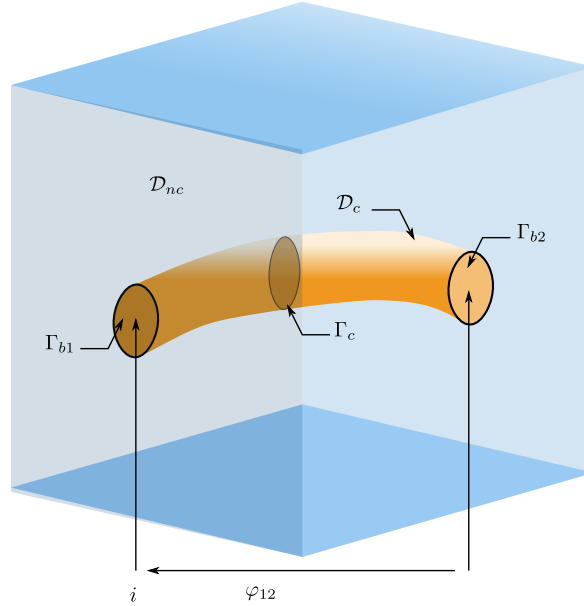


Figure 3.1: Definition of overall values and boundary conditions for the problem studied

On the surfaces Γ_{b1} and Γ_{b2} we impose either a current density flux I_s or a potential difference $V = \varphi_1 - \varphi_2$.

It is assumed that the current density \mathbf{J}_s in the inductor is uniformly distributed. In addition, the cross-section S_s^i of the inductor is assumed to be constant. If \mathbf{n} denotes the normal to S_s^i , we can define a vector field \mathbf{N} such that:

$$\mathbf{N} = \frac{1}{S_s^i} \mathbf{n} \text{ dans } \mathcal{D}_s^i \quad (3.1)$$

$$\mathbf{N} = \mathbf{0} \text{ dans } \mathcal{D}_{nc} \quad (3.2)$$

The normal component of \mathbf{N} is zero on $\Gamma_s^i - (\Gamma_{b1} \cup \Gamma_{b2})$. In addition, by definition, \mathbf{N} is a zero divergence vector field and allows definition of the geometry of the inductor. Based on the properties of \mathbf{N} , a vector \mathbf{K} can be introduced such that:

$$\text{rot } \mathbf{K} = \mathbf{N} \quad (3.3)$$

If the surface of the conductor is not fully in contact with surface Γ_h , then:

$$\mathbf{K} \times \mathbf{n}|_{\Gamma_h} = \mathbf{0} \quad (3.4)$$

Under these conditions, the vectors \mathbf{N} and \mathbf{K} belong respectively to $\mathbf{H}_{0,x}(\text{div}, \mathcal{D})$ and to $\mathbf{H}(\text{rot}, \mathcal{D})$. If $\Gamma_s^i \setminus (\Gamma_{b1} \cup \Gamma_{b2})$ is included in Γ_h (the electrokinetic problem), then \mathbf{N} still belongs to $\mathbf{H}(\text{div}, \mathcal{D})$. Conversely, \mathbf{K} then belongs to $\mathbf{H}(\text{rot}, \mathcal{D})$ as the circulation of \mathbf{K} on Γ_h is not zero. It is important to note that there are an infinite number of vectors \mathbf{K} for which the curl is equal to \mathbf{N} .

For solid conductors, there is no direct link between the current density distribution \mathbf{J} and the vector \mathbf{N} . Conversely, for multifilamentary conductors it can be assumed that the current density is uniformly distributed, and thus:

$$I_s = \int_{S_s^i} \mathbf{J}_s \cdot d\mathbf{s} = \mathbf{J}_s \cdot \mathbf{S}_s^i \quad (3.5)$$

where I_s represents the current flowing through the inductor. The current density \mathbf{J}_s can then be expressed as a function of vector \mathbf{N} by the equation:

$$\mathbf{J}_s = \mathbf{N} I_s \quad (3.6)$$

In addition, based on equations 2.28 and 3.3, we have:

$$\mathbf{H}_s = I_s \mathbf{K} \quad (3.7)$$

Vectors \mathbf{N} and \mathbf{K} , which are the carriers of vectors \mathbf{J}_s and \mathbf{H}_s , allow coupling of the electromagnetism equations and those of the electrical circuit.

3.2 Introduction of the function α et du champ β

A vector field β is defined with the following properties:

$$\begin{aligned} \text{rot } \beta &= \mathbf{0} \\ \beta \times \mathbf{n}|_{\Gamma_m} &= \mathbf{0} \\ \int_{\gamma_{12}} \beta \cdot d\mathbf{l} &= 1 \end{aligned} \quad (3.8)$$

As field β has zero curl, a scalar function α is defined such that:

$$\begin{aligned}
\beta &= -\mathbf{grad} \alpha \\
\alpha|_{\Gamma_m} &= Cte \\
\alpha_{2-1} &= 1
\end{aligned} \tag{3.9}$$

3.3 Electrokinetics

The previous equations are applied here to the case of an electrokinetic model. According to the formulation, to impose overall electrical values, the following conditions must be verified:

- for the voltage:

$$V_{\Gamma_{b2}} - V_{\Gamma_{b1}} = \int_{\gamma} \mathbf{E} \cdot d\mathbf{l} = V \tag{3.10}$$

- for the current:

$$\int_{\Gamma_{b1}} \mathbf{J} \cdot \mathbf{n} ds = - \int_{\Gamma_{b2}} \mathbf{J} \cdot \mathbf{n} ds = I \tag{3.11}$$

These two equations can be used to determine the voltage if the current is imposed and vice versa, after solving the problem.

3.3.1 Vector electric potential formulation \mathbf{T}

The current density can be broken down into two terms: an unknown term \mathbf{J}_{ind} and a source term \mathbf{J}_s .

$$\mathbf{J} = \mathbf{J}_{ind} + \mathbf{J}_s \tag{3.12}$$

In the case of this formulation, it is possible to show the current I in the source term expression.

3.3.1.1 Imposition of the current

The vector field \mathbf{N} has the same properties as the source current density at a given I . It can thus be used to characterise this source current density:

$$\mathbf{J}_s = I \mathbf{N} \tag{3.13}$$

Hence, we validate equation 3.11 which is now written in the form:

$$\begin{aligned}
\int_{\Gamma_{b1}} \mathbf{J} \cdot \mathbf{n} ds &= \int_{\Gamma_{b1}} \mathbf{J}_{ind} \cdot \mathbf{n} ds + \int_{\Gamma_{b1}} \mathbf{J}_s \cdot \mathbf{n} ds = I \\
\int_{\Gamma_{b2}} \mathbf{J} \cdot \mathbf{n} ds &= \int_{\Gamma_{b2}} \mathbf{J}_{ind} \cdot \mathbf{n} ds + \int_{\Gamma_{b2}} \mathbf{J}_s \cdot \mathbf{n} ds = -I
\end{aligned} \tag{3.14}$$

With:

$$\int_{\Gamma_{b1}} \mathbf{J}_{ind} \cdot \mathbf{n} ds = 0 \quad \text{et} \quad \int_{\Gamma_{b1}} \mathbf{J}_s \cdot \mathbf{n} ds = \int_{\Gamma_{b1}} I \mathbf{N} \cdot \mathbf{n} ds = I \tag{3.15}$$

By considering C , a non-contractible contour defined on Γ_h , the following equation is also validated:

$$\oint_C \mathbf{T} \cdot d\mathbf{l} = I \tag{3.16}$$

Where \mathbf{T} is the vector electric potential, which itself is broken down into two terms:

$$\mathbf{T} = \mathbf{T}_I + \mathbf{T}_s \quad (3.17)$$

By introducing vector potential \mathbf{K} in the expression for the source term \mathbf{T}_s , the current density can be written:

$$\mathbf{J} = \text{rot} (\mathbf{T}_I + I \mathbf{K}) \quad (3.18)$$

Here we find the concept of cut-out (see chapter 4):

$$\mathbf{K} \times \mathbf{n} \neq 0 \quad \text{sur} \quad \Gamma_h$$

It is carried out here using vector \mathbf{K} .

Using this vector field, the current appears in the vector electric potential formulation, which is now written as follows:

$$\text{rot} \frac{1}{\sigma} \text{rot} \mathbf{T}_I = -\text{rot} \frac{1}{\sigma} \text{rot} (I \mathbf{K}) \quad (3.19)$$

It has been shown [Henneron 2004] that the expression for the voltage is given by:

$$V = \int_{\mathcal{D}_c} \mathbf{E} \cdot \mathbf{N} d\mathcal{D}_c \quad (3.20)$$

3.3.1.2 Imposition of the voltage

To impose the voltage in this formulation, we use equation 3.20 which is added to the original system of equations 3.19. The current then becomes an unknown and the system to be solved is written:

$$\begin{aligned} \text{rot} \frac{1}{\sigma} \text{rot} \mathbf{T}_I + \text{rot} \frac{1}{\sigma} \text{rot} (I \mathbf{K}) &= 0 \\ \int_{\mathcal{D}_c} \mathbf{E} \cdot \mathbf{N} d\mathcal{D}_c &= V \end{aligned} \quad (3.21)$$

3.3.2 Scalar electric potential formulation φ

The electric field \mathbf{E} is here broken down into two terms: a source field \mathbf{E}_s and an unknown field \mathbf{E}_I such that:

$$\begin{aligned} \mathbf{E} &= \mathbf{E}_s + \mathbf{E}_I \\ &= -\text{grad} \varphi_s - \text{grad} \varphi_I \end{aligned} \quad (3.22)$$

This source field allows the introduction of voltage V into the expression for the total electric field.

3.3.2.1 Imposition of the voltage

This source electric field has the same properties as field β at a given V . It is thus written as follows:

$$\mathbf{E}_s = \beta V \quad (3.23)$$

It is then verified that:

$$\int_{\gamma} \mathbf{E}_S \cdot d\mathbf{l} = \int_{\gamma} (\beta V) \cdot d\mathbf{l} = V \quad (3.24)$$

For the electric field \mathbf{E}_I it is verified that:

$$\int_{\gamma} \mathbf{E}_I \cdot d\mathbf{l} = 0 \quad (3.25)$$

By using potential α in the expression for the source field, we establish the electric potential formulation φ at imposed voltage:

$$\text{div } \sigma \mathbf{grad} \varphi_I = -\text{div } \sigma \mathbf{grad} \alpha V \quad (3.26)$$

A power balance provides the expression for current I in which the vector field β appears:

$$I = \int_{\mathcal{D}_c} \beta \cdot \mathbf{J} d\mathcal{D}_c \quad (3.27)$$

3.3.2.2 Imposition of the current

To impose the current with the scalar potential formulation, we use equation 3.27. We then express β and \mathbf{J} as a function of α and the scalar electric potential φ_I . The scalar electric potential formulation with an imposed current is written:

$$\begin{aligned} \text{div } \sigma \mathbf{grad} \varphi_I + \text{div } \sigma \mathbf{grad} \alpha V &= 0 \\ \int_{\mathcal{D}_c} \mathbf{grad} \alpha \cdot \sigma \mathbf{grad} (\varphi_I + \alpha V) d\mathcal{D}_c &= I \end{aligned} \quad (3.28)$$

The voltage thus becomes an unknown when the current is imposed.

3.3.3 Review of imposing overall values in electrokinetics

Imposing a current with the vector potential formulation and a potential difference with the scalar potential formulation is natural. In this case, the overall values are revealed by acting on the source terms. Conversely, if we wish to impose a voltage with the vector potential formulation and a current with the scalar potential formulation, it is necessary to add an equation derived from a power balance, established using the vector fields \mathbf{N} or β .

The table (see Table 3.1) summarises the two potential formulations used in electrokinetics with the imposition of overall electrical values.

These vector fields can be used on systems that require the consideration of several electrical sources (currents and/or voltages). Several electric potential differences and several current sources can be defined according to the values to be determined.

3.4 Magnetostatics

3.4.1 Formulation A

3.4.1.1 Imposition of a flux

The magnetic induction is broken down into two terms: a source term \mathbf{B}_s and an unknown term \mathbf{B}_i such that:

Formulations	Imposition de la tension	Imposition du courant
φ	$\text{div } \sigma \mathbf{grad} (\varphi_I + \alpha V) = 0$	$\text{div } \sigma \mathbf{grad} (\mathbf{J}_I + \alpha V) = 0$ $\int_{\mathcal{D}_c} \beta \cdot \mathbf{J} d\mathcal{D}_c = I$
\mathbf{T}	$\mathbf{rot} \frac{1}{\sigma} \mathbf{rot} (\mathbf{T}_I + I \mathbf{K}) = 0$ $\int_{\mathcal{D}_c} \mathbf{E} \cdot \mathbf{N} d\mathcal{D}_c = V$	$\mathbf{rot} \frac{1}{\sigma} \mathbf{rot} (\mathbf{T}_I + I \mathbf{K}) = 0$

Table 3.1: Imposition of overall values in electrokinetics

$$\mathbf{B} = \mathbf{B}_s + \mathbf{B}_i$$

The source term is expressed as a function of an \mathbf{N} or \mathbf{K} vector field and the imposed flux ϕ :

$$\mathbf{B}_s = \mathbf{N} \phi = \mathbf{rot} \mathbf{K} \phi \quad (3.29)$$

The unknown field \mathbf{B}_i allows introduction of the vector magnetic potential \mathbf{A} such that:

$$\mathbf{B}_i = \mathbf{rot} \mathbf{A}$$

The flux thus appears in the formulation through us of the source flux density \mathbf{B}_s :

$$\mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{A} = \mathbf{J}_s - \mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{K} \phi \quad (3.30)$$

3.4.1.2 Imposition of a magnetic potential difference

By only taking into account overall magnetic values, ϵ and ϕ , the magnetic energy can be written as follows:

$$W = \frac{1}{2} \int_{\mathcal{D}} \mathbf{H} \cdot \mathbf{B} d\tau = \frac{1}{2} \epsilon \cdot \phi \quad (3.31)$$

After development, the magnetic potential difference ϵ resulting from imposition of the flow is written:

$$\epsilon = \int_{\mathcal{D}} \mathbf{H} \cdot \mathbf{N} d\tau \quad (3.32)$$

By combining this equation with the formulation previously found, the magnetic potential difference can be imposed:

$$\begin{aligned} \mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{A} + \mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{K} \phi &= \mathbf{J}_s \\ \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} (\mathbf{A} + \mathbf{K} \phi) \cdot \mathbf{N} d\tau &= \epsilon \end{aligned} \quad (3.33)$$

3.4.2 Formulation in Ω

3.4.2.1 Imposition of a flux

Still using equation 3.31, we can establish an expression for the flux as a function of the flux density \mathbf{B} and function β

$$\phi = \int_{\mathcal{D}} \beta \cdot \mathbf{B} \, d\tau \quad (3.34)$$

By analogy with the scalar electric potential formulation φ , equation 3.34 is used to impose the magnetic flux using the scalar magnetic potential formulation Ω :

$\begin{aligned} \operatorname{div} \mu \operatorname{grad} \Omega + \operatorname{div} \mu \operatorname{grad} \alpha \epsilon &= \operatorname{div} \mu \mathbf{H}_s \\ \int_{\mathcal{D}} \operatorname{grad} \alpha \cdot \mu (\operatorname{grad} (\Omega + \alpha \epsilon) - \mathbf{H}_s) \, d\tau &= \phi \end{aligned} \quad (3.35)$

3.4.2.2 Imposition of a magnetic potential difference

In the case of scalar potential formulation, the imposition of a magnetic potential difference ϵ requires the addition of a term \mathbf{H}_G which is expressed as a function of vector field β :

$$\int_{\gamma} \mathbf{H}_G \cdot d\mathbf{l} = \int_{\gamma} (\beta \epsilon) \cdot d\mathbf{l} = \epsilon \quad (3.36)$$

where γ is some path connecting Γ_{H1} and Γ_{H2} .

The magnetic field \mathbf{H} is thus broken down into three terms:

- The unknown term \mathbf{H}_I that introduces the scalar magnetic potential Ω ;
- The source term \mathbf{H}_s relating to the presence of inductors;
- The term \mathbf{H}_G , which corresponds to the introduction of the overall value ϵ , the magnetic potential difference.

$$\mathbf{H} = \mathbf{H}_I + \mathbf{H}_s + \mathbf{H}_G = -\operatorname{grad} \Omega + \mathbf{H}_s - \operatorname{grad} \alpha \epsilon \quad (3.37)$$

The scalar magnetic potential formulation is then written:

$\operatorname{div} \mu \operatorname{grad} \Omega = -\operatorname{div} \mu \operatorname{grad} \alpha \epsilon + \operatorname{div} \mu \mathbf{H}_s \quad (3.38)$

3.4.3 Review of imposing overall values in magnetostatics

As with the electrokinetic formulations, some values appear naturally in the formulations, such as the flux for vector magnetic potential formulation \mathbf{A} and the magnetic potential difference for scalar magnetic potential formulation Ω . With the tools introduced, it is possible to use one or the other to solve a problem involving imposed flux or imposed magnetic potential difference.

The table below groups the different magnetostatic formulations according to the values to be imposed.

Formulation	Imposition of magnetic p.d.	Imposition of flux
Ω	$\operatorname{div} \mu (\mathbf{grad} (\Omega + \alpha \epsilon) - \mathbf{H}_s) = 0$	$\operatorname{div} \mu (\mathbf{grad} (\Omega + \alpha \epsilon) - \mathbf{H}_s) = 0$ $\int_{\mathcal{D}} \boldsymbol{\beta} \cdot \mathbf{B} d\tau = \phi$
\mathbf{A}	$\operatorname{rot} \frac{1}{\mu} \operatorname{rot} \mathbf{A} + \operatorname{rot} \frac{1}{\mu} \operatorname{rot} \mathbf{K} \phi = \mathbf{J}_s$ $\int_{\mathcal{D}} \frac{1}{\mu} \operatorname{rot} (\mathbf{A} + \mathbf{K} \phi) \cdot \mathbf{N} d\tau = \epsilon$	$\operatorname{rot} \frac{1}{\mu} \operatorname{rot} \mathbf{A} + \operatorname{rot} \frac{1}{\mu} \operatorname{rot} \mathbf{K} \phi = \mathbf{J}_s$

Table 3.2: Imposition of overall values in magnetostatics

3.5 Magnetodynamics

3.5.1 Formulation \mathbf{A} - φ

3.5.1.1 Imposition of a voltage in a wound conductor

The current intensity denoted i is unknown. Coupling will take place using the magnetic induction flux denoted ϕ . By using vector \mathbf{K} , the expression for flux ϕ is [Le Menach 1999]:

$$\phi = \int_{\mathcal{D}} \mathbf{B} \cdot \mathbf{K} d\mathcal{D} \quad (3.39)$$

The value of this expression over the conventional form for flux $\phi = \int \mathbf{B} \cdot d\mathbf{s}$ is in its volume integral. If the inductor geometry is complex, it is difficult to determine the 3D surface bounded by the inductor. In this case, it is difficult to calculate flux \mathbf{B} across such a surface. Since vector field \mathbf{K} must be calculated to determine the source field, this creates no additional difficulties. Its general form facilitates coupling with the formulations. To do this, equation 3.39 is introduced into Faraday's law thus giving:

$$\frac{d}{dt} \int_{\mathcal{D}} \mathbf{B} \cdot \mathbf{K} d\mathcal{D} = V - Ri \quad (3.40)$$

where:

- V represents the potential difference between the inductor terminals;
- R is the resistance of the inductor;
- i is the current flowing through the inductor.

In equation 3.39 the magnetic induction can be replaced by the curl of the vector magnetic potential \mathbf{A} . We then obtain an expression of the form $\operatorname{rot} \mathbf{A} \cdot \mathbf{K}$ that can be transformed using the properties of the vector operators. Given the boundary conditions on \mathbf{A} and \mathbf{K} , this gives:

$$\phi = \int_{\mathcal{D}} \mathbf{A} \cdot \operatorname{rot} \mathbf{K} d\mathcal{D} \quad (3.41)$$

By using equation 3.3, the magnetic induction flux in a winding made up of wound inductors is then written:

$$\phi = \int_{\mathcal{D}} \mathbf{A} \cdot \mathbf{N} d\mathcal{D} \quad (3.42)$$

From this, the equations for imposed voltage on a wire conductor are deduced:

$$\begin{aligned} \mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{A} - \mathbf{N} i &= 0 \\ \frac{d}{dt} \int_{\mathcal{D}} \mathbf{A} \cdot \mathbf{N} d\mathcal{D} + R i &= V \end{aligned} \quad (3.43)$$

3.5.1.2 Imposition of a flux and of a voltage in a solid conductor

For formulation $\mathbf{A} - \varphi$, the overall values that appear naturally are: the magnetic flux (via the introduction of the source term $\mathbf{N} \phi$) and the electric potential difference (using the source electric field βV). They appear by breaking down the magnetic induction \mathbf{B} (3.44) and the electric field \mathbf{E} into source terms and unknown terms:

$$\mathbf{B} = \mathbf{rot} (\mathbf{A} + \mathbf{K} \phi) \quad (3.44)$$

$$\mathbf{E} = -\mathbf{grad} (\varphi + \alpha V) - \frac{\partial (\mathbf{A} + \mathbf{K} \phi)}{\partial t} \quad (3.45)$$

La formulation $\mathbf{A} - \varphi$ à flux et tensions imposés s'écrit alors :

$$\begin{cases} \mathbf{rot} \frac{1}{\mu} \mathbf{rot} (\mathbf{A} + \mathbf{K} \phi) = -\sigma \left(\frac{\partial (\mathbf{A} + \mathbf{K} \phi)}{\partial t} + \mathbf{grad} (\varphi + \alpha V) \right) \\ \text{div } \sigma \left(\frac{\partial (\mathbf{A} + \mathbf{K} \phi)}{\partial t} + \mathbf{grad} (\varphi + \alpha V) \right) = 0 \end{cases} \quad (3.46)$$

3.5.1.3 Imposition of a magnetomotive force and an electric current in a solid conductor

To determine or impose a magnetic potential difference ε as well as a current I , the power balance is used, expressed as a function of either electrical values (3.47), or magnetic values (3.48) [Henneron 2004] [Henneron et al 2005]

$$P_e = \int_{\mathcal{D}_c} \mathbf{E} \cdot \mathbf{J} d\tau + \int_{\mathcal{D}} \frac{\partial \mathbf{B}}{\partial t} \cdot \mathbf{H} d\tau = V i \quad (3.47)$$

$$P_m = \int_{\mathcal{D}_c} \mathbf{E} \cdot \mathbf{J} d\tau + \int_{\mathcal{D}} \frac{\partial \mathbf{B}}{\partial t} \cdot \mathbf{H} d\tau = \varepsilon \frac{d\varphi}{dt} \quad (3.48)$$

The electrical power balance is established by considering a system powered by an electrical source, and conversely, the magnetic power balance is established by considering a system powered by a magnetic source.

From these power balances and using the method of mean weighted residuals, it can be shown that current I and the magnetic potential difference ε can be expressed in the following forms using the introduced source potentials:

$$I = \int_{\mathcal{D}_c} \beta \cdot \mathbf{J} d\tau \quad (3.49)$$

$$\varepsilon = \int_{\mathcal{D}} \mathbf{H} \cdot \mathbf{N} d\tau - \int_{\mathcal{D}_c} \mathbf{K} \cdot \mathbf{J} d\tau \quad (3.50)$$

Equation 3.49 is derived from the electric power balance 3.47 and equation 3.50 is derived from the magnetic power balance 3.48. The potential formulation $\mathbf{A} - \varphi$ at current I and imposed

magnetomotive force ε is obtained from the system of equations 3.46 by adding equations 3.49 and 3.50:

$$\left\{ \begin{array}{l} \text{rot } \frac{1}{\mu} \text{rot } (\mathbf{A} + \mathbf{K} \phi) = -\sigma \left(\frac{\partial (\mathbf{A} + \mathbf{K} \phi)}{\partial t} + \mathbf{grad} (\varphi + \alpha V) \right) \\ \text{div } \sigma \left(-\frac{\partial (\mathbf{A} + \mathbf{K} \phi)}{\partial t} - \mathbf{grad} (\varphi + \alpha V) \right) = 0 \\ \int_{\mathcal{D}_c} \boldsymbol{\beta} \cdot \boldsymbol{\sigma} \left(-\frac{\partial (\mathbf{A} + \mathbf{K} \phi)}{\partial t} - \mathbf{grad} (\varphi + \alpha V) \right) d\tau = I \\ \int_{\mathcal{D}} \frac{1}{\mu} \text{rot } (\mathbf{A} + \mathbf{K} \phi) \cdot \mathbf{N} d\tau - \frac{\partial}{\partial t} \int_{\mathcal{D}_c} \mathbf{K} \cdot \boldsymbol{\sigma} \left(-\frac{\partial (\mathbf{A} + \mathbf{K} \phi)}{\partial t} - \mathbf{grad} (\varphi + \alpha V) \right) d\tau = \varepsilon \end{array} \right. \quad (3.51)$$

3.5.2 Formulation $\mathbf{T} - \Omega$

Unlike the previous formulation, the values that appear naturally in this formulation are the current I and the magnetic potential difference ε .

3.5.2.1 Imposition of a magnetomotive force and an electric current in a solid conductor

To show the current I and the magnetic potential difference ε in this formulation, the current density \mathbf{J} and the magnetic field \mathbf{H} are broken down into source terms and unknown terms. These values are expressed as a function of the fields \mathbf{N} , \mathbf{K} , $\boldsymbol{\beta}$ and α (3.52 and 3.53):

$$\mathbf{J} = \text{rot } (\mathbf{K} I + \mathbf{T}) \quad (3.52)$$

$$\mathbf{H} = \mathbf{K} I + \mathbf{T} - \mathbf{grad} (\Omega + \varepsilon \alpha) \quad (3.53)$$

These source fields naturally show these overall values within the formulation $\mathbf{T} - \Omega$:

$$\left\{ \begin{array}{l} \text{rot } \frac{1}{\sigma} \text{rot } (\mathbf{T} + \mathbf{K} I) = -\frac{\partial}{\partial t} \mu (\mathbf{K} I + \mathbf{T} - \mathbf{grad} (\Omega + \varepsilon \alpha)) \\ \text{div } \mu (\mathbf{K} I + \mathbf{T} - \mathbf{grad} (\Omega + \varepsilon \alpha)) = 0 \end{array} \right. \quad (3.54)$$

3.5.2.2 Imposition of a flux and a voltage

To impose or calculate a flux ϕ or a voltage V , the following equations are established from the electric and magnetic power balances:

$$\phi = \int_{\mathcal{D}} \boldsymbol{\beta} \cdot \mathbf{B} d\tau \quad (3.55)$$

$$V = \int_{\mathcal{D}_c} \mathbf{E} \cdot \mathbf{N} d\tau + \frac{\partial}{\partial t} \int_{\mathcal{D}} \mathbf{K} \cdot \mathbf{B} d\tau \quad (3.56)$$

These last equations lead to the formulation $\mathbf{T} - \Omega$ at imposed flux and voltage by defining the current I and the magnetic potential difference ε as unknowns:

$$\left\{ \begin{array}{l} \mathbf{rot} \frac{1}{\sigma} \mathbf{rot} (\mathbf{T} + \mathbf{K} I) = -\frac{\partial}{\partial t} \mu (\mathbf{K} I + \mathbf{T} - \mathbf{grad} (\Omega + \varepsilon \alpha)) \\ \operatorname{div} \mu (\mathbf{K} I + \mathbf{T} - \mathbf{grad} (\Omega + \varepsilon \alpha)) = 0 \\ \int_{\mathcal{D}} \beta \cdot \mu (\mathbf{K} I + \mathbf{T} - \mathbf{grad} (\Omega + \varepsilon \alpha)) d\tau = \phi \\ \int_{\mathcal{D}_e} \frac{1}{\sigma} \mathbf{rot} (\mathbf{T} + \mathbf{K} I) \cdot \mathbf{N} d\tau - \frac{\partial}{\partial t} \int_{\mathcal{D}} \mathbf{K} \cdot \mu (\mathbf{K} I + \mathbf{T} - \mathbf{grad} (\Omega + \varepsilon \alpha)) d\tau = V \end{array} \right. \quad (3.57)$$

The vector fields \mathbf{N} , \mathbf{K} , β and function α are used to impose several values. Depending on the nature of the problem and the formulation used, it is possible to impose either a magnetic flux ϕ or an electric flux I .

Chapter 4

Dealing with regions that are not simply connected

Summary

If the geometry is not simply connected, the formulations given above are no longer valid. The tools presented in the previous chapter, the vector fields \mathbf{N} and \mathbf{K} , allow clarification of the formulations.

4.1 Electrokinetics

In the case of an electrokinetic problem, the presence of holes in the conductive domain leads to modelling difficulties. The domain is no longer simply connected. Consider the conductive domain shown in Figure 4.1 through which a current I is to flow via Γ_{E1} and Γ_{E2} .

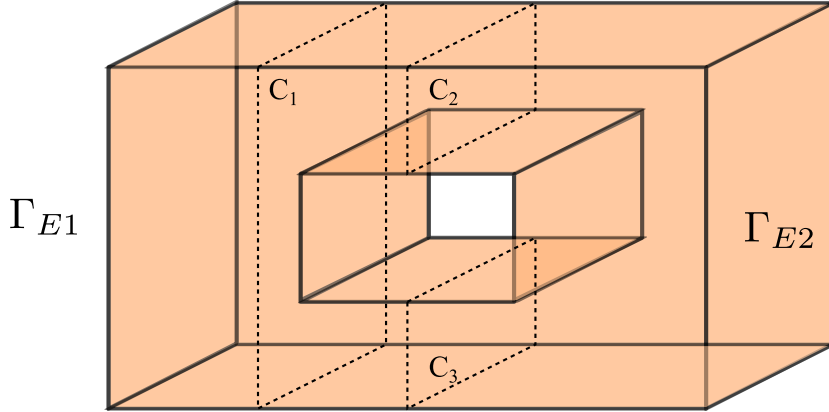


Figure 4.1: Electrokinetic example of a domain that is not simply connected

The scalar potential formulation φ does not require any special precautions to be taken to resolve the problem. Indeed, whatever the non-closed path γ between Γ_{E1} and Γ_{E2} , the condition 3.20 is validated.

$$V = \int_{\mathcal{D}_c} \mathbf{E} \cdot \mathbf{N} d\mathcal{D}_c \quad (3.20)$$

However, for the vector potential formulation, considering the paths C_1 , C_2 , C_3 , it must be checked that:

$$I = \int_{C_1} \mathbf{J} \cdot d\mathbf{s} \quad I_1 = \int_{C_2} \mathbf{J} \cdot d\mathbf{s} \quad I_2 = \int_{C_3} \mathbf{J} \cdot d\mathbf{s} \quad (4.1)$$

With:

$$I = I_1 + I_2 \quad (4.2)$$

However, neither I_1 nor I_2 is known. To impose current I , a vector field \mathbf{N} is used. It is defined such that:

$$\int_{C_1} \mathbf{N} I \cdot d\mathbf{s} = I \quad (4.3)$$

To account for the fact that part of the current flows through the surfaces defined by C_2 and C_3 , a field \mathbf{N}' is used. This second source term, combined with a second current I' , follows a hole contour (see Figure 4.2) and validates the following conditions:

$$\int_{C_1} \mathbf{N}' I' \cdot d\mathbf{s} = 0 \quad \int_{C_2} \mathbf{N}' I' \cdot d\mathbf{s} = -I' \quad \int_{C_3} \mathbf{N}' I' \cdot d\mathbf{s} = I' \quad (4.4)$$

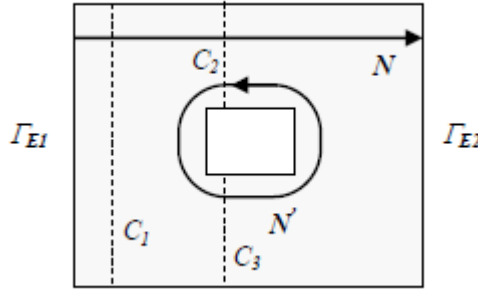


Figure 4.2: Source terms \mathbf{N} and \mathbf{N}' take the non-connectedness into account

Current I' becomes an additional unknown in the problem. This second source term can be compared to a short-circuited conductor with zero voltage imposed at its terminals. The voltage is imposed by verifying equation 3.20

$$V = \int_{\mathcal{D}_c} \mathbf{E} \cdot \mathbf{N} d\mathcal{D}_c \quad (3.20)$$

And it can be expressed as a function of vector field \mathbf{N}' :

$$\oint_C \mathbf{E} \cdot d\mathbf{l} = \int_{\mathcal{D}_c} \mathbf{E} \cdot \mathbf{N}' d\tau = 0 \quad (4.5)$$

The source fields \mathbf{N} and \mathbf{N}' thus defined verify the following conditions:

$$\begin{aligned} \int_{C_1} \mathbf{N} I \cdot d\mathbf{s} &= \oint_{C_1} \mathbf{K} I \cdot d\mathbf{l} = I \\ \int_{C_2} (\mathbf{N} I + \mathbf{N}' I') \cdot d\mathbf{s} &= \oint_{C_2} (\mathbf{K} I + \mathbf{K}' I') \cdot d\mathbf{l} = I - I' = I_1 \\ \int_{C_3} \mathbf{N}' I' \cdot d\mathbf{s} &= \oint_{C_3} \mathbf{K}' I' \cdot d\mathbf{l} = I' = I_2 \end{aligned} \quad (4.6)$$

The boundary Γ_J is not simply connected but the source vector potentials $\mathbf{K} I$ and $\mathbf{K}' I'$ perform the cut-out role previously defined.

In this configuration, the vector potential formulation is written:

$$\left\{ \begin{array}{lcl} \mathbf{rot} \frac{1}{\sigma} \mathbf{rot} (\mathbf{T}_I + \mathbf{K} I + \mathbf{K}' I') & = & 0 \\ \int_{\mathcal{D}_c} \frac{1}{\sigma} \mathbf{rot} (\mathbf{T}_I + \mathbf{K} I + \mathbf{K}' I') \cdot \mathbf{N} d\tau & = & V_1 \\ \int_{\mathcal{D}_c} \frac{1}{\sigma} \mathbf{rot} (\mathbf{T}_I + \mathbf{K} I + \mathbf{K}' I') \cdot \mathbf{N}' d\tau & = & 0 \end{array} \right. \quad (4.7)$$

Currents I and I' are two additional unknowns in this case.

If the system has several holes, a corresponding number of source terms are added, hence increasing the number of additional equations 4.5 to be verified.

4.2 Magnetostatics

For cases where the domains under study are not simply connected, the use of vector fields \mathbf{K} and \mathbf{N} is thus necessary. Cases that are not simply connected in magnetostatics are dealt with in the same way as the electrokinetic case.

Chapter 5

Weak form of the equations

Summary

The preceding chapters provided the main equations for each of the applications processed using code_Carmel to date: electrokinetic, magnetostatic and magnetodynamic. To obtain systems that are easier to use, a weak form of these equations is applied. This chapter details the transition to weak form for all applications targeted by code_Carmel: magnetodynamic, magnetostatic and electrokinetic.

5.1 Function spaces

Maxwell's equations form a set of partial derivative equations to which different operators are applied, notably curl **rot**, divergence **div** and gradient **grad**. To construct the variational formulations for solving the equations, it is thus necessary to construct spaces in which these operations are well defined.

5.1.1 Definitions

Let \mathcal{D} be a bounded open set of \mathbb{R}^3 of boundary Γ and let \mathbf{n} be the outgoing normal to \mathcal{D} . $L^2(\mathcal{D})$ is the function space of the square-integrable scalar functions on \mathcal{D} .

$$L^2(\mathcal{D}) = \left\{ X \text{ measurable ; } \int_{\mathcal{D}} |X|^2 < +\infty \right\} \quad (5.1)$$

Further, $\mathbf{L}^2(\mathcal{D})$ is the function space of square-integrable vector fields on \mathcal{D} .

$$\mathbf{L}^2(\mathcal{D}) = \left\{ \mathbf{X} \text{ measurable ; } \int_{\mathcal{D}} \|\mathbf{X}\|^2 < +\infty \right\} \quad (5.2)$$

where $\|\cdot\|$ is the conventional Euclidean norm of \mathbb{R}^3 , associated with the usual scalar product $\mathbf{a} \cdot \mathbf{b}$ of two vectors of \mathbb{R}^3 , \mathbf{a} and \mathbf{b} .

More regular function sub-spaces, so that the energy retains a finite value, can be introduced by adding a constraint relative to each operator (**grad**, **rot** and **div**).

$$H(\mathbf{grad}, \mathcal{D}) = \{ X \in L^2(\mathcal{D}) ; \mathbf{grad} X \in \mathbf{L}^2(\mathcal{D}) \}$$

$$H(\mathbf{rot}, \mathcal{D}) = \{ \mathbf{X} \in \mathbf{L}^2(\mathcal{D}) ; \mathbf{rot} \mathbf{X} \in \mathbf{L}^2(\mathcal{D}) \}$$

$$H(\mathbf{div}, \mathcal{D}) = \{ \mathbf{X} \in \mathbf{L}^2(\mathcal{D}) ; \mathbf{div} \mathbf{X} \in L^2(\mathcal{D}) \}$$

In this case, the vector fields \mathbf{H} , \mathbf{B} , \mathbf{J} , \mathbf{E} belong to $\mathbf{L}^2(\mathcal{D})$.

In $H(\mathbf{grad}, \mathcal{D})$, X is continuous at each point of \mathcal{D} . However, for $\mathbf{H}(\mathbf{rot}, \mathcal{D})$, the tangential component of \mathbf{X} is continuous on \mathcal{D} , and for $\mathbf{H}(\mathbf{div}, \mathcal{D})$, the normal component of \mathbf{X} is continuous.

By introducing the boundary conditions, the spaces are restricted:

$$\begin{aligned} H_0(\mathbf{grad}, \mathcal{D}) &= \{X \in L^2(\mathcal{D}); \mathbf{grad} X \in \mathbf{L}^2(\mathcal{D}); X = 0|_{\Gamma}\} \\ \mathbf{H}_0(\mathbf{rot}, \mathcal{D}) &= \{\mathbf{X} \in \mathbf{L}^2(\mathcal{D}); \mathbf{rot} \mathbf{X} \in \mathbf{L}^2(\mathcal{D}); \mathbf{X} \times \mathbf{n} = \mathbf{0}|_{\Gamma}\} \\ \mathbf{H}_0(\mathbf{div}, \mathcal{D}) &= \{\mathbf{X} \in \mathbf{L}^2(\mathcal{D}); \mathbf{div} \mathbf{X} \in L^2(\mathcal{D}); \mathbf{X} \cdot \mathbf{n} = 0|_{\Gamma}\} \end{aligned}$$

In this case, Γ represents the entire area containing \mathcal{D} . In mathematics, the spaces described above are conventional or “reasonable” [Costabel] as the boundary conditions are uniform. However, in physics we generally apply conditions of symmetry or impose boundary conditions on fields of different types. The boundary condition is imposed on part of Γ , here denoted Γ_x . In this case, solutions are sought in a wider space:

$$\begin{aligned} H_{0,x}(\mathbf{grad}, \mathcal{D}) &= \{X \in L^2(\mathcal{D}); \mathbf{grad} X \in \mathbf{L}^2(\mathcal{D}); X = 0|_{\Gamma_x}\} \\ \mathbf{H}_{0,x}(\mathbf{rot}, \mathcal{D}) &= \{\mathbf{X} \in \mathbf{L}^2(\mathcal{D}); \mathbf{rot} \mathbf{X} \in \mathbf{L}^2(\mathcal{D}); \mathbf{X} \times \mathbf{n} = \mathbf{0}|_{\Gamma_x}\} \\ \mathbf{H}_{0,x}(\mathbf{div}, \mathcal{D}) &= \{\mathbf{X} \in \mathbf{L}^2(\mathcal{D}); \mathbf{div} \mathbf{X} \in L^2(\mathcal{D}); \mathbf{X} \cdot \mathbf{n} = 0|_{\Gamma_x}\} \end{aligned}$$

We also introduce the function space $L^2(\mathcal{T})$ of square-integrable scalar functions over the time interval \mathcal{T} . This leads to introduction of the spaces ¹:

$$\begin{aligned} \mathcal{S}_x^0(\mathcal{D}) &= H_{0,x}(\mathbf{grad}, \mathcal{D}) \otimes L^2(\mathcal{T}) \\ \mathcal{S}_x^1(\mathcal{D}) &= \mathbf{H}_{0,x}(\mathbf{rot}, \mathcal{D}) \otimes L^2(\mathcal{T}) \\ \mathcal{S}_x^2(\mathcal{D}) &= \mathbf{H}_{0,x}(\mathbf{div}, \mathcal{D}) \otimes L^2(\mathcal{T}) \end{aligned} \tag{5.3}$$

5.1.2 Property of continuous function spaces

In electromagnetism, when applying the gradient operator to a scalar field belonging to $H(\mathbf{grad}, \mathcal{D})$, the resulting vector field is found in $\mathbf{H}(\mathbf{rot}, \mathcal{D})$. The same is found when applying the curl operator to a field belonging to $\mathbf{H}(\mathbf{rot}, \mathcal{D})$. The resulting field is found in $\mathbf{H}(\mathbf{div}, \mathcal{D})$ ².

Under these conditions, we can create a sequence of function spaces connected by the differential operators. In addition, there are inclusion properties for function spaces as shown in [Bossavit 1993]:

$$\begin{aligned} \text{Im}(\mathbf{H}(\mathbf{rot}, \mathcal{D})) &\subset \text{Ker}(\mathbf{H}(\mathbf{div}, \mathcal{D})) \\ \text{Im}(H(\mathbf{grad}, \mathcal{D})) &\subset \text{Ker}(\mathbf{H}(\mathbf{rot}, \mathcal{D})) \end{aligned}$$

If domain \mathcal{D} is simply connected and Γ is connected, the inclusions become equal:

$$\begin{aligned} \text{Im}(\mathbf{H}(\mathbf{rot}, \mathcal{D})) &= \text{Ker}(\mathbf{H}(\mathbf{div}, \mathcal{D})) \\ \text{Im}(H(\mathbf{grad}, \mathcal{D})) &= \text{Ker}(\mathbf{H}(\mathbf{rot}, \mathcal{D})) \end{aligned} \tag{5.4}$$

As these spaces include the H_0 spaces described above, we can write:

$$\begin{aligned} H_0(\mathbf{grad}, \mathcal{D}) &\subset H_{0,x}(\mathbf{grad}, \mathcal{D}) \\ \mathbf{H}_0(\mathbf{rot}, \mathcal{D}) &\subset \mathbf{H}_{0,x}(\mathbf{rot}, \mathcal{D}) \\ \mathbf{H}_0(\mathbf{div}, \mathcal{D}) &\subset \mathbf{H}_{0,x}(\mathbf{div}, \mathcal{D}) \end{aligned}$$

¹Generally: $L^2(D \times \Omega) \neq L^2(D) \times \Omega$, but $L^2(D \times \Omega) = L^2(D) \otimes L^2(\Omega)$

²it will be recalled that $\mathbf{rot}[\mathbf{grad}] = \mathbf{0}$ and $\mathbf{div}[\mathbf{rot}] = 0$.

5.1.3 Electromagnetic fields

From these definitions and in the most general case (spectral version), it can be established that:

$$\begin{aligned}\mathbf{B} &\in \mathcal{S}_b^2(\mathcal{D}) \\ \mathbf{J} &\in \mathcal{S}_h^2(\mathcal{D}_c) \\ \mathbf{H} &\in \mathcal{S}_h^1(\mathcal{D}) \\ \mathbf{E} &\in \mathcal{S}_b^1(\mathcal{D}_c)\end{aligned}\tag{5.5}$$

If the time dimension is not dealt with using the finite element method (more exactly the Galerkin projection, in the case of the time-based version), set membership is restricted to:

$$\begin{aligned}\mathbf{B} &\in \mathbf{H}_{0,b}(\operatorname{div}, \mathcal{D}) \\ \mathbf{J} &\in \mathbf{H}_{0,h}(\operatorname{div}, \mathcal{D}_c) \\ \mathbf{H} &\in \mathbf{H}_{0,h}(\mathbf{rot}, \mathcal{D}) \\ \mathbf{E} &\in \mathbf{H}_{0,b}(\mathbf{rot}, \mathcal{D}_c)\end{aligned}\tag{5.6}$$

5.1.4 Potential

Let us apply the same reasoning to scalar potentials and vector potentials. For the vector potential \mathbf{A} , defined from equation 2.29, it belongs to the same spaces as \mathbf{H} .

$$\mathbf{A} \in \mathbf{H}_{0,b}(\mathbf{rot}, \mathcal{D})\tag{5.7}$$

However, if we apply the Coulomb gauge:

$$\operatorname{div} \mathbf{A} = 0$$

it must meet the conditions on both $\mathbf{H}(\operatorname{div}, \mathcal{D})$ and $\mathbf{H}(\mathbf{rot}, \mathcal{D})$. Finally, if we apply a gauge of this type:

$$(\mathbf{A}, \mathbf{grad} \xi) = 0, \quad \forall \xi \in H_0^1(\mathcal{D})$$

Then we must define two new spaces such that:

$$\begin{aligned}P_0(\mathcal{D}) &= \{\mathbf{X} \in \mathbf{H}_0(\mathbf{rot}, \mathcal{D}); (\mathbf{X}, \mathbf{grad} \xi) = 0; \forall \xi \in H_0(\mathbf{grad}, \mathcal{D})\} \\ P_{0,x}(\mathcal{D}) &= \{\mathbf{X} \in \mathbf{H}_{0,x}(\mathbf{rot}, \mathcal{D}); (\mathbf{X}, \mathbf{grad} \xi) = 0; \forall \xi \in H_{0,x}(\mathbf{grad}, \mathcal{D})\}\end{aligned}\tag{5.8}$$

Using these two new spaces, it is possible to establish that:

$$\mathbf{A} \in P_0(\mathcal{D})\tag{5.9}$$

or

$$\mathbf{A} \in P_{0,x}(\mathcal{D})\tag{5.10}$$

But, unlike spaces $\mathbf{H}_0(\mathbf{rot}, \mathcal{D})$ and $\mathbf{H}_{0,x}(\mathbf{rot}, \mathcal{D})$, it is not possible to establish that $P_0(\mathcal{D})$ is included in $P_{0,x}(\mathcal{D})$. Indeed, the gauge described in $P_{0,x}(\mathcal{D})$ requires that ξ must satisfy more constraints because $\mathbf{H}_{0,x}(\mathbf{rot}, \mathcal{D})$ contains $\mathbf{H}_0(\mathbf{rot}, \mathcal{D})$.

As a result, the space that satisfies:

$$\{(X, \mathbf{grad} \xi) = 0; \forall \xi \in H_{0,x}(\mathbf{grad}, \mathcal{D})\}$$

contains the space:

$$\{(X, \mathbf{grad} \xi) = 0; \forall \xi \in H_0(\mathbf{grad}, \mathcal{D})\}$$

The vector electric potential \mathbf{T} has the same properties as \mathbf{A} but exists only in \mathcal{D}_c . Hence, depending on the gauge used, this gives either:

$$\mathbf{T} \in \mathbf{H}_{0,x}(\text{div}, \mathcal{D}_c) \wedge \mathbf{H}_{0,h}(\mathbf{rot}, \mathcal{D}_c) \quad \text{for Coulomb gauge} \quad (5.11)$$

namely:

$$\mathbf{T} \in P_0(\mathcal{D}_c) \quad \text{for Coulomb gauge} \quad (5.12)$$

or:

$$\mathbf{T} \in P_{0,h}(\mathcal{D}_c) \quad \text{for the other gauge} \quad (5.13)$$

Let us finish with the scalar potentials φ and Ω which are in $H_{0,b}(\mathbf{grad}, \mathcal{D}_c)$ and $H_{0,h}(\mathbf{grad}, \mathcal{D})$ respectively.

5.2 Projection principles

The formulations can be written as follows:

$$\mathcal{L}(U) + f = 0 \quad \text{in } \mathcal{D} \quad (5.14)$$

$$\mathcal{C}(U) + f_s = 0 \quad \text{on } \Gamma \quad (5.15)$$

where \mathcal{L} and \mathcal{C} are operators, while f and f_s represent source terms which are generally known.

Applying the method of mean weighted residuals [Dhatt, Thouzot 1984] gives the following integral forms depending on the type of test function. If time dependency is explicitly considered (spectral version):

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \cdot (\mathcal{L}(U) + f) d\mathcal{D} = 0 \quad (5.16)$$

If this is not the case (time-based version):

$$\int_{\mathcal{D}} \mathcal{U} \cdot (\mathcal{L}(U) + f) d\mathcal{D} = 0 \quad (5.17)$$

If \mathcal{U} is a solution of equation 5.16 or 5.17 and validates the boundary conditions defined by 5.15 for all test functions \mathcal{U} then \mathcal{U} is also a solution of equation 5.14. However, to reduce the order of differentiation, we often use integration by parts. This results in the weak integral forms that we are usually required to solve:

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \cdot \mathcal{L}_D(U) d\mathcal{D} + \int_{\mathcal{T}} \int_{\Gamma} \mathcal{U} \cdot \mathcal{S}_L(U) d\gamma + \int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} f d\mathcal{D} = 0 \quad (5.18)$$

$$\int_{\mathcal{D}} \mathcal{U} \cdot \mathcal{L}_D(U) d\mathcal{D} + \int_{\Gamma} \mathcal{U} \cdot \mathcal{S}_L(U) d\gamma + \int_{\mathcal{D}} \mathcal{U} f d\mathcal{D} = 0 \quad (5.19)$$

where \mathcal{L}_D and \mathcal{S}_L are operators.

To solve the formulations in their weak forms in the function spaces defined above, we then take the test functions \mathcal{U} equal to the interpolation functions (Galerkin method).

5.3 Magnetodynamic problem

5.3.1 Formulation A - φ

We apply the previous method to magnetodynamic equations with a properly chosen test function \mathcal{U} .

5.3.1.1 Projection in space only

Formulation A - φ is written as follows without the time dimension (time-based version):

$$\int_{\mathcal{D}} \mathcal{U} \cdot \left[\mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{A} + \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \right] d\mathcal{D} = \int_{\mathcal{D}} \mathcal{U} \cdot \left[\mathbf{J}_s + \frac{1}{\mu} \mathbf{rot} \mathbf{B}_r \right] d\mathcal{D} \quad (5.20)$$

First, a test function is chosen for the vector magnetic potential:

$$\mathcal{U} = \mathbf{A}' \quad \mathbf{A}' \in \mathbf{H}_{0,b}(\mathbf{rot}, \mathcal{D})$$

The first part is integrated by parts.

$$\int_{\mathcal{D}} \mathbf{A}' \cdot \mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{A} d\mathcal{D} = \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{A} \cdot \mathbf{rot} \mathbf{A}' d\mathcal{D} + \int_{\partial \mathcal{D}} \left(\frac{1}{\mu} \mathbf{rot} \mathbf{A} \times \mathbf{A}' \right) \cdot d\boldsymbol{\gamma} \quad (5.21)$$

where $\partial \mathcal{D} = \Gamma$ is the edge of \mathcal{D} .

If the outgoing normal is denoted \mathbf{n} , letting $d\boldsymbol{\gamma} = \mathbf{n} \cdot d\boldsymbol{\gamma}$, the previous equation becomes:

$$\int_{\mathcal{D}} \mathbf{A}' \cdot \mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{A} d\mathcal{D} = \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{A} \cdot \mathbf{rot} \mathbf{A}' d\mathcal{D} - \int_{\Gamma} \left(\frac{1}{\mu} \mathbf{rot} \mathbf{A} \times \mathbf{n} \right) \cdot \mathbf{A}' d\gamma \quad (5.22)$$

Similarly, we can rewrite the term in \mathbf{B}_r .

$$\int_{\mathcal{D}} \mathbf{A}' \cdot \frac{1}{\mu} \mathbf{rot} \mathbf{B}_r d\mathcal{D} = \int_{\mathcal{D}} \mathbf{rot} \mathbf{A}' \cdot \frac{1}{\mu} \mathbf{B}_r d\mathcal{D} - \int_{\Gamma} \left(\frac{1}{\mu} \mathbf{B}_r \times \mathbf{n} \right) \cdot \mathbf{A}' d\gamma \quad (5.23)$$

We then obtain, with $\mathbf{H} = \frac{1}{\mu} \mathbf{rot} \mathbf{A} - \frac{1}{\mu} \mathbf{B}_r$:

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{A} \cdot \mathbf{rot} \mathbf{A}' + \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathbf{A}' \right] d\mathcal{D} = \\ \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{A}' d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{A}' \cdot \mathbf{B}_r d\mathcal{D} + \int_{\Gamma} (\mathbf{H} \times \mathbf{n}) \cdot \mathbf{A}' d\gamma \end{aligned} \quad (5.24)$$

Recalling that (see paragraph 1.7):

$$\Gamma = \Gamma_B \cup \Gamma_H$$

On Γ_B , a uniform Dirichlet condition is imposed on \mathbf{A} (and the same on \mathbf{A}'):

$$\mathbf{A} \times \mathbf{n} = \mathbf{0} \quad \text{sur } \Gamma_B \quad (5.25)$$

Thus, $\mathbf{B} \cdot \mathbf{n} = 0$.

On Γ_H , a uniform Neumann condition is imposed on \mathbf{A} :

$$\mathbf{H} \times \mathbf{n} = \mathbf{0} \quad (5.26)$$

Thus, by imposing a proper test function, the edge term (on Γ) is cancelled out.

Secondly, a test function is chosen for the scalar electric potential:

$$\mathcal{U} = \mathbf{grad}\varphi' \quad \varphi' \in H_{0,b}(\mathbf{grad}, \mathcal{D}_c)$$

Given that:

$$\int_{\mathcal{D}} \mathbf{grad}\varphi' \cdot \left[\mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{A} + \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \right] d\mathcal{D} - \int_{\Gamma} \mathbf{grad}\varphi' \cdot (\mathbf{n} \times \mathbf{H}) d\gamma = 0$$

Hence:

$$\int_{\mathcal{D}} \sigma \mathbf{grad}\varphi' \cdot \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) d\mathcal{D} - \int_{\Gamma} \mathbf{grad}\varphi' \cdot \left(\mathbf{n} \times \left(\frac{1}{\mu} \mathbf{rot} \mathbf{A} \right) \right) d\gamma = 0 \quad (5.27)$$

The surface term is:

$$\begin{aligned} \int_{\Gamma} \mathbf{grad}\varphi' \cdot \left(\mathbf{n} \times \left(\frac{1}{\mu} \mathbf{rot} \mathbf{A} \right) \right) d\gamma &= - \int_{\Gamma} \varphi' \operatorname{div} \left(\mathbf{n} \times \frac{1}{\mu} \mathbf{rot} \mathbf{A} \right) d\gamma \\ &= \int_{\Gamma} \varphi' \mathbf{rot} \left(\frac{1}{\mu} \mathbf{rot} \mathbf{A} \right) \cdot \mathbf{n} d\gamma \\ &= - \int_{\Gamma} \varphi' \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathbf{n} d\gamma \end{aligned}$$

It is recalled that:

$$\mathbf{J}_{ind} = \sigma \mathbf{E} = -\sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right)$$

As we have demonstrated above, the surface integrals disappear. This amounts to a strong imposition of boundary conditions on Γ_b ($\mathbf{E} \times \mathbf{n} = 0$ and $\mathbf{B} \cdot \mathbf{n} = 0$) and weak imposition on Γ_h ($\mathbf{H} \times \mathbf{n} = 0$ and $\mathbf{J} \cdot \mathbf{n} = 0$).

Hence, equation (5.27) is written:

$$\int_{\mathcal{D}} \sigma \mathbf{grad}\varphi' \cdot \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) d\mathcal{D} = 0 \quad (5.28)$$

The first equation corresponds to the flux conservation of the current density and the second to Ampère's circuital law.

The system to be solved in magnetodynamics without time projection is as follows:

Find $\mathbf{A} \in P_{0,x}(\mathcal{D})$ and $\varphi \in H_{0,b}(\mathbf{grad}, \mathcal{D}_c)$ such that $\forall \mathbf{A}' \in \mathbf{H}_{0,b}(\mathbf{rot}, \mathcal{D})$, $\forall \varphi' \in H_{0,b}(\mathbf{grad}, \mathcal{D}_c)$

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{A}' \cdot \mathbf{rot} \mathbf{A} + \sigma \mathbf{A}' \cdot \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \right] d\mathcal{D} &= \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{A}' d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{A}' \cdot \mathbf{B}_r d\mathcal{D} \\ \int_{\mathcal{D}} \sigma \mathbf{grad}\varphi' \cdot \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) d\mathcal{D} &= 0 \end{aligned} \quad (5.29)$$

5.3.1.2 Projection in space and time

If the time dimension is to be dealt with (spectral version), the weak formulation in $\mathbf{A} - \varphi$ is written as follows, according to equation 5.20:

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \cdot \left[\mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{A} + \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \right] d\mathcal{D} = \int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \cdot \left[\mathbf{J}_s + \frac{1}{\mu} \mathbf{B}_r \right] d\mathcal{D} \quad (5.30)$$

The first part is integrated by parts.

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \cdot \mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{A} d\mathcal{D} = \int_{\mathcal{T}} \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{A} \cdot \mathbf{rot} \mathcal{U} d\mathcal{D} + \int_{\mathcal{T}} \int_{\partial \mathcal{D}} \left(\frac{1}{\mu} \mathbf{rot} \mathbf{A} \times \mathcal{U} \right) \cdot d\gamma \quad (5.31)$$

where $\partial \mathcal{D} = \Gamma$ is the edge of \mathcal{D} .

If the outgoing normal is denoted \mathbf{n} , letting $d\gamma = \mathbf{n} \cdot d\gamma$, the previous equation becomes:

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \cdot \mathbf{rot} \frac{1}{\mu} \mathbf{rot} \mathbf{A} d\mathcal{D} = \int_{\mathcal{T}} \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{A} \cdot \mathbf{rot} \mathcal{U} d\mathcal{D} - \int_{\mathcal{T}} \int_{\Gamma} \left(\frac{1}{\mu} \mathbf{rot} \mathbf{A} \times \mathbf{n} \right) \cdot \mathcal{U} d\gamma \quad (5.32)$$

Similarly, we can rewrite the term in \mathbf{B}_r :

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \cdot \frac{1}{\mu} \mathbf{rot} \mathbf{B}_r d\mathcal{D} = \int_{\mathcal{T}} \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathcal{U} \cdot \mathbf{B}_r d\mathcal{D} - \int_{\mathcal{T}} \int_{\Gamma} \left(\frac{1}{\mu} \mathbf{B}_r \times \mathbf{n} \right) \cdot \mathcal{U} d\gamma \quad (5.33)$$

This gives:

$$\begin{aligned} \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{A} \cdot \mathbf{rot} \mathcal{U} + \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathcal{U} \right] d\mathcal{D} = \\ \int_{\mathcal{T}} \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathcal{U} d\mathcal{D} + \int_{\mathcal{T}} \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{rot} \mathcal{U} d\mathcal{D} + \int_{\mathcal{T}} \int_{\Gamma} (\mathbf{H} \times \mathbf{n}) \cdot \mathcal{U} d\gamma \end{aligned} \quad (5.34)$$

First, a test function is chosen for the scalar electric potential:

$$\mathcal{U} = \mathbf{grad} \varphi' \quad \varphi' \in \mathcal{S}_{\mathbf{E}}^0$$

Hence:

$$\begin{aligned} \int_{\mathcal{T}} \int_{\mathcal{D}} \mathbf{grad} \varphi' \cdot \left[\sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \right] d\mathcal{D} - \int_{\mathcal{T}} \int_{\Gamma} \mathbf{grad} \varphi' \cdot (\mathbf{n} \times \mathbf{H}) d\gamma = \\ \int_{\mathcal{T}} \int_{\mathcal{D}} \mathbf{grad} \varphi' \cdot \mathbf{J}_s d\mathcal{D} \end{aligned}$$

We use the equation:

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \mathbf{grad} \varphi' \cdot \mathbf{v} d\mathcal{D} = - \int_{\mathcal{T}} \int_{\mathcal{D}} \varphi' \operatorname{div} \mathbf{v} d\mathcal{D} + \int_{\mathcal{T}} \int_{\Gamma} \varphi' (\mathbf{v} \cdot \mathbf{n}) d\gamma \quad (5.35)$$

This gives:

$$\begin{aligned} \int_{\mathcal{T}} \int_{\mathcal{D}} \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathbf{grad} \varphi' d\mathcal{D} = \\ - \int_{\mathcal{T}} \int_{\mathcal{D}} \operatorname{div} \mathbf{J}_s \varphi' d\mathcal{D} + \int_{\mathcal{T}} \int_{\Gamma} \varphi' (\mathbf{J}_s \cdot \mathbf{n}) d\gamma \end{aligned} \quad (5.36)$$

We have:

$$\operatorname{div} \mathbf{J}_s = 0 \quad (5.37)$$

The edge integral is cancelled out by the choice of test function.

This gives:

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \sigma \left(\frac{\mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathbf{grad} \varphi' d\mathcal{D} = 0 \quad (5.38)$$

Secondly, a test function is chosen for the vector magnetic potential:

$$\mathcal{U} = \mathbf{A}' \quad \mathbf{A}' \in \mathcal{S}_{\mathbf{E}}^1$$

This leads to the equation:

$$\begin{aligned} & \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\mu^{-1} \mathbf{rot} \mathbf{A} \cdot \mathbf{rot} \mathbf{A}' + \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathbf{A}' \right] d\mathcal{D} = \\ & \int_{\mathcal{T}} \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{A}' d\mathcal{D} + \int_{\mathcal{T}} \int_{\mathcal{D}} \mu^{-1} \mathbf{B}^r \cdot \mathbf{rot} \mathbf{A}' d\mathcal{D} + \int_{\mathcal{T}} \int_{\Gamma} \left(\mathbf{H}^{\Gamma} \times \mathbf{n} \right) \cdot \mathbf{A}' d\gamma \end{aligned} \quad (5.39)$$

The system to be solved in magnetodynamics with time projection is as follows:

Find $\mathbf{A} \in \mathcal{S}_{\mathbf{E}}^1$ and $\varphi \in \mathcal{S}_{\mathbf{E}}^0$ such that $\forall \mathbf{A}' \in \mathcal{S}_{\mathbf{E}}^1$, $\forall \varphi' \in \mathcal{S}_{\mathbf{E}}^0$

$$\begin{aligned} & \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\mu^{-1} \mathbf{rot} \mathbf{A} \cdot \mathbf{rot} \mathbf{A}' + \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathbf{A}' \right] d\mathcal{D} = \\ & \int_{\mathcal{T}} \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{A}' d\mathcal{D} + \int_{\mathcal{T}} \int_{\mathcal{D}} \mu^{-1} \mathbf{B}^r \cdot \mathbf{rot} \mathbf{A}' d\mathcal{D} + \int_{\mathcal{T}} \int_{\Gamma} \left(\mathbf{H}^{\Gamma} \times \mathbf{n} \right) \cdot \mathbf{A}' d\gamma \\ & \int_{\mathcal{T}} \int_{\mathcal{D}} \sigma \left(\frac{\mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathbf{grad} \varphi' d\mathcal{D} = 0 \end{aligned} \quad (5.40)$$

Remark 5.3.1 *The values \mathbf{A} , φ , \mathbf{A}' and φ' are time dependent.*

5.3.2 Formulation T- Ω

5.3.2.1 Projection in space only

The magnetodynamic system of equations with the formulation T- Ω is given below (in the case of the time-based version of code_Carmel):

$$\begin{aligned} \mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{T}(\mathbf{x}, t) + \frac{\partial}{\partial t} \mu (\mathbf{T}(\mathbf{x}, t) - \mathbf{grad} \Omega(\mathbf{x}, t)) = \\ - \mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s(\mathbf{x}, t) - \frac{\partial}{\partial t} (\mu \mathbf{H}_s(\mathbf{x}, t) + \mathbf{B}_r) \end{aligned} \quad (2.36)$$

$$\operatorname{div} \mu (\mathbf{T}(\mathbf{x}, t) - \mathbf{grad} \Omega(\mathbf{x}, t)) = -\operatorname{div} (\mu \mathbf{H}_s(\mathbf{x}, t) + \mathbf{B}_r) \quad (2.37)$$

The first expression is multiplied by a test function \mathcal{U} . Formulation T- Ω is written as follows without the time dimension (time-based version):

$$\begin{aligned} \int_{\mathcal{D}} \mathcal{U} \cdot \left[\mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{T} + \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} = \\ - \int_{\mathcal{D}} \mathcal{U} \cdot \left[\mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s + \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \end{aligned} \quad (5.41)$$

hence:

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \cdot \mathbf{rot} \mathcal{U} + \mathcal{U} \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} \left(\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \times \mathcal{U} \right) \cdot \mathbf{n} d\gamma = \\ - \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} \mathcal{U} + \mathcal{U} \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} + \int_{\partial \mathcal{D}} \left(\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \times \mathcal{U} \right) \cdot \mathbf{n} d\gamma \end{aligned} \quad (5.42)$$

and further:

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \cdot \mathbf{rot} \mathcal{U} + \mathcal{U} \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} \left(\frac{1}{\sigma} \mathbf{J} \times \mathbf{n} \right) \cdot \mathcal{U} d\gamma = \\ - \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} \mathcal{U} + \mathcal{U} \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \end{aligned} \quad (5.43)$$

If we initially take:

$$\mathcal{U} = T' \quad \text{avec } T' \in H_{0,h}(\mathbf{rot}, \mathcal{D}) \quad (5.44)$$

Then:

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \cdot \mathbf{rot} T' + T' \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot T' d\gamma = \\ \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} T' + T' \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \end{aligned} \quad (5.45)$$

If we secondly take:

$$\mathcal{U} = \mathbf{grad} \Omega' \quad \text{avec } \Omega' \in H_{0,h}(\mathbf{grad}, \mathcal{D}) \quad (5.46)$$

Then:

$$\begin{aligned} \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{T} \times \mathbf{grad} \Omega' d\gamma = \\ \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \times \mathbf{grad} \Omega' d\gamma \end{aligned} \quad (5.47)$$

Hence:

$$\begin{aligned} \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{grad} \Omega' d\gamma = \\ \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \end{aligned} \quad (5.48)$$

The equations to be solved are thus:

Find $\mathbf{T} \in P_{0,x}(\mathcal{D})$ and $\Omega \in H_{0,h}(\mathbf{grad}, \mathcal{D})$ such that $\forall \mathbf{T}' \in \mathbf{H}_{0,h}(\mathbf{rot}, \mathcal{D})$, $\forall \Omega' \in H_{0,h}(\mathbf{grad}, \mathcal{D}_c)$

$$\int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \cdot \mathbf{rot} \mathbf{T}' + \mathbf{T}' \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{T}' d\gamma = \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} \mathbf{T}' + \mathbf{T}' \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \quad (5.49)$$

$$\int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{grad} \Omega' d\gamma = \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \quad (5.50)$$

5.3.2.2 Projection in space and time

And, with the time dimension (with the spectral version of code_Carmel, for the moment we have $\mathbf{B}_r = \mathbf{0}$), this gives:

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \cdot \left[\mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{T} + \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} = - \int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \cdot \left[\mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s + \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \quad (5.51)$$

Hence:

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \cdot \mathbf{rot} \mathcal{U} + \mathcal{U} \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} \left(\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \times \mathcal{U} \right) \cdot \mathbf{n} d\gamma = - \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} \mathcal{U} + \mathcal{U} \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} \left(\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \times \mathcal{U} \right) \cdot \mathbf{n} d\gamma = \quad (5.52)$$

This approach is comparable to that in the preceding paragraph. The equations to be solved are thus:

Find $\mathbf{T} \in \mathcal{S}_b^1(\mathcal{D})$ and $\Omega \in \mathcal{S}_h^0(\mathcal{D})$ such that $\forall \mathbf{T}' \in \mathcal{S}_b^1(\mathcal{D})$, $\forall \Omega' \in \mathcal{S}_h^0(\mathcal{D})$

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \cdot \mathbf{rot} \mathbf{T}' + \mathbf{T}' \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{T}' d\gamma = \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} \mathbf{T}' + \mathbf{T}' \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \quad (5.53)$$

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{grad} \Omega' d\gamma = \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \quad (5.54)$$

5.4 Magnetostatic problem

Here, the terms corresponding to induced currents disappear.

5.4.1 Formulation A

5.4.1.1 Projection in space only

The integral form of the formulation to be solved is thus (see the strong form of equation 2.22):

$$\int_{\mathcal{D}} \mathcal{U} \cdot \left[\left(\text{rot} \frac{1}{\mu} \text{rot} \mathbf{A} - \mathbf{J}_s - \frac{1}{\mu} \text{rot} \mathbf{B}_r \right) \right] d\mathcal{D} = 0 \quad (5.55)$$

Hence:

$$\int_{\mathcal{D}} \frac{1}{\mu} \text{rot} \mathcal{U} \cdot \text{rot} \mathbf{A} d\mathcal{D} - \int_{\Gamma} \mathcal{U} \cdot \left(\mathbf{n} \times \frac{1}{\mu} \text{rot} \mathbf{A} \right) d\Gamma = \int_{\mathcal{D}} \mathcal{U} \cdot \mathbf{J}_s d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathcal{U} \cdot \text{rot} \mathbf{B}_r d\mathcal{D} \quad (5.56)$$

If we take:

$$\mathcal{U} = \mathbf{A}' \quad \text{avec } \mathbf{A}' \in H_{0,b}(\text{rot}, \mathcal{D}) \quad (5.57)$$

Then the system to be solved is written:

$$\int_{\mathcal{D}} \frac{1}{\mu} \text{rot} \mathbf{A}' \cdot \text{rot} \mathbf{A} d\mathcal{D} - \int_{\Gamma} \mathbf{A}' \cdot \left(\mathbf{n} \times \frac{1}{\mu} \text{rot} \mathbf{A} \right) d\Gamma = \int_{\mathcal{D}} \mathbf{A}' \cdot \mathbf{J}_s d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{A}' \cdot \text{rot} \mathbf{B}_r d\mathcal{D} \quad (5.58)$$

By integrating by parts:

$$\int_{\mathcal{D}} \frac{1}{\mu} \mathbf{A}' \cdot \text{rot} \mathbf{B}_r d\mathcal{D} = \int_{\mathcal{D}} \frac{1}{\mu} \text{rot} \mathbf{A}' \cdot \mathbf{B}_r d\mathcal{D} + \int_{\Gamma} \frac{1}{\mu} \mathbf{A}' \cdot (\mathbf{n} \times \mathbf{B}_r) d\Gamma \quad (5.59)$$

The equation becomes:

$$\int_{\mathcal{D}} \frac{1}{\mu} \text{rot} \mathbf{A}' \cdot \text{rot} \mathbf{A} d\mathcal{D} - \int_{\Gamma} \mathbf{A}' \cdot (\mathbf{n} \times \mathbf{H}) d\Gamma = \int_{\mathcal{D}} \mathbf{A}' \cdot \mathbf{J}_s d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \text{rot} \mathbf{A}' \cdot \mathbf{B}_r d\mathcal{D} \quad (5.60)$$

The boundary Γ is the union of a boundary portion where the normal component of the flux density is zero (Γ_B) and a portion where the tangential component of the magnetic field is zero (Γ_H).

With the proper choice of test function, the edge term is cancelled out.

This thus reduces to:

Find $\mathbf{A} \in P_{0,x}(\mathcal{D})$ such that $\forall \mathbf{A}' \in H_{0,b}(\text{rot}, \mathcal{D})$

$$\int_{\mathcal{D}} \frac{1}{\mu} \text{rot} \mathbf{A}' \cdot \text{rot} \mathbf{A} d\mathcal{D} = \int_{\mathcal{D}} \mathbf{A}' \cdot \mathbf{J}_s d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{A}' \cdot \text{rot} \mathbf{B}_r d\mathcal{D} \quad (5.61)$$

5.4.1.2 Projection in space and time

By removing the terms associated with induced currents and the conductive domain in equation 5.40, the system to be solved becomes:

Find $\mathbf{A} \in \mathcal{S}_{\mathbf{E}}^1$ such that $\forall \mathbf{A}' \in \mathcal{S}_{\mathbf{E}}^1$

$$\int_{\mathcal{T}} \int_{\mathcal{D}} [\mu^{-1} \mathbf{rot} \mathbf{A} \cdot \mathbf{rot} \mathbf{A}'] d\mathcal{D} = \int_{\mathcal{T}} \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{A}' d\mathcal{D} + \int_{\mathcal{T}} \int_{\mathcal{D}} \mu^{-1} \mathbf{B}^r \cdot \mathbf{rot} \mathbf{A}' d\mathcal{D} + \int_{\mathcal{T}} \int_{\Gamma_H} (\mathbf{H}^\Gamma \times \mathbf{n}) \cdot \mathbf{A}' d\gamma \quad (5.62)$$

5.4.2 Formulation Ω

5.4.2.1 Projection in space only

The integral form of the formulation to be solved is thus (see equation 2.23):

$$\int_{\mathcal{D}} \mathcal{U} [\operatorname{div} \mu (\mathbf{H}_s - \mathbf{grad} \Omega)] d\mathcal{D} = - \int_{\mathcal{D}} \mathcal{U} \operatorname{div} \mathbf{B}_r d\mathcal{D} \quad (5.63)$$

where \mathbf{H}_s , which represents the source field, is calculated from \mathbf{J}_s .

Hence:

$$\int_{\mathcal{D}} \mu (\mathbf{grad} \mathcal{U} \cdot \mathbf{grad} \Omega - \mathbf{grad} \mathcal{U} \cdot \mathbf{H}_s) d\mathcal{D} + \int_{\Gamma} \mathcal{U} (\mu \mathbf{grad} \Omega) d\gamma = - \int_{\mathcal{D}} \mathcal{U} \operatorname{div} \mathbf{B}_r d\mathcal{D} \quad (5.64)$$

However:

$$\int_{\mathcal{D}} \mathcal{U} \operatorname{div} \mathbf{B}_r d\mathcal{D} = - \int_{\mathcal{D}} \mathbf{grad} \mathcal{U} \cdot \mathbf{B}_r d\mathcal{D} \quad (5.65)$$

We take:

$$\mathcal{U} = \Omega' \quad \text{avec } \Omega' \in H_{0,h}(\mathbf{grad}, \mathcal{D}) \quad (5.66)$$

Thus, the problem is reduced to:

Find $\Omega \in H_{0,h}(\mathbf{grad}, \mathcal{D})$ such that $\forall \Omega' \in H_{0,h}(\mathbf{grad}, \mathcal{D}_c)$

$$\int_{\mathcal{D}} \mu (\mathbf{grad} \Omega' \cdot \mathbf{grad} \Omega - \mathbf{grad} \Omega' \cdot \mathbf{H}_s) d\mathcal{D} + \int_{\Gamma} \Omega' (\mu \mathbf{grad} \Omega) d\gamma = \int_{\mathcal{D}} \mathbf{grad} \Omega' \cdot \mathbf{B}_r d\mathcal{D} \quad (5.67)$$

5.4.2.2 Projection in space and time

The magnetodynamic system of equations with the formulation T- Ω is given below (with the spectral version of code_Carmel, for the moment we have $\mathbf{B}_r = \mathbf{0}$):

$$\begin{aligned} \mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{T}(\mathbf{x}, t) + \frac{\partial}{\partial t} \mu (\mathbf{T}(\mathbf{x}, t) - \mathbf{grad} \Omega(\mathbf{x}, t)) = \\ - \mathbf{rot} \frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s(\mathbf{x}, t) - \frac{\partial}{\partial t} (\mu \mathbf{H}_s(\mathbf{x}, t) + \mathbf{B}_r) \end{aligned} \quad (2.36)$$

$$\operatorname{div} \mu (\mathbf{T}(\mathbf{x}, t) - \mathbf{grad} \Omega(\mathbf{x}, t)) = -\operatorname{div} (\mu \mathbf{H}_s(\mathbf{x}, t) + \mathbf{B}_r) \quad (2.37)$$

In magnetostatics, it becomes:

$$-\operatorname{div} \mu (\mathbf{grad} \Omega(\mathbf{x})) = -\operatorname{div} (\mu \mathbf{H}_s(\mathbf{x}) + \mathbf{B}_r) \quad (5.68)$$

This expression is multiplied by a test function \mathcal{U} . The formulation Ω is written as follows (spectral version):

$$\int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \operatorname{div} \mu (\mathbf{grad} \Omega(\mathbf{x})) \, d\mathcal{D} = \int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \operatorname{div} (\mu \mathbf{H}_s(\mathbf{x}) + \mathbf{B}_r) \, d\mathcal{D} \quad (5.69)$$

Hence:

$$\begin{aligned} - \int_{\mathcal{T}} \int_{\mathcal{D}} \mu \mathbf{grad} \mathcal{U} \mathbf{grad} \Omega(\mathbf{x}) \, d\mathcal{D} = \\ \int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{U} \operatorname{div} (\mu \mathbf{H}_s(\mathbf{x}) + \mathbf{B}_r) \, d\mathcal{D} - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} \mathcal{U} \mu \mathbf{grad} \Omega(\mathbf{x}) \cdot d\partial \mathcal{D} \end{aligned} \quad (5.70)$$

And further:

$$\begin{aligned} - \int_{\mathcal{T}} \int_{\mathcal{D}} \mu \mathbf{grad} \mathcal{U} \cdot \mathbf{grad} \Omega(\mathbf{x}) \, d\mathcal{D} = - \int_{\mathcal{T}} \int_{\mathcal{D}} (\mu \mathbf{H}_s(\mathbf{x}) + \mathbf{B}_r) \cdot \mathbf{grad} \mathcal{U} \, d\mathcal{D} \\ + \int_{\mathcal{T}} \int_{\partial \mathcal{D}} \mathcal{U} (\mu \mathbf{H}_s(\mathbf{x}) + \mathbf{B}_r) \cdot d\partial \mathcal{D} - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} \mathcal{U} \mu \mathbf{grad} \Omega(\mathbf{x}) \cdot d\partial \mathcal{D} \end{aligned} \quad (5.71)$$

This thus reduces to:

$$\begin{aligned} - \int_{\mathcal{T}} \int_{\mathcal{D}} \mu \mathbf{grad} \mathcal{U} \cdot \mathbf{grad} \Omega(\mathbf{x}) \, d\mathcal{D} = - \int_{\mathcal{T}} \int_{\mathcal{D}} (\mu \mathbf{H}_s(\mathbf{x}) + \mathbf{B}_r) \cdot \mathbf{grad} \mathcal{U} \, d\mathcal{D} + \\ \int_{\mathcal{T}} \int_{\partial \mathcal{D}} \mathcal{U} \mu \mathbf{H} \cdot d\partial \mathcal{D} \end{aligned} \quad (5.72)$$

For the test function, we take:

$$\mathcal{U} = \Omega' \quad \text{avec } \Omega' \in H_{0,h}(\operatorname{grad}, \mathcal{D}) \quad (5.73)$$

As a result, the system to be resolved is:

Find $\Omega \in H_{0,h}(\operatorname{grad}, \mathcal{D})$ such that $\forall \Omega' \in H_{0,h}(\operatorname{grad}, \mathcal{D})$

$$\begin{aligned} - \int_{\mathcal{T}} \int_{\mathcal{D}} \mu \mathbf{grad} \Omega' \cdot \mathbf{grad} \Omega(\mathbf{x}) \, d\mathcal{D} = - \int_{\mathcal{T}} \int_{\mathcal{D}} (\mu \mathbf{H}_s(\mathbf{x}) + \mathbf{B}_r) \cdot \mathbf{grad} \Omega' \, d\mathcal{D} + \\ \int_{\mathcal{T}} \int_{\partial \mathcal{D}} \Omega' \mu \mathbf{H} \cdot d\partial \mathcal{D} \end{aligned} \quad (5.74)$$

5.5 Electrokinetic problem

Here, the time dimension disappears. This concerns only the time-based version.

5.5.1 Formulation φ

The scalar electric potential equation for the electrokinetic formulation is given below:

$$\operatorname{div} \sigma \mathbf{grad} \varphi = \operatorname{div} \sigma \mathbf{grad} \alpha V \quad (3.26)$$

By multiplying both sides of this equation by a test function \mathcal{U} , the integral form of the formulation is thus:

$$\int_{\mathcal{D}_c} \mathcal{U} [\operatorname{div} \sigma (\mathbf{grad} \varphi + \mathbf{grad} \alpha V)] d\mathcal{D}_c \quad (5.75)$$

The weak form of the equation is:

$$\int_{\mathcal{D}_c} \sigma \mathbf{grad} \mathcal{U} \cdot \mathbf{grad} \varphi d\mathcal{D}_c + \int_{\Gamma} \mathcal{U} (\sigma \mathbf{grad} \varphi) \cdot \mathbf{n} d\Gamma = - \int_{\mathcal{D}_c} \sigma \mathbf{grad} \mathcal{U} \cdot \mathbf{grad} \alpha V d\mathcal{D}_c \quad (5.76)$$

A test function is chosen for the scalar electric potential:

$$\mathcal{U} = \varphi' \quad \varphi' \in H_{0,b}(\operatorname{grad}, \mathcal{D}_c)$$

It is recalled that $\Gamma = \Gamma_h \cup \Gamma_b$. By its definition in $\varphi' \in H_{0,x}(\operatorname{grad}, \mathcal{D}_c)$ the potential φ' is zero on Γ_b . We thus naturally impose $\mathbf{E} \times \mathbf{n} = \mathbf{0}$ on Γ_b in the strong sense. In addition, by eliminating the calculation of the surface integral on Γ_h , we impose $\mathbf{J} \cdot \mathbf{n} = 0$ in the weak sense.

The weak form of the equation to be solved is thus:

Find $\varphi \in H_{0,b}(\operatorname{grad}, \mathcal{D}_c)$ such that $\forall \varphi' \in H_{0,b}(\operatorname{grad}, \mathcal{D}_c)$

$$\int_{\mathcal{D}_c} \sigma \mathbf{grad} \varphi' \cdot \mathbf{grad} \varphi d\mathcal{D} = - \int_{\mathcal{D}_c} \sigma \mathbf{grad} \varphi' \cdot \mathbf{grad} \alpha V d\mathcal{D}_c \quad (5.77)$$

5.5.2 Formulation \mathbf{T}

The integral form of the formulation to be solved is thus:

$$\int_{\mathcal{D}} \mathcal{U} \cdot \left[\mathbf{rot} \frac{1}{\sigma} \mathbf{rot} (\mathbf{T} + \mathbf{H}_s) \right] d\mathcal{D} = 0 \quad (5.78)$$

The weak form is obtained by integration by parts:

$$\int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathcal{U} \cdot \mathbf{rot} (\mathbf{T} + \mathbf{H}_s) d\mathcal{D} - \int_{\Gamma} \mathcal{U} \cdot (\mathbf{n} \times \mathbf{E}) d\Gamma = 0 \quad (5.79)$$

If we take:

$$\mathcal{U} = \mathbf{T}' \quad \text{avec } \mathbf{T}' \in H_{0,h}(\mathbf{rot}, \mathcal{D}) \quad (5.80)$$

Thus, with $\Gamma = \Gamma_h \cup \Gamma_b$, on Γ_h we have $\mathbf{T}' = \mathbf{0}$, hence we strongly impose $\mathbf{J} \cdot \mathbf{n}|_{\Gamma_b} = 0$. Conversely, by eliminating the surface integral on Γ_b , we ensure $\mathbf{E} \times \mathbf{n} = 0$ in the weak sense.

As a result, the problem to be solved is:

Find $\mathbf{T} \in H_{0,h}(\mathbf{rot}, \mathcal{D})$ such that $\forall \mathbf{T}' \in H_{0,h}(\mathbf{rot}, \mathcal{D})$

$$\int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{T}' \cdot \mathbf{rot} (\mathbf{T} + \mathbf{H}_s) d\mathcal{D} = 0 \quad (5.81)$$

Chapter 6

Coupling with external circuits

Summary

This chapter is devoted to methods taking account of electrical circuits external to the finite element problem. Before presenting these methods, however, certain limitations or assumptions should be mentioned. Coupling with an electrical circuit is functional in the following cases:

- Linear or non-linear materials;
- Vector magnetic potential formulation;
- Conductive domains not coupled with an external circuit;
- Taking motion into account;
- Imposition of voltages in electrical circuits.

The potential formulations \mathbf{A} and $\mathbf{A} - \varphi$ are first recalled. A circuit resolution method will be presented and these two models (magnetic and electric) will be coupled.

6.1 Breakdown of the source current

When a device is powered by n^I wound inductors, the total source current density $\mathbf{J}_s(\mathbf{X}, t)$ in broken down in the form:

$$\mathbf{J}_s(\mathbf{X}, t) = \sum_{k=1}^{n^I} \mathbf{N}_k(\mathbf{x}) i_k(t) \quad (6.1)$$

where $\mathbf{N}_k(\mathbf{x})$ (m^{-2}) is the coil density associated with inductor k , $k = 1, \dots, n^I$ and $i_k(t)$ (A) is the current flowing inside. $\mathbf{N}_k(\mathbf{x})$ can be defined by:

$$\mathbf{N}_k(\mathbf{x}) = \frac{n_k^s}{|\Sigma_k|} \mathbf{n}_k(\mathbf{x}) \quad (6.2)$$

with $|\Sigma_k|$ the surface generated by the inductor, n_k^s its number of coils and \mathbf{n}_k the normal unit vector at the cross-section of the coil.

6.2 Circuit equation

We can impose either the current flowing through the wound inductors or the voltage at their terminals. In the first case, the current is the premise of the problem. In the second, the current flowing inside becomes an unknown in the problem.

It is assumed that a voltage $v_k(t)$ is imposed on the inductor terminals k in a circuit containing a series voltage source $v_k(t)$ with resistance R_k and inductance L_k . R_k represents the resistance of the winding and possibly an external resistance, while L_k models for magnetic leaks associated with non-modelled winding overhang and/or an external inductance. Finally, the current $i_k(t)$ in this circuit is a solution of:

$$\frac{\partial \phi_k(t)}{\partial t} + L_k \frac{\partial i_k(t)}{\partial t} + R_k i_k(t) = v_k(t) \quad (6.3)$$

where ϕ_k is the magnetic flux captured by the coil k . This is the term that will be used to couple the circuit equations with the magnetoquasistatic problem.

6.2.1 Expression for the magnetic flux

The flux generated by the inductor is expressed by definition as:

$$\phi_k = n_k^s \int_{S_k} (\mathbf{B} \cdot d\mathbf{S}_k) \quad (6.4)$$

where \mathbf{S}_k is the surface generated by the contour of the coil k as shown in Figure 6.1.

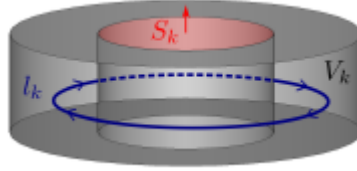


Figure 6.1: Wound inductor

Applying Stokes' theorem and using $\mathbf{B} = \text{rot } \mathbf{A}$, we have:

$$\phi_k = n_k^s \oint_{l_k} (\mathbf{A} \cdot d\mathbf{l}_k) = n_k^s \oint_{l_k} (\mathbf{A} \cdot \mathbf{N}_k) dV_k \quad (6.5)$$

where l_k is the closed contour bounding the surface S_k , again shown in Figure 6.1. Using the definition of \mathbf{N}_k (see equation 6.2), we finally find:

$$\phi_k = \int_{V_k} (\mathbf{A} \cdot \mathbf{N}_k) dV_k \quad (6.6)$$

where: $V_k = \oint_{l_k} |\Sigma_k| d\mathbf{l}_k$ is the inductor volume.

6.2.2 Formulation of the electrical problem

This section deals with linear circuits consisting of passive dipoles (R, L and C) and voltage sources. Current sources are not taken into account, as the current can be directly applied to the wound inductors in the electromagnetic problem.

6.2.3 Mesh current method

To couple the electromagnetic problem with the circuit problem, the mesh current method has been chosen. Initially, the finite element model is not involved.

Consider the circuit shown in Figure 6.2 which represents a star coupling of three phases feeding loads R, L and C.

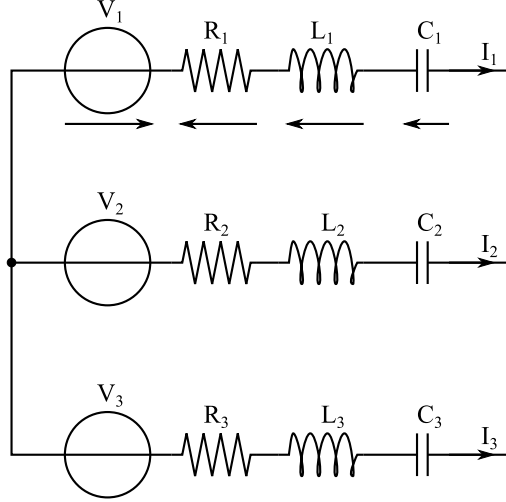


Figure 6.2: RLC circuit

This circuit has:

- 12 dipoles (which can also be called branches or edges);
- 11 nodes (in the sense of electrical circuits and graphs).

There are thus “ b_{cir} ” independent loops such that:

$$n_{cir} - a_{cir} + b_{cir} = 1 \quad (6.7)$$

With:

- n_{cir} : the number of nodes in the circuit;
- a_{cir} : the number of branches in the circuit;
- b_{cir} : the number of independent loops in the circuit.

The number of loops to be considered in the example is thus 2. Initially, they are chosen arbitrarily.

For each loop, Kirchhoff's voltage law can be written as follows:

$$K M U = 0 \quad (6.8)$$

With:

$$U = U_S + U_R + U_L + U_C \quad (6.9)$$

Where:

- U_S is the source voltage vector;

- U_R is the resistive dipole voltage vector;
- U_L is the inductive dipole voltage vector;
- U_C is the capacitive dipole voltage vector;

and KM is the branch–mesh incidence matrix (or loop of size $b_{cir} \times a_{cir}$ such that:

- $KM(b_{cir}, a_{cir}) = 1$, if the loop orientation b_{cir} is in the same direction as the dipole voltage a_{cir} ;
- $KM(b_{cir}, a_{cir}) = -1$, if the loop orientation b_{cir} is in the opposite direction to the dipole tension a_{cir} ;
- $KM(b_{cir}, a_{cir}) = 0$, otherwise.

In the case shown in Figure 6.2, the matrix KM is:

$$KM = \begin{bmatrix} 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 \end{bmatrix} \quad (6.10)$$

The current I through each dipole can be written according to the previously determined fictitious currents J_{cir} flowing in the loops:

$$I_{(n)} = KM^T J_{cir(n)} \quad (6.11)$$

The following notation is adopted:

- $I_{(n)}$ is the current at the time iteration t ;
- $I_{(n-1)}$ is the current at the time iteration $t - \Delta t$;

With these conventions, equation 6.9 becomes:

$$U_{(n)} = U_{S(n)} + RI_{(n)} + L \frac{I_{(n)} - I_{(n-1)}}{\Delta t} + \frac{\Delta t}{C} I_{(n)} + U_{C(n-1)} \quad (6.12)$$

where:

$$I_{C(n)} = C \frac{U_{C(n)} - U_{C(n-1)}}{\Delta t} \quad (6.13)$$

We then resolve the following system:

$$KM \left[\mathbf{R} + \frac{\mathbf{L}}{\Delta t} + \frac{\Delta t}{\mathbf{C}} \right] KM^T J_{cir(n)} = -KM (U_S + U_{C(n-1)}) + KM \frac{L}{\Delta t} KM^T J_{cir(n-1)} \quad (6.14)$$

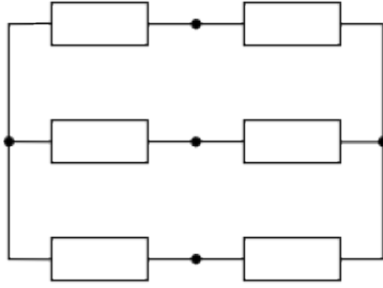
The matrices \mathbf{R} , \mathbf{L} and \mathbf{C} are squares of size equal to the number of dipoles. They contain the value of the corresponding dipole on their diagonal, according to the numbering of the branches of the circuit. An example is shown below:

$$\mathbf{R} + \mathbf{L} + \mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & R_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & L_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & C_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & R_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & L_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & C_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & R_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & L_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & C_3 \end{bmatrix}$$

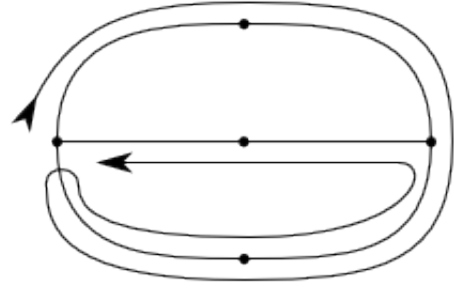
6.2.4 Method for calculating the tree of the electrical circuit

Closing currents are determined in two steps. Figures 6.3 and 6.4 illustrate the methods used.

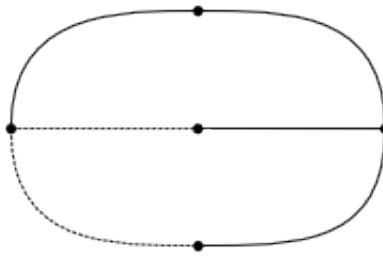
1. Calculating a tree
2. Calculating closing currents



(a) Sample circuit



(b) Graph of the circuit and tree search path



(c) Tree (solid line) and co-tree (dotted line)

Figure 6.3: Calculating a circuit tree

From the graph of the circuit (see Figure 6.3a), the various branches are crossed to construct a spanning tree. This spanning tree must link all nodes of the graph without forming a loop.

By crossing the branches one by one, it is possible to form a loop without having passed through all the branches. In this case, the method is to go back one branch and look for another possible path (see Figure 6.3b).

This process continues until a spanning tree is obtained (see Figure 6.3c).

Together with this tree, a co-tree is also calculated. In our example, the co-tree is made up of two branches. This number of branches is equal to the number of independent loops (see equation 6.7).

The calculation of mesh currents is carried out by crossing the spanning tree, starting with the branches of the co-tree. Figure 6.4 shows two pathways for mesh currents.

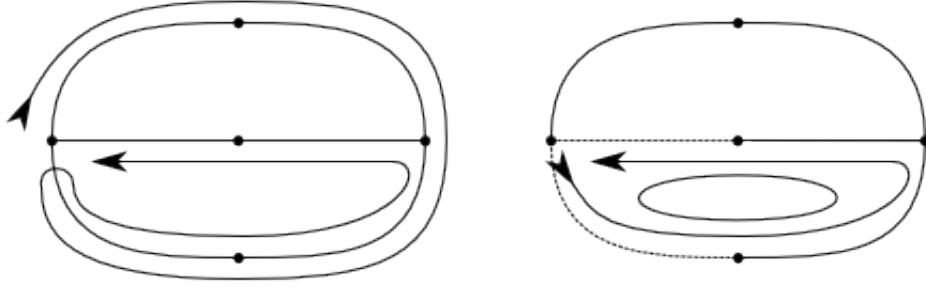


Figure 6.4: Determining independent current loops

6.3 Coupling solid conductors in code_Carmel spectral version

In a solid conductor, the voltage between two terminals or surfaces S_1 and S_2 , assumed to be electrical equipotentials, is the difference between the potential levels of S_1 and S_2 , i.e. $U = \phi_{S_2} - \phi_{S_1}$.

It will be recalled that with the vector magnetic potential formulation, the current density is expressed as:

$$\mathbf{J} = -\sigma \left(\mathbf{grad} \varphi + \frac{\partial \mathbf{A}}{\partial t} \right) \quad (6.15)$$

The weak expression of current I can be written:

$$I = \int \mathbf{J} \cdot \mathbf{grad} \hat{w}^0 dS = - \int \sigma \left(\mathbf{grad} \varphi + \frac{\partial \mathbf{A}}{\partial t} \right) \cdot \mathbf{grad} \hat{w}^0 dS \quad (6.16)$$

where: \hat{w}^0 is a nodal shape function restricted to cross-section S of the conductor.

When the solid conductor is connected to an external circuit, the voltage U at the terminals of the conductor and the current flowing through it are unknown and imposed by the external circuit. The coupling between the external circuit and the solid conductor will be performed using these two overall values, U and I.

We start by writing the two systems to be coupled.

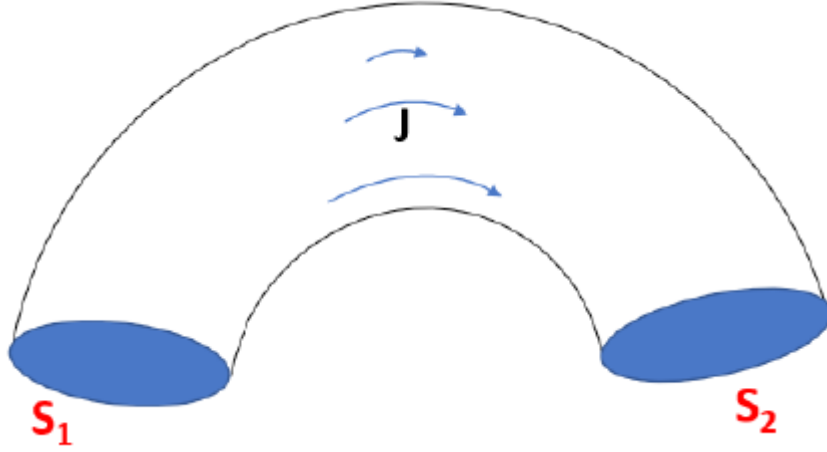


Figure 6.5: Model of a bar made up of two pieces, electrically insulated with an imperfect insulator

If the voltage at the conductor terminal is imposed, then the matrix system to be resolved is written:

$$\begin{pmatrix} RotRot + WW & WGrad^T & 0 \\ WGrad & GradGrad & N^T \\ 0 & N & 0 \end{pmatrix} \begin{pmatrix} A^{inc} \\ \varphi^{inc} \\ I \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ U \end{pmatrix} \quad (6.17)$$

With N a vector of size equal to the number of scalar unknowns ϕ , zero everywhere except at the index of the unknown ϕ_{S_1} where it is $-\alpha$ and the index of ϕ_{S_2} where it is $+\alpha$.

The matrix system of the external circuit is constructed using circuit analysis methods. It is known that in electrical formulation $\mathbf{A} - \varphi$, closing current analysis is appropriate for the coupling of wound inductors (conductors without eddy currents) while nodal voltage circuit analysis is the most natural for solid inductors. Closing current analysis is already implemented in the spectral version of code_Carmel. For reasons of ease of maintenance and implementation of the spectral version, nodal voltage circuit analysis is not being developed.

However, closing current circuit analysis as developed poses problems when considering solid conductors. To overcome this difficulty, we have modified the closing current analysis by incorporating fictitious voltage sources corresponding to each solid conductor. Hence, the matrix system of the external electrical circuit is written:

$$\begin{pmatrix} Y_{11} & \cdots & Y_{1m} \\ \vdots & \ddots & \vdots \\ Y_{m1} & \cdots & L_{mm} \end{pmatrix} I = V_s + N \varphi^{inc} \quad (6.18)$$

With:

- m the number of passive components (resistors, chokes, capacitors) in the external electrical circuit;
- Y is identical to an impedance;
- I is the closing current vector;
- V_s is the voltage source vector imposed in the external circuit.

Part II

Overview of space and time discretisation

Chapter 7

Discretisation spaces

Summary

The formulations developed in the preceding chapters cannot be solved analytically due to the complex geometries of the devices. We must use numerical methods to solve these equations. Hence, the problem must be discretised. The local electromagnetic values, which are actually the unknowns of the problem, are defined in series of function spaces. Hence, we must discretise the sequences of function spaces as well as the differential operators. To do this, we will define a discrete structure similar to that of the continuous domain presented in the preceding chapters.

This is done using the Finite Element Method (FEM), which generates a double discretisation. The first consists in breaking down the domain under study (space discretisation) into small elements of simple shape (tetrahedra, prisms, hexahedra or pyramids). The second discretisation is of the unknowns.

7.1 Interpolation spaces

7.1.1 Overview

In FEM, the continuous domain \mathcal{D} is partitioned into sub-domains of simple shapes in which Maxwell's equations are approached numerically. This means a finite element mesh consisting of n_0 nodes, n_1 edges, n_2 facets, and n_3 volume elements.

There is a relationship (the Euler-Poincaré formula) between these numbers:

$$n_0 - n_1 + n_2 - n_3 = \xi \quad (7.1)$$

where ξ is the Betti number which is equal to 1 plus the number of loops minus the number of holes in the meshed domain.

7.1.2 Shape functions

7.1.2.1 Nodal function

The mesh vertices, at each node n , are associated with continuous scalar nodal functions w_n^0 . Their expression depends on the type of element used. With nodal functions, we can verify the relationship at each point of the domain:

$$\sum_{n \in \mathcal{N}_h} w_n^0 = 1 \quad (7.2)$$

All mesh functions w_n^0 generate a space of finite dimension denoted \mathcal{W}^0 . If a function U belongs to \mathcal{W}^0 , this gives:

$$U = \sum_{n \in \mathcal{N}_h} w_n^0 u_n \quad (7.3)$$

u_n represents the value at node n of function U . Denoting \mathbf{U}_n the vector $(u_n)_{n \in \mathcal{N}_h}$ and \mathbf{W}_n the vector containing the interpolation functions at the nodes, equation 7.3 is expressed as follows:

$$U = \mathbf{W}_n \mathbf{U}_n \quad (7.4)$$

The properties of the interpolation functions w_n^0 require that the function U is continuous across domain \mathcal{D} .

7.1.2.2 Edge function

As shown in Figure 7.1, let there be an edge A_{nm} formed by nodes N_n and N_m , to which we associate the edge function \mathbf{w}_a^1 .

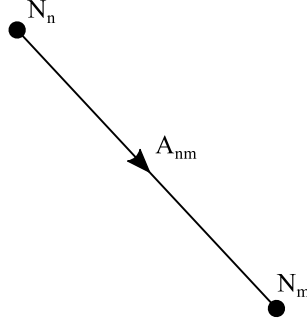


Figure 7.1: Definition of the edge A_{nm}

In the case of tetrahedra, we have [Bossavit 1993]:

$$\mathbf{w}_a^1 = w_n^0 \mathbf{grad} w_m^0 - w_m^0 \mathbf{grad} w_n^0 \quad (7.5)$$

where w_n^0 and w_m^0 are the nodal functions associated with nodes N_n and N_m .

The circulation of w_a^1 is equal to 1 along edge A_{nm} and is zero on the other edges.

All of these functions w_a^1 generate the space of edge elements of finite dimension \mathcal{W}^1 .

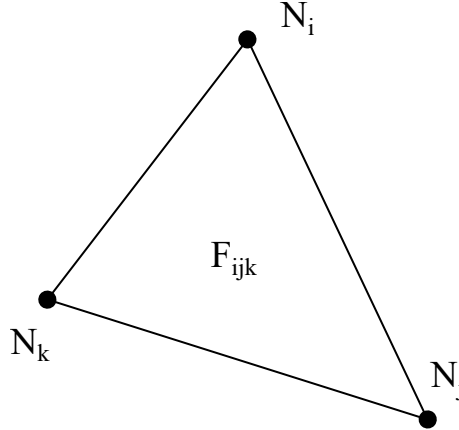
Let there be a vector \mathbf{U} belonging to \mathcal{W}^1 , thus giving

$$\mathbf{U} = \sum_{a \in \mathcal{A}_h} \mathbf{w}_a^1 u_a \quad (7.6)$$

u_a represents the circulation of \mathbf{U} along edge 'a' defined by:

$$u_a = \int_a \mathbf{U} \cdot d\mathbf{l} \quad (7.7)$$

Finally, at the interface between two elements, the tangential component of the value discretised by the edge elements is preserved.

Figure 7.2: Definition of a triangular facet F_{ijk}

7.1.2.3 Facet function

Depending on the type of mesh element, a facet can be triangular or quadrangular. By way of example, a triangular facet is shown in Figure 7.2 .

The function \mathbf{w}_f associated with a triangular facet F_{ijk} is written [Bossavit 1993]:

$$\mathbf{w}_f^2 = 2 \left(w_k^0 \mathbf{grad} w_i^0 \times \mathbf{grad} w_j^0 + w_j^0 \mathbf{grad} w_i^0 \times \mathbf{grad} w_k^0 + w_i^0 \mathbf{grad} w_j^0 \times \mathbf{grad} w_k^0 \right) \quad (7.8)$$

where w_i^0 , w_j^0 and w_k^0 are, respectively, the interpolation functions at nodes N_i , N_j and N_k .

We denote \mathcal{W}^2 the space of the facet elements generated by the functions w_f^2 .

By definition, the flux of the function w_f^2 is equal to 1 through facet 'f' and zero on the other mesh facets.

Taking a function \mathbf{U} belonging to \mathcal{W}^2 , it is expressed as follows:

$$\mathbf{U} = \sum_{f \in \mathcal{F}_h} \mathbf{w}_f^2 u_f \quad (7.9)$$

where u_f represents the flux of vector \mathbf{U} through facet 'f', thus:

$$u_f = \int_f \mathbf{U} \cdot \mathbf{n} ds \quad (7.10)$$

Since the normal component of the functions w_f^2 is continuous across each facet, hence the normal component of a function belonging to \mathcal{W}^2 is also continuous.

7.1.2.4 Volume function

Finally, on each volume element v , we introduce the scalar function w_v^3 equal to the inverse of the volume of the element on it, and zero on the other elements.

$$\begin{aligned} w_v^3(x) &= \frac{1}{\text{vol}(v)} & \text{si } x \in v \\ w_v^3(x) &= 0 & \text{si } x \notin v \end{aligned} \quad (7.11)$$

where x is a point of \mathcal{D} and $\text{vol}(v)$ is the volume of the element considered.

The space generated by the functions w_v^3 is denoted \mathcal{W}^3 . A function U belongs to \mathcal{W}^3 if:

$$U = \sum_{v \in \mathcal{D}_h} w_v^3 u_v \quad (7.12)$$

In this expression, u_v represents the volume integral of function U on element v .

7.1.3 Discrete spaces

As in the case of continuous domains, boundary condition restrictions can be introduced in \mathcal{W}^i spaces.

On Γ_h we have:

$$\mathcal{W}_h^0 = \{u \in \mathcal{W}^0, u|_{\Gamma_h} = 0\} \quad (7.13)$$

$$\mathcal{W}_h^1 = \{\mathbf{u} \in \mathcal{W}^1, \mathbf{u} \times \mathbf{n}|_{\Gamma_h} = 0\} \quad (7.14)$$

$$\mathcal{W}_h^2 = \{\mathbf{u} \in \mathcal{W}^2, \mathbf{u} \cdot \mathbf{n}|_{\Gamma_h} = 0\} \quad (7.15)$$

and, on Γ_b :

$$\mathcal{W}_b^0 = \{u \in \mathcal{W}^0, u|_{\Gamma_b} = 0\} \quad (7.16)$$

$$\mathcal{W}_b^1 = \{\mathbf{u} \in \mathcal{W}^1, \mathbf{u} \times \mathbf{n}|_{\Gamma_b} = 0\} \quad (7.17)$$

$$\mathcal{W}_b^2 = \{\mathbf{u} \in \mathcal{W}^2, \mathbf{u} \cdot \mathbf{n}|_{\Gamma_b} = 0\} \quad (7.18)$$

7.1.4 Potentials

Hence, there is a similar structure to that established in the continuous domain. We can thus naturally define the interpolation spaces of the fields and potentials introduced above.

We thus have:

- the scalar potentials: $\Omega \in \mathcal{W}_h^0; \varphi \in \mathcal{W}_b^0$.
- the fields \mathbf{E} and \mathbf{H} and the vector potentials: $\mathbf{H} \in \mathcal{W}_h^1; \mathbf{E} \in \mathcal{W}_b^1; \mathbf{A} \in \mathcal{W}_b^1; \mathbf{T} \in \mathcal{W}_h^1$.
- the current density \mathbf{J} and the magnetic induction \mathbf{B} : $\mathbf{J} \in \mathcal{W}_h^2; \mathbf{B} \in \mathcal{W}_b^2$.

7.2 Discrete differential operators

As in the case of continuous function spaces, discrete operators can be used to establish a link between nodes, edges, facets and mesh elements (primal and dual).

In fact, these are the incidence matrices introduced by A. Bossavit. To illustrate these matrices, we will deal with the case of the hexahedron presented in Figure 7.3. It should be noted that the orientations are chosen arbitrarily on the primal mesh, while the orientation of the dual mesh is deduced from the orientation of the primal mesh.

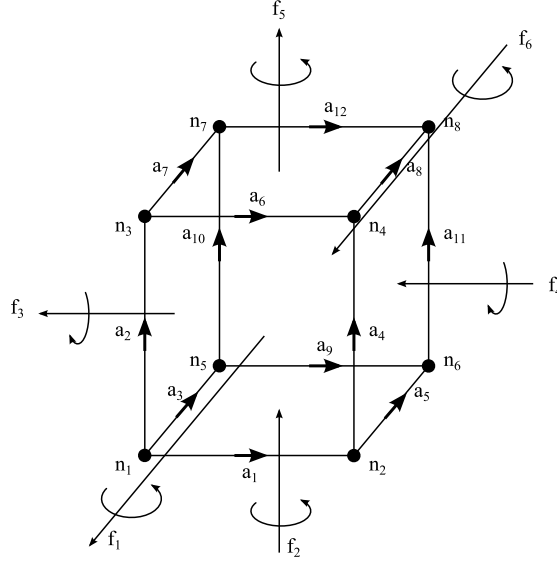


Figure 7.3: Oriented hexahedron

7.2.1 The discrete gradient G_{an} .

The discrete form of the gradient is the edge–node incidence matrix that connects all primal nodes and all primal edges of the mesh. Its size corresponds to the number of edges and nodes in the mesh. In addition, the incidence $i(a, n)$ of a node on an edge can take only three values:

- 0 if the node does not belong to the edge;
- 1 if the node is the start node of the edge;
- -1 if the node is the end node of the edge.

Table 7.1 shows the incidence matrix for the hexahedron in Figure 7.3.

G_{an}	n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8
a_1	-1	1						
a_2	-1		1					
a_3	-1				1			
a_4		-1		1				
a_5		-1				1		
a_6			-1	1				
a_7			-1					1
a_8				-1		1		
a_9					-1		1	
a_{10}					-1			
a_{11}						-1		1
a_{12}							-1	1

Table 7.1: Incidence matrix G_{an} of the hexahedron in Figure 7.3

7.2.2 The discrete curl R_{fa}

This is the facet–edge incidence matrix that connects edges to facets of the mesh. As with G_{an} the terms of R_{fa} can be 0, 1 or -1. Each facet is associated with a normal which is either incoming

or outgoing and a direction of rotation (see Figure 7.3). In addition, the incidence R_{fa} of a facet on an edge can take only three values:

- 0 if the edge does not belong to the facet;
- 1 if the edge is orientated in the same direction as the direction of rotation associated with the facet;
- -1 if the edge is orientated in the opposite direction to the direction of rotation associated with the facet.

R_{fa}	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}
f_1	1	-1		1		-1						
f_2	1		-1		1				-1			
f_3		1	-1				1			-1		
f_4				1	-1			1			-1	
f_5						1	-1	1				-1
f_6									-1	-1	1	-1

Table 7.2: Incidence matrix R_{fa} pour un hexaèdre

7.2.3 The discrete divergence D_{vf}

This operator associates all elements with all mesh facets. The divergence is defined by the volume–facet incidence matrix. Again, its terms are 0, 1 or -1. In addition, the incidence D_{vf} of a facet on a volume can take only three values:

- 0 if the facet does not belong to the volume;
- 1 if the normal of the facet is orientated outwards from the volume;
- -1 if the normal of the facet is orientated inwards into the volume.

D_{vf}	f_1	f_2	f_3	f_4	f_5	f_6
V_1	-1	1	-1	1	-1	1

Table 7.3: Incidence matrix D_{vf} pour un hexaèdre

7.2.4 The dual mesh concept

As mentioned in the previous paragraph, it is necessary to discretise the space to solve the problem numerically. In 3D, a discretised function space \mathcal{V}_h (mesh) is made up of (volume) elements, facets, edges and nodes. The sets of volumes, facets, edges and nodes are respectively called $\mathcal{V}_v, \mathcal{V}_f, \mathcal{V}_a, \mathcal{V}_n$. The union of all these sets forms the discrete space: $\mathcal{V}_h = \mathcal{V}_v \cup \mathcal{V}_f \cup \mathcal{V}_a \cup \mathcal{V}_n$. This mesh can be combined with a dual mesh (grid) $\tilde{\mathcal{V}}_h$ which is also made up of volumes, facets, edges and nodes, also referred to as dual. By analogy, it can be established that: $\tilde{\mathcal{V}}_h = \tilde{\mathcal{V}}_v \cup \tilde{\mathcal{V}}_f \cup \tilde{\mathcal{V}}_a \cup \tilde{\mathcal{V}}_n$. The dual mesh is built from the initial mesh, referred to as the ‘primal’ mesh, by associating:

- each primal node n with a dual volume \tilde{v}
- each primal edge a with a dual facet \tilde{f}
- each primal facet f with a dual edge \tilde{a}
- each primal volume v with a dual node \tilde{n}

In Figure 7.4 a 2D primal mesh and its dual are represented.

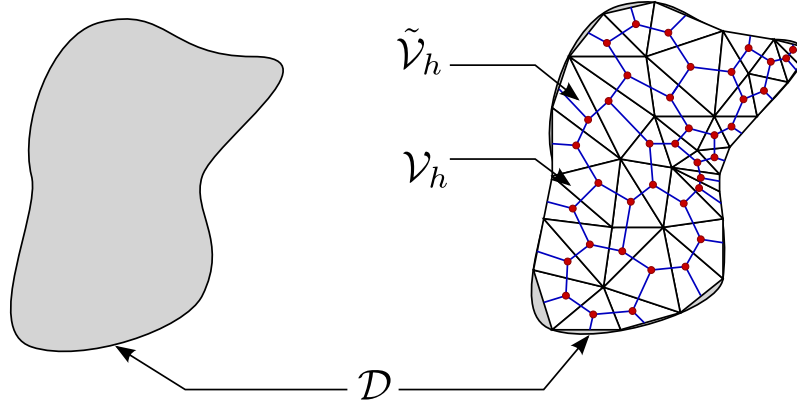


Figure 7.4: Discretisation of a continuous domain and its dual mesh

7.2.5 Properties of the operators

As seen with the primal mesh, we can similarly define the operators \tilde{G}_{an} , \tilde{R}_{fa} and \tilde{D}_{vf} of the dual mesh. If the orientation of the edges, facets and elements of the dual mesh is deduced from the orientation of the edges, facets and elements of the primal mesh, we can demonstrate the following relationships between the discrete operators [Bossavit 2003]:

$$G_{an} = -\tilde{D}_{vf}^t$$

$$\tilde{G}_{an} = -D_{vf}^t$$

$$R_{fa} = \tilde{R}_{fa}^t$$

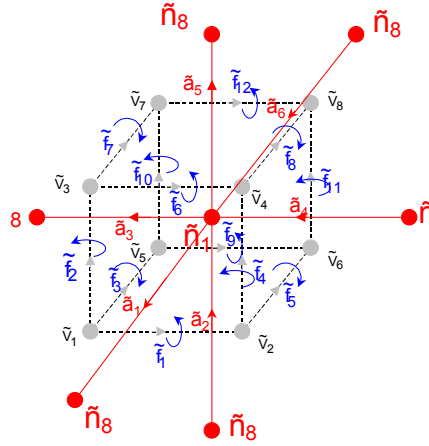


Figure 7.5: Dual mesh of the hexahedron in Figure 2.3

Using these properties, it can be noted that if the operators on the primal grid are known, it is easy to deduce the operators on the dual grid and vice versa.

7.3 Properties of interpolation spaces

Let there be a scalar function U belonging to $\mathcal{W}^0(\mathcal{D})$, thus giving:

$$U = \sum_1^{n=n0} w_n^0 u_n \quad (7.19)$$

The gradient of function U is written:

$$\mathbf{grad} U = \sum_1^{n=n0} \mathbf{grad} w_n^0 u_n \quad (7.20)$$

however, as was shown in the references [Bossavit 1993] and [Dular 1994], we have:

$$\mathbf{grad} w_n^0 = \sum_{a \in \varepsilon_h} G_{an} \mathbf{w}_a^1 \quad (7.21)$$

From the preceding equations, we obtain:

$$\mathbf{grad} U = \sum_{n \in \mathcal{N}_h} \left(\sum_{a \in \varepsilon_h} G_{an} \mathbf{w}_a^1 \right) u_n \quad (7.22)$$

and further:

$$\mathbf{grad} U = \sum_{a \in \varepsilon_h} \left(\sum_{n \in \mathcal{N}_h} G_{an} u_n \right) \mathbf{w}_a^1 \quad (7.23)$$

Hence, the gradient of a function of \mathcal{W}_0 is included in \mathcal{W}_1 . Hence:

$$Im(\mathbf{grad} \mathcal{W}^0) \subset \mathcal{W}^1 \quad (7.24)$$

Thus we have:

$$Im(\mathbf{grad} \mathcal{W}^0) \subset Ker(\mathbf{rot} \mathcal{W}^1) \quad (7.25)$$

In the case of a simply connected domain, we find the equation previously defined in the continuous domain:

$$Im(\mathbf{grad} \mathcal{W}^0) = Ker(\mathbf{rot} \mathcal{W}^1) \quad (7.26)$$

We show [Bossavit 1993][Dular 1994], through the same approach as before, that if \mathbf{U} is a function belonging to \mathcal{W}^1 , then:

$$\mathbf{U} = \sum_{a \in \varepsilon_h} \mathbf{w}_a^1 u_a \quad (7.27)$$

Under these conditions, the function $\mathbf{rot} \mathbf{U}$ is written:

$$\mathbf{rot} \mathbf{U} = \sum_{f \in \mathcal{F}_h} \left(\sum_{a \in \varepsilon_h} R_{fa} \mathbf{u}_a \right) \mathbf{w}_f^2 \quad (7.28)$$

Thus we have:

$$\mathbf{rot} \mathbf{U} \in \mathcal{W}^2 \quad (7.29)$$

In addition, if domain \mathcal{D} is simply connected with a connected surface Γ , the following equation applies:

$$Im(\mathbf{rot} \mathcal{W}^1) = Ker(div \mathcal{W}^2) \quad (7.30)$$

If \mathbf{U} is a function of \mathcal{W}^2 then:

$$\mathbf{U} = \sum_{f \in \mathcal{F}_h} \mathbf{w}_f^2 u_f \quad (7.31)$$

Under these conditions, calculating $\text{div } \mathbf{U}$ gives:

$$\text{div } \mathbf{U} = \sum_{v \in \mathcal{D}_h} (D_{vf} u_f) \mathbf{w}_v^3 \quad (7.32)$$

we thus have:

$$\text{div } \mathbf{U} \in \mathcal{W}^3 \quad (7.33)$$

and further:

$$\text{Im}(\text{div } \mathcal{W}^2) = \mathcal{W}^3 \quad (7.34)$$

The properties set out above can be put in the form of a sequence of discrete spaces as shown in Figure 7.6:

$$\mathcal{W}^0 \xrightarrow{\text{grad}} \mathcal{W}^1 \xrightarrow{\text{rot}} \mathcal{W}^2 \xrightarrow{\text{div}} \mathcal{W}^3$$

Figure 7.6: Sequence of discrete spaces \mathcal{W}^i

7.4 Discretisation of fields and potentials

The physical values are thus linear combinations of space functions. Hence, we can write:

- for the scalar magnetic potential Ω defined in \mathcal{W}_h^0 :

$$\Omega(\mathbf{x}, t) = \sum_i \Omega_i(t) w_i^0(\mathbf{x}) \quad (7.35)$$

- for scalar electric potential φ expressed in \mathcal{W}_b^0 :

$$\varphi(\mathbf{x}, t) = \sum_i \varphi_i(t) w_i^0(\mathbf{x}) \quad (7.36)$$

- for the vector magnetic potential \mathbf{A} in \mathcal{W}_b^1 :

$$\mathbf{A}(\mathbf{x}, t) = \sum_i A_i(t) \mathbf{w}_i^1(\mathbf{x}) \quad (7.37)$$

- for the vector electric potential \mathbf{T} in \mathcal{W}_h^1 :

$$\mathbf{T}(\mathbf{x}, t) = \sum_i T_i(t) \mathbf{w}_i^1(\mathbf{x}) \quad (7.38)$$

- for the time derivative of \mathbf{A} which is in \mathcal{W}_b^1 , only for the spectral version:

$$\frac{\partial \mathbf{A}(\mathbf{x}, t)}{\partial t} = \sum_i A_i^\partial(t) \mathbf{w}_i^1(\mathbf{x}) \quad (7.39)$$

- for the magnetic field \mathbf{H} expressed in \mathcal{W}_h^1 :

$$\mathbf{H}(\mathbf{x}, t) = \sum_l H_l(t) \mathbf{w}_l^1(\mathbf{x}) \quad (7.40)$$

- for the electric field \mathbf{E} defined in \mathcal{W}_b^1 :

$$\mathbf{E}(\mathbf{x}, t) = \sum_l E_l(t) \mathbf{w}_l^1(\mathbf{x}) \quad (7.41)$$

- for the magnetic induction \mathbf{B} in \mathcal{W}_b^2 :

$$\mathbf{B}(\mathbf{x}, t) = \sum_l B_l(t) \mathbf{w}_l^2(\mathbf{x}) \quad (7.42)$$

- for the electric current density \mathbf{J} in \mathcal{W}_h^2 :

$$\mathbf{J}(\mathbf{x}, t) = \sum_l J_l(t) \mathbf{w}_l^2(\mathbf{x}) \quad (7.43)$$

- for the magnetic induction \mathbf{B}_r in \mathcal{W}_b^2 :

$$\mathbf{B}^r(\mathbf{x}, t) = \sum_l B_l^r(t) \mathbf{w}_l^2(\mathbf{x}) \quad (7.44)$$

- for the electric current density \mathbf{J}^Γ in \mathcal{W}_h^2 :

$$\mathbf{J}^\Gamma(\mathbf{x}, t) = \sum_l J_l^\Gamma(t) \mathbf{w}_l^2(\mathbf{x}) \quad (7.45)$$

- for the magnetic field \mathbf{H}^Γ expressed in \mathcal{W}_h^1 :

$$\mathbf{H}^\Gamma(\mathbf{x}, t) = \sum_l H_l^\Gamma(t) \mathbf{w}_l^1(\mathbf{x}) \quad (7.46)$$

Chapter 8

Discretisation of source terms and global quantities

Summary

The purpose of this chapter is to present the specific features of `code_Carmel` on tree techniques and the determination of source values in general. `code_Carmel` uses an original method to introduce a gauge. In the potential equations seen above, it uses a tree technique. This approach is detailed here.

More generally, this chapter explains how overall values are discretised from vectors \mathbf{K} and \mathbf{N} .

8.1 Introduction of a gauge (edge and facet trees)

8.1.1 Value of trees

As discussed above, to obtain the uniqueness of a vector field, it is necessary to impose a gauge condition. In the case of curl, as shown in equation 7.26, a gradient must be set. For divergence, a condition on the curl must be imposed. In what follows, the conditions to be imposed in the discrete domain will be detailed.

To ensure the uniqueness of a vector \mathbf{U} belonging to \mathcal{W}^1 such that:

$$\mathbf{rot} \mathbf{U} = \mathbf{V},$$

it suffices to fix the circulation of \mathbf{U} on the edges of a tree. A tree consists of a set of edges that connect all the mesh nodes without forming loops (the gauge $\mathbf{U} \cdot \mathbf{w} = f(\mathbf{r})$ introduced in the continuous domain is found here at the discrete level).

Let there be two vectors \mathbf{U}_1 and \mathbf{U}_2 belonging to \mathcal{W}^1 such that $\mathbf{rot} \mathbf{U}_i = \mathbf{V}$ and such that circulation of \mathbf{U}_1 and \mathbf{U}_2 on the edges of the tree are fixed. We denote:

$$\Delta \mathbf{U} = \mathbf{U}_1 - \mathbf{U}_2$$

We thus have:

$$\mathbf{rot} \Delta \mathbf{U} = 0 \quad \text{avec } \Delta \mathbf{U} \in \mathcal{W}^1 \tag{8.1}$$

The domain being simply connected, there is a scalar λ with:

$$\lambda \in \mathcal{W}^0$$

such that:

$$\Delta \mathbf{U} = \mathbf{grad} \lambda \quad (8.2)$$

In addition, the circulation of $\Delta \mathbf{U}$ on the edges of the tree is equal to zero. If n_1 and n_2 are two mesh nodes, we have:

$$\lambda_{n_1} - \lambda_{n_2} = \int_{\mathbf{x}_{n_1}}^{\mathbf{x}_{n_2}} \mathbf{grad} \lambda \cdot d\mathbf{l} = \int_{\mathbf{x}_{n_1}}^{\mathbf{x}_{n_2}} \Delta \mathbf{U} \cdot d\mathbf{l} \quad (8.3)$$

where λ_{n_1} and λ_{n_2} are the nodal values of λ in n_1 and n_2 , while x_{n_1} and x_{n_2} are the coordinates of these nodes. To reach n_2 from n_1 , any path can be taken along the edges of the mesh. In this case, the chosen path can be on an edge tree where the circulation of $\Delta \mathbf{U}$ is zero. Under these conditions:

$$\lambda_{n_1} - \lambda_{n_2} = 0 \quad (8.4)$$

which requires:

$$\mathbf{grad} \lambda = 0$$

Thus we have:

$$\mathbf{U}_1 = \mathbf{U}_2$$

Let us now consider \mathbf{U}_1 and \mathbf{U}_2 belonging to \mathcal{W}^2 such that:

$$\text{div} \mathbf{U}_i = V$$

with: $V \in \mathcal{W}^3$, we then have:

$$\mathbf{U}_1 = \mathbf{U}_2 + \mathbf{rot} \Lambda \quad (8.5)$$

with: $\Lambda \in \mathcal{W}^1$

We take:

$$\mathbf{U}_1 - \mathbf{U}_2 = \Delta \mathbf{U}$$

and:

$$\text{div} (\Delta \mathbf{U}) = 0$$

By analogy with the previous case, a facet tree must be constructed on which the flux values of $\Delta \mathbf{U}$ are fixed.

Edge tree construction techniques are widely covered in the literature [[Albanese, Rubinacci 2000](#)], [[Golias, Tsiboukis 1994](#)], [[Gondran, Minoux 1995](#)]. For this reason, in what follows, we only detail the method we have developed for construction of a facet tree.

However, by way of example, [Figure 8.2](#) shows an edge tree relating to the mesh in [Figure 8.1](#).

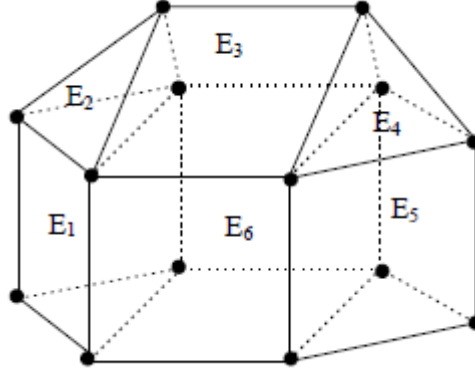


Figure 8.1: Example of a mesh

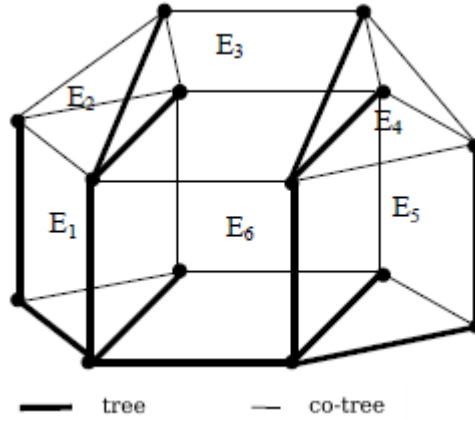


Figure 8.2: Example of an edge tree

8.1.2 Construction of a facet tree

To develop a facet tree, we will rely on the algorithm for construction of edge trees [Albanese, Rubinacci 2000]. This algorithm is based on graph properties. In the construction of an edge tree, because an edge connects two nodes, it is possible to obtain an edge-node graph of a mesh. However, there is an analogous relationship between facets and elements, as one facet connects two elements [Le Menach et al 1998].

First, we define a new element E_Γ that symbolises the exterior of the domain. It is noted that all facets belonging to the exterior boundary Γ are part of E_Γ . By way of example, Figure 8.3 shows the transposition of two elements and one facet into one edge and two nodes.

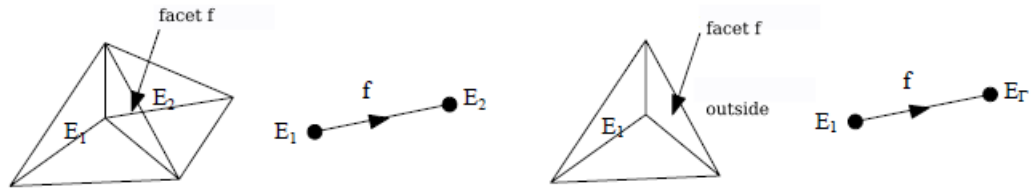


Figure 8.3: Facet–element link

In this figure, two cases are considered: a link between two elements (on the left of Figure 8.3) and a link between an element and the exterior boundary (on the right of Figure 8.3).

Again taking the mesh in Figure 8.1 and numbering the facets as shown in Figure 8.4:

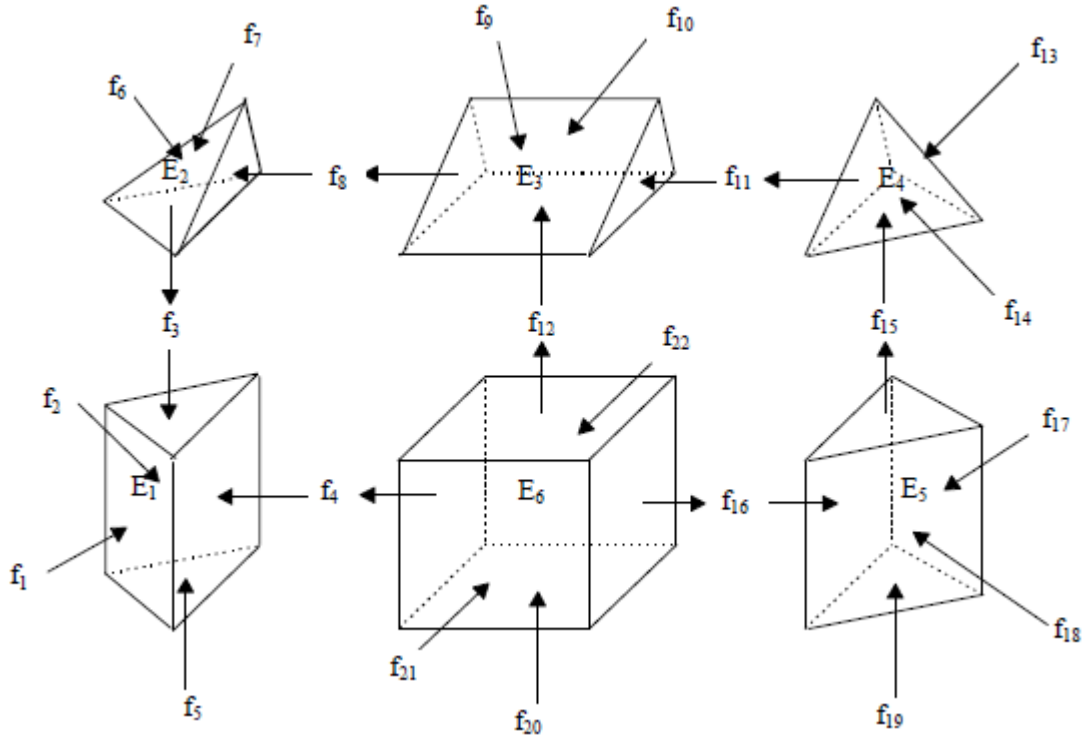


Figure 8.4: Definition of mesh facets

We can now plot the facet–element graph of this mesh (see Figure 8.5).

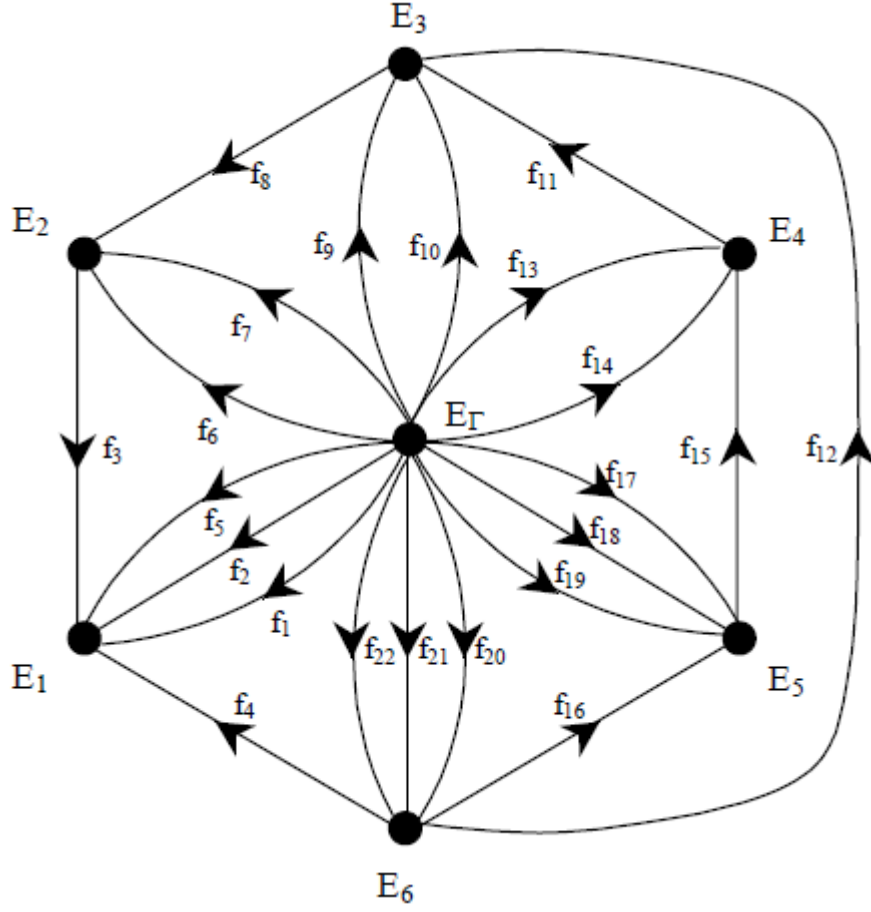


Figure 8.5: Graph of the facet–element link

The boundary element E_Γ is placed at the centre of the graph, where all the edges representing facets belonging to Γ converge. The orientation of the facets corresponds to the direction of the normals of the facets defined in Figure 8.4.

Consider a vector \mathbf{U} belonging to \mathcal{W}^2 such that:

$$\operatorname{div} \mathbf{U} = V \quad \text{avec } V \in \mathcal{W}^3.$$

Vector \mathbf{U} and function V can be expressed by the equations:

$$\mathbf{U} = \sum_{f \in \mathcal{F}} u_f \mathbf{w}_f \quad (8.6)$$

$$V = \sum_{v \in \mathcal{V}} v_v w_v \quad (8.7)$$

where v_v represents the integral of function V on the element considered. Because of the relationship between \mathbf{U} and V , each element 'e' has the following property:

$$\sum_{f \in \mathcal{F}} i(v, f) u_f = v_v \quad (8.8)$$

Under these conditions, constructing a facet tree comes down to searching for facets with flux values that can be set arbitrarily while satisfying equation 8.8 for all the elements. It is thus demonstrated that the facets, for which flux values are deduced, belong to a 'co-tree'. Taking the example of a tetrahedron, it is possible to fix the flux on three of its facets. As we have indicated, the flux is imposed on the fourth facet in order to verify equation 8.8. From the edge-node representation, a tree \mathcal{A} can be constructed connecting all the nodes representing the elements, without forming a closed loop. It is thus shown that the facets, which correspond to the edges not belonging to \mathcal{A} , form a facet tree.

By way of example, Figure 8.6 shows a facet co-tree corresponding to the mesh in Figure 8.4 and Figure 8.7 shows a facet tree.

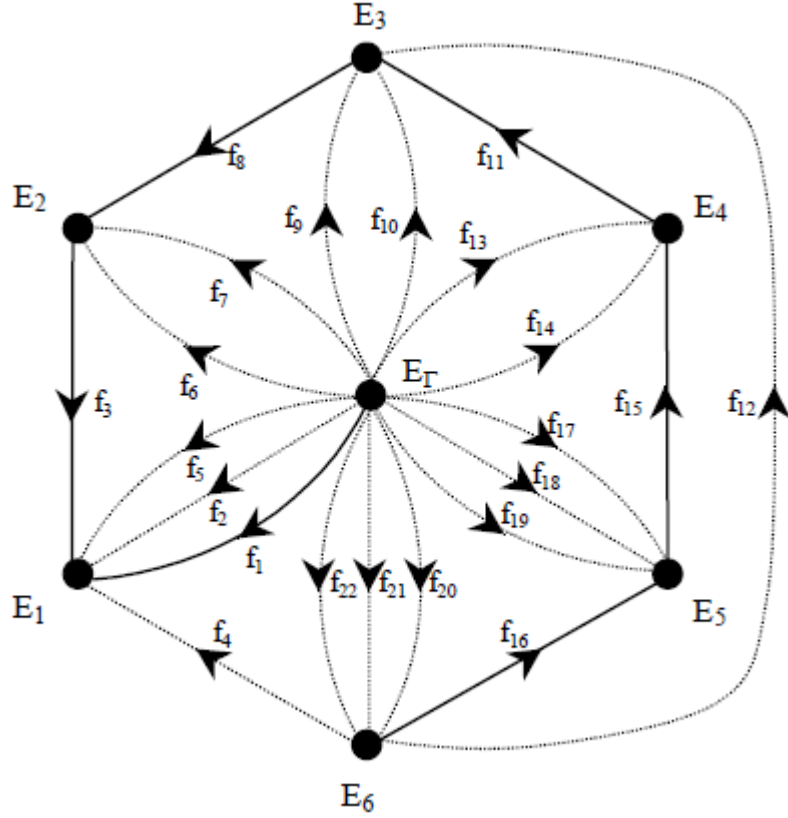


Figure 8.6: facet co-tree (solid lines)

8.2 Discretisation of \mathbf{K} and \mathbf{N}

In Chapter 3, by introducing two vector fields \mathbf{N} and \mathbf{K} , we developed the coupling of the potential formulations with the electrical equations of the circuits. We must determine which discrete spaces these two vectors belong to. To do this, we recall their definitions:

$$\mathbf{J}_s = I_s \mathbf{N} \quad (3.6)$$

$$\mathbf{H}_s = I_s \mathbf{K} \quad (3.7)$$

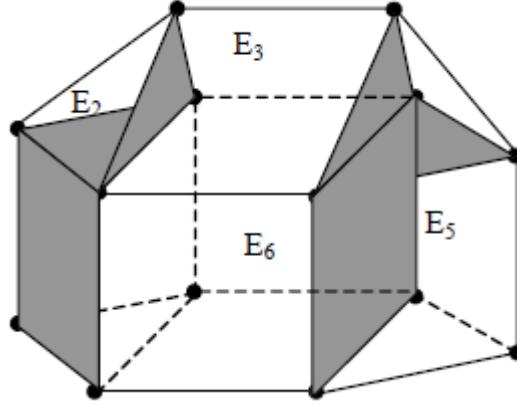


Figure 8.7: Representation of a facet tree (facets not greyed out)

In the discrete domain, two vectors are thus introduced \mathbf{N}^d and \mathbf{K}^d :

$$\mathbf{J}_s^d = I_s \mathbf{N}^d \quad (8.9)$$

$$\text{rot } \mathbf{K}^d = \mathbf{N}^d \quad (8.10)$$

hence the following set memberships:

$$\begin{aligned} \mathbf{N}^d &\in \mathcal{W}^2 \\ \mathbf{K}^d &\in \mathcal{W}^1 \end{aligned}$$

8.2.1 Discretisation of \mathbf{N}

To obtain a divergence of \mathbf{N} equal to zero, several methods may be considered. Some express vector \mathbf{N} on the basis of a source vector potential [Ren 1996b], [Golovanov 1997] which naturally ensures the conservation of vector \mathbf{N} . This potential is obtained either analytically for inductors of simple shape or by minimisation of a functional, which requires finite element calculation. The curl of the potential is then introduced into the formulations as a source term.

Other methods involve searching for a zero divergence vector \mathbf{N} without using the artifice of a vector potential. Using tensor conductivity and an electrokinetic calculation, it is possible to obtain a current density, i.e. \mathbf{N} at given I_s , that is uniform at zero divergence [Dular et al 1996].

Another technique is to introduce two scalar potentials defined on the inductor surfaces. The vector product of the potential gradients indicates the direction of current density [Kameari, Koganezawa 1997].

Hence, we suggest an alternative method that does not require finite element calculation and which applies to wound inductors with a constant cross-section.

To discretise the current density \mathbf{J}_s , and thus vector \mathbf{N} , four conditions must be met:

- the discretised current density must be as close as possible to the actual current density;
- it must be broken down in the space of the facet elements;
- the boundary conditions are: $\mathbf{J} \cdot \mathbf{n} = 0$ on the outer envelope of the inductor and $\mathbf{J} \cdot \mathbf{n} = J$ on Γ_b ;

- its divergence must be zero on all mesh elements.

Consider the inductor in Figure 8.8 meshed with tetrahedra.

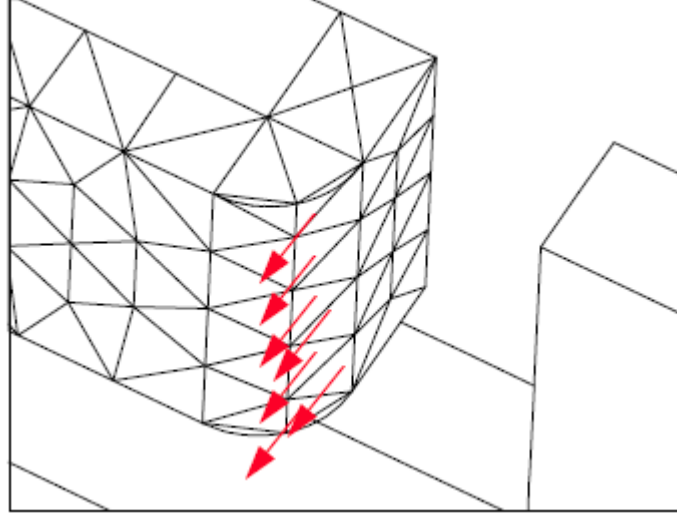


Figure 8.8: Space discretisation error leading to an outgoing current density.

As shown in Figure 8.8, the mesh does not exactly match the shape of the inductor in the corners. Hence, if we directly project \mathbf{J}_s in \mathcal{W}^2 , the flux $j_f^{d'}$ through a facet 'f' of the inductor is written:

$$j_f^{d'} = \int_{S_f} \mathbf{J}_s \cdot \mathbf{n} dS \quad (8.11)$$

where S_f is the surface of the facet surface \mathbf{n} its normal. The facet fluxes not belonging to the inductor are zero. The current density $\mathbf{J}_s^{d'}$ thus broken down is expressed as:

$$\mathbf{J}_s^{d'} = \sum_{f \in \mathcal{F}} j_f^{d'} \mathbf{w}_f \quad (8.12)$$

Under these conditions, let us consider an element 'v' exterior to the inductor that has one of its facets in contact with the corner of the coil. Because of the discretisation, the flux $j_f^{d'}$ is not zero through this facet. However, it is zero on the other facets of element 'v' because they do not belong to the inductor. As a result, not only is the divergence of $\mathbf{J}_s^{d'}$ not zero, but a current density also appears in this element.

To construct a zero divergence field \mathbf{J}_s^d in \mathcal{W}^2 and close to \mathbf{J}_s , we propose a method based on the use of a facet tree. We denote F_{ext} the set of facets that belong to the outer surface of the inductor and F_b the set of facets of the conductor in contact with Γ_b . The tree is constructed by including all facets of F_{ext} and F_b except one (otherwise a closed surface is created). On the facets of F_{ext} , a zero flow ($\mathbf{J} \cdot \mathbf{n} = 0$) is imposed. On the facets of F_b and on the other tree facets we impose:

$$j_f^{d'} = \int_{S_f} \mathbf{J}_s \cdot \mathbf{n} dS \quad (8.13)$$

Since the flux $j_f^{d'}$ equal to zero is strongly imposed on the outer facets of the inductor, there will be no outgoing current through the facets F_{ext} . Inside the inductor, the flow is calculated on the facets of the co-tree by imposing zero divergence on each element, namely:

$$\sum_{f \in V} j_f^{d'} = 0 \quad (8.14)$$

The current density vector in \mathcal{W}^2 thus obtained is close to \mathbf{J}_s while having zero divergence. From this vector, we can thus calculate a source field \mathbf{H}_s belonging to \mathcal{W}^1 such that:

$$\mathbf{rot} \mathbf{H}_s = \mathbf{J}_s$$

8.2.2 Discretisation of K

There are several ways to determine the source field \mathbf{H}_s . For inductors of simple shape, it can be calculated analytically [Kladas, Tegopoulos 1992], [Bouissou 1994], [Nakata et al 1988].

The field \mathbf{H}_s can be determined by minimising the difference between $\mathbf{rot} \mathbf{H}_s$ and the current density \mathbf{J}_s flowing in the inductor [Golovanov 1997]. This calculation can be performed on a sub-domain of \mathcal{D} containing the inductor. It is then possible to choose \mathbf{H}_s in \mathcal{W}^1 . Finally, if we have a current density \mathbf{J}_s^d belonging to the space of the facet elements, it is possible to determine the source field by an iterative method [Webb, Forghani 1989] [Biro et al 1993b] [Le Menach et al 1998]. \mathbf{H}_s must fulfil two conditions:

$$\begin{cases} \mathbf{rot} \mathbf{H}_s &= \mathbf{J}_s^d \\ \mathbf{H}_s &\in \mathcal{W}^1 \end{cases}$$

There is an infinite number of fields fulfilling these conditions. To ensure uniqueness, an edge tree is used (see gauge conditions). On the edges of this tree, we impose circulations of \mathbf{H}_s at arbitrary values (zero, for example). Circulations of \mathbf{H}_s on the co-tree edges are calculated iteratively by checking Ampère's circuital law for each mesh facet. Figure 8.9 illustrates the application of this theorem for a triangular facet.

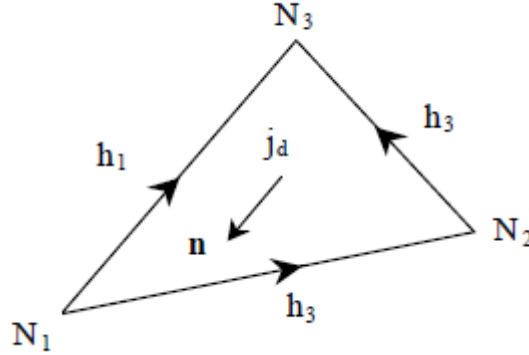


Figure 8.9: Facet crossed by a current j_d and definition of the circulations of \mathbf{H}_s .

In the magnetostatic or magnetodynamic case, it is useful to impose a field \mathbf{H}_s with a zero tangential component on Γ_h . To ensure this condition, the construction of the tree begins first on this surface and then spreads to the whole domain.

8.3 Discretisation of α et β

Function β is discretised on the mesh edges and function α at the nodes. In practise, β is not determined. Only α and its gradient are evaluated.

We define a function α such that [Dular, Legos 1998]:

$$\alpha = \sum_{n \in N_{\Gamma_c^1}} w_n \quad (8.15)$$

with $N_{\Gamma_c^1}$ the set of nodes of Γ_c^1 and w_n the nodal function associated with node n . We note that function α belongs to \mathcal{W}^0 . This is equal to 1 on Γ_c^1 and zero outside a domain \mathcal{D}_α defined by the set of elements containing at least one node of Γ_c^1 . In \mathcal{D}_α , function α varies continuously from 1 on the boundary Γ_c^1 to 0 on the boundary of \mathcal{D}_α . From this, we can define α_n the vector of the node values of α of components α_n with $1 \leq n \leq n_n$. The components of α_n are defined by:

$$\alpha_n = 1 \quad \text{si } n \in N_{\Gamma_c^1} \quad (8.16)$$

$$\alpha_n = 0 \quad \text{ailleurs} \quad (8.17)$$

8.4 Discretisation of the current density of a wound inductor

8.4.1 Introduction

The complexity of the domains and equations in electromagnetic modelling means that discretisation methods must be used to solve them. These methods require discretisation of the domain as well as various fields, especially the source term, which may be the current density \mathbf{J} . In this specific case, however, the discretised current density must respect a number of highly restrictive characteristics: zero divergence to ensure compatibility of the source term [Ren et al 1996], no current output on the edges of the domain, boundary conditions for the input or output of the current, etc., while remaining as close as possible to the exact current density.

An effective way to achieve zero divergence is to use graph theory, making use of a facet tree [Bossavit 1993], [Dlotko et al 2011], [Le Menach et al 1998]. A formulation of the type $\mathbf{J} = \mathbf{rotT}$, [Golovanov et al 1999], can be used to minimise the error between exact and discrete current densities. There is also the possibility of resolving a matrix system guaranteeing the divergence and error minimisation conditions [Badics et al 2007].

This idea is taken up here, but still making use of the facet tree technique to solve the matrix system. We will apply this method to a volume in which the edge mesh introduces a strong variation in the cross-section and low compliance with the geometric dimensions.

8.4.2 Discretisation using a Whitney complex

We break down a simply connected domain \mathcal{D} , of edge $\partial\mathcal{D}$, into E elements and we denote:

- \mathbf{F} the set of F facets.
- \mathbf{E} the set of E elements.
- N_f the number of facets per element.
- $\partial\mathcal{D}_{es}$ the current input and output edges.

We seek to discretise a uniform current density \mathbf{J} . The discretised current density vector belongs to the space of the facet elements [Bossavit 1993], and is thus written as the following linear combination:

$$\mathbf{J}_d = \sum_{f \in \mathbf{F}} \Phi_f \mathcal{F}_f \quad (8.18)$$

where \mathcal{F}_f represents the facet function associated with facet f , and Φ_f represents the flux of the current density vector \mathbf{J} through facet f , i.e. $\int_f \mathbf{J} \cdot \mathbf{n}_f ds$, \mathbf{n}_f representing the unit normal of facet f .

To discretise the source term \mathbf{J} of the current density, it must be ensured that the following conditions are met:

- $\text{div} \mathbf{J}_d = 0$,
- the discretised current density \mathbf{J}_d must be as close as possible to the actual current density \mathbf{J} ,
- $\mathbf{J}_d \cdot \mathbf{n} = 0$ on $\partial \mathcal{D} \setminus \partial \mathcal{D}_{es}$.

8.4.2.1 Incidence matrix

Let us consider constraint $\text{div} \mathbf{J}_d = 0$ for all $e \in \mathbf{E}$, and integrate to use the Green-Ostrogradski theorem:

$$\int_e \text{div} \mathbf{J}_d dv = \int_{\partial e} \mathbf{J}_d \cdot \mathbf{n} dS = \sum_{f \in e} \Phi_f = 0 \quad (8.19)$$

We translate this equation into a matrix system:

$$D\Phi = 0, \quad (8.20)$$

where $\Phi_i = \Phi_{f_i}$, flux of \mathbf{J} through f_i , i -th facet of \mathbf{F} .

Matrix D is of dimension $\mathbf{E} \times \mathbf{F}$, with $\mathbf{F} > \mathbf{E}$. A row corresponds to an element of the mesh, a column to a facet. This matrix corresponds to the discrete divergence operator, but also to the facet–element incidence matrix [Bossavit 1993]. For the reasons developed in [Ren et al 1996], this condition must be strongly verified.

8.4.2.2 Mass matrix

To ensure a vector \mathbf{J}_d that is as close as possible to the exact current density, we will try to minimise the norm of the difference between \mathbf{J}_d and \mathbf{J} [Badics et al 2007]:

$$\varepsilon = \int_{\mathcal{D}} (\mathbf{J}_d - \mathbf{J})^2 dv \quad (8.21)$$

Developing \mathbf{J}_d using its equation (8.18) and by differentiating ε with respect to the variable Φ_{f_i} , the minimisation problem becomes:

$$\sum_{f \in \mathbf{F}} \Phi_f \int_{\mathcal{D}} \mathcal{F}_f \cdot \mathcal{F}_{f_i} dv = \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathbf{J} dv \quad (8.22)$$

By considering this equation for all $f_i \in \mathbf{F}$, we obtain the following matrix system, of size $\mathbf{F} \times \mathbf{F}$:

$$M\Phi = v \quad (8.23)$$

where

$$M_{i,j} = \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathcal{F}_{f_j} dv, \quad v_i = \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathbf{J} dv.$$

Matrix M is conventionally called the *mass matrix* of the facet functions.

Equations (8.20) and (8.23) thus form an overall system of size $(\mathbf{E} + \mathbf{F}) \times \mathbf{F}$:

$$\begin{bmatrix} D \\ M \end{bmatrix} \Phi = \begin{bmatrix} 0 \\ v \end{bmatrix} \quad (8.24)$$

8.4.3 Use of the facet tree

The tree used here is an edge and facet tree [Bossavit 1993], [Dlotko et al 2011], [Le Menach et al 1998]: a vertex of the tree represents a mesh facet, while a link represents an edge. This tree includes all edge facets, except one, and some of the internal facets. All remaining facets form what is called the co-tree, which is a facet–element tree.

Use of the facet tree allows the set of facets to be split into two groups: the tree facets, and those of the co-tree. This means that matrix D is separated into two matrices A and C , the tree matrix and the co-tree respectively, such that:

$$D\Phi = [C, A] \begin{bmatrix} \Phi_C \\ \Phi_A \end{bmatrix} = 0 \quad (8.25)$$

with

- $C, E \times E$, invertible, where the columns represent the co-tree facets,
- $A, E \times (F - E)$, where the columns represent the tree facets.

This leads to the possibility of expressing the fluxes across the co-tree facets as a function of the fluxes across the tree facets:

$$\Phi_C = -C^{-1}A\Phi_A \quad (8.26)$$

Similarly, we can separate the system (8.23) as a function of Φ_C and Φ_A :

$$[M_C, M_A] \begin{bmatrix} \Phi_C \\ \Phi_A \end{bmatrix} = v \iff M_C\Phi_C + M_A\Phi_A = v \quad (8.27)$$

However, $\Phi_C = -C^{-1}A\Phi_A$, so system (8.27) is finally written:

$$(-M_C C^{-1}A + M_A)\Phi_A = v \quad (8.28)$$

This system is of size $F \times (F - E)$, and it is overdetermined. By a method of least squares, we obtain the vector Φ_A then deduce Φ_C from equation (8.26). Note that the condition $\text{div}\mathbf{J}_d = 0$ is fulfilled.

8.4.4 Inversion of the co-tree matrix

Matrix C is made up of 0, 1 and -1 , and it has a set of rows with a single non-zero term, another set with two non-zero terms, ..., and finally a set of rows with N_f non-zero terms. This characteristic of the matrix allows implementation of an effective algorithm for obtaining C^{-1} .

8.4.5 Application to an elbow of circular cross-section

We discretise the unit current density in an elbow of circular cross-section using `code_Carmel` developed by EDF R&D and L2EP. The mesh of the domain is shown in Figure 8.10.

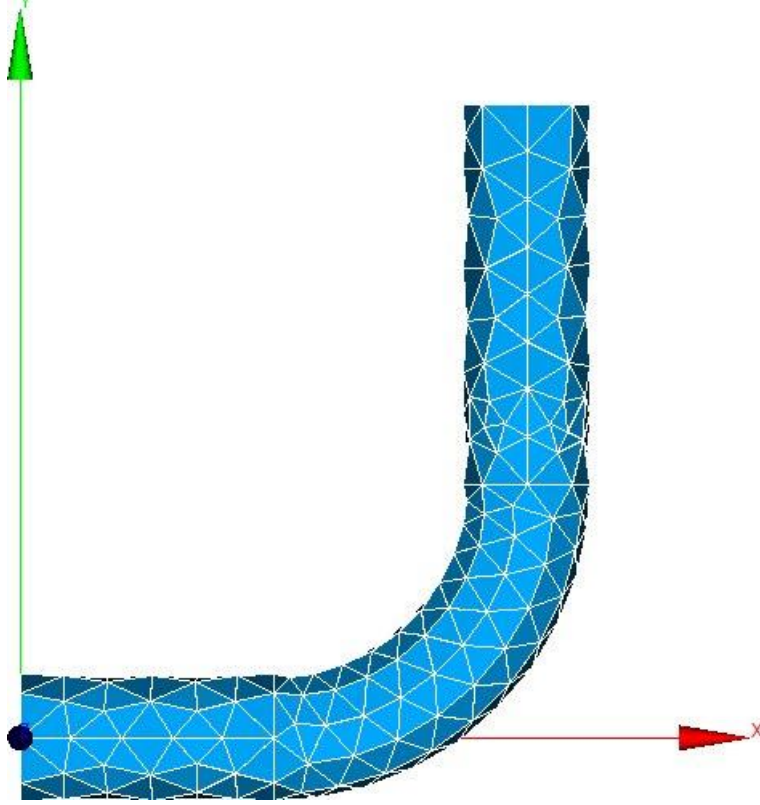
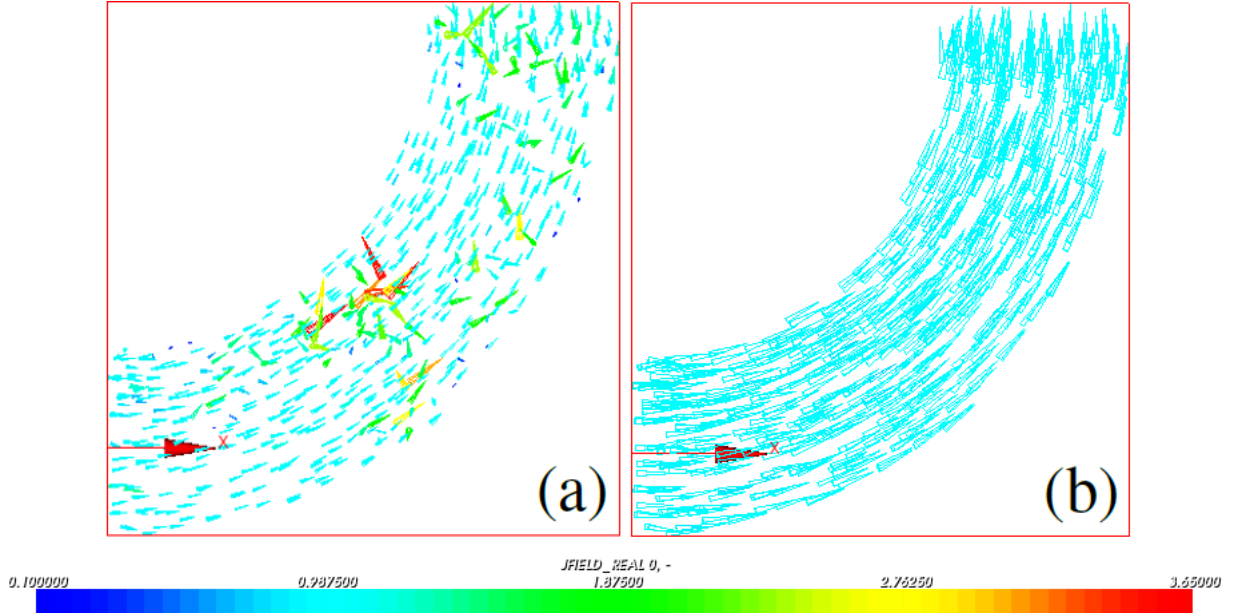


Figure 8.10: Mesh of an elbow of circular cross-section.

We focus on the bent area to view the field, as the error is located there.

Figure 8.11: (a) \mathbf{J}_d before minimisation. (b) \mathbf{J}_d after minimisation.

The field in Figure 8.11.(a) represents the current density obtained using only equation (8.26):

the fluxes are fixed on the tree facets with their exact values, then the fluxes on the co-tree facets are deduced from equation (8.26). Hence, the divergence is zero.

Figure 8.11.(b) shows the current density obtained by the error minimisation method: zero divergence is ensured, and the error with respect to the exact current density is minimised.

8.4.6 Conclusion

Using the technique presented, we have forced a divergence equal to zero while minimising the error between the exact and approximate current density fields using the facet tree method.

8.5 Imposing a uniform current per section in any conductor

This work comes from [Pierquin 2011].

Earlier version of code_Carmel could only deal with inductors made of straight parts or parts obtained by rotation about an axis. Indeed, the current density is easy to calculate in both these cases. This fact was highly restrictive, in terms of limitation of the possible geometries, but also in terms of use. It was necessary to break down the solid into different parts and find for each part:

- the current direction in the straight parts;
- a point and a vector forming the rotation axis in the bent parts.

Hence, the Salome platform enables the creation of complex geometries through the use of numerous tools. A classic example that shows the limitations of earlier version of code_Carmel, is the ability to place different points and plot the spline curve passing through those points.

Then, to create a solid, it suffices to place a flat face (disk, rectangle, etc.) at one end of this curve and translate it along the curve. This process is called *extrusion along a curve*. Such a construction clearly highlights the impossibility of addressing this problem using only the tools initially offered in code_Carmel.

It is based on this construction method that this new functionality will be introduced.

8.5.1 Use of a guideline

As it is impossible, or tedious, to search for a function that defines current density \mathbf{J}_s at any point in the inductor, or to deal with continuous cases, the idea is to use a discretised guideline of known density \mathbf{J}_s , and to deduce from this the current density vector at any point in the domain. In fact, the only data to be obtained is the direction of vector \mathbf{J}_s , the norm being constant.

This method is described by distinguishing the case of a constant extrusion cross-section from the case of a non-constant extrusion cross-section, which is nothing other than a geometric construction.

8.5.2 Case of a constant cross-section

8.5.2.1 Description of the method

The solid is meshed, and the guide curve, denoted \mathcal{C} , is then discretised. We denote $(X_n)_{n=0}^N$ the set of points of \mathcal{C} and \mathcal{D} the domain defined by the solid.

The first step is to identify which point of the sequence $(X_n)_{n=0}^N$ is closest to Y . This point is denoted X_j . Once X_j is identified, it remains to determine if Y should be associated with the direction $X_{j+1} - X_j$ or $X_j - X_{j-1}$.

To do this, we introduce the plane \mathcal{P}_j which is the orthogonal plane to the line $\delta_j = (X_{j-1}, X_{j+1})$ and passing through X_j , and $X_{P,\delta}^{(j)}$ their point of intersection. The vector \mathbf{J}_s is the same at any point of \mathcal{P}_j , namely $\mathbf{J}_s(X_j)$. By projecting point Y , following \mathcal{P}_j , on line δ_j , we obtain point $X_{Y,\delta}^{(j)}$.

We then perform the scalar product $\langle X_{j+1} - X_{j-1}, X_{Y,\delta}^{(j)} - X_{P,\delta}^{(j)} \rangle$:

- if $\langle X_{j+1} - X_{j-1}, X_{Y,\delta}^{(j)} - X_{P,\delta}^{(j)} \rangle \geq 0$ then we associate Y with the direction $d = d^+ = X_{j+1} - X_j$;
- if $\langle X_{j+1} - X_{j-1}, X_{Y,\delta}^{(j)} - X_{P,\delta}^{(j)} \rangle < 0$ then we associate Y with the direction $d = d^- = X_j - X_{j-1}$.

Once this direction is known, we must find the points of intersection \tilde{X}_j and $\tilde{X}_{j\pm 1}$ between line (Y, d^\pm) and the planes \mathcal{P}_j and $\mathcal{P}_{j\pm 1}$.

All that remains is to calculate:

$$\lambda = \frac{\|Y - \tilde{X}_j\|}{\|\tilde{X}_{j\pm 1} - \tilde{X}_j\|}$$

to obtain \mathbf{J}_s from the equation:

$$\mathbf{J}_s(Y) = \lambda \mathbf{J}_s(X_{j\pm 1}) + (1 - \lambda) \mathbf{J}_s(X_j)$$

8.5.2.2 Illustration of the principle

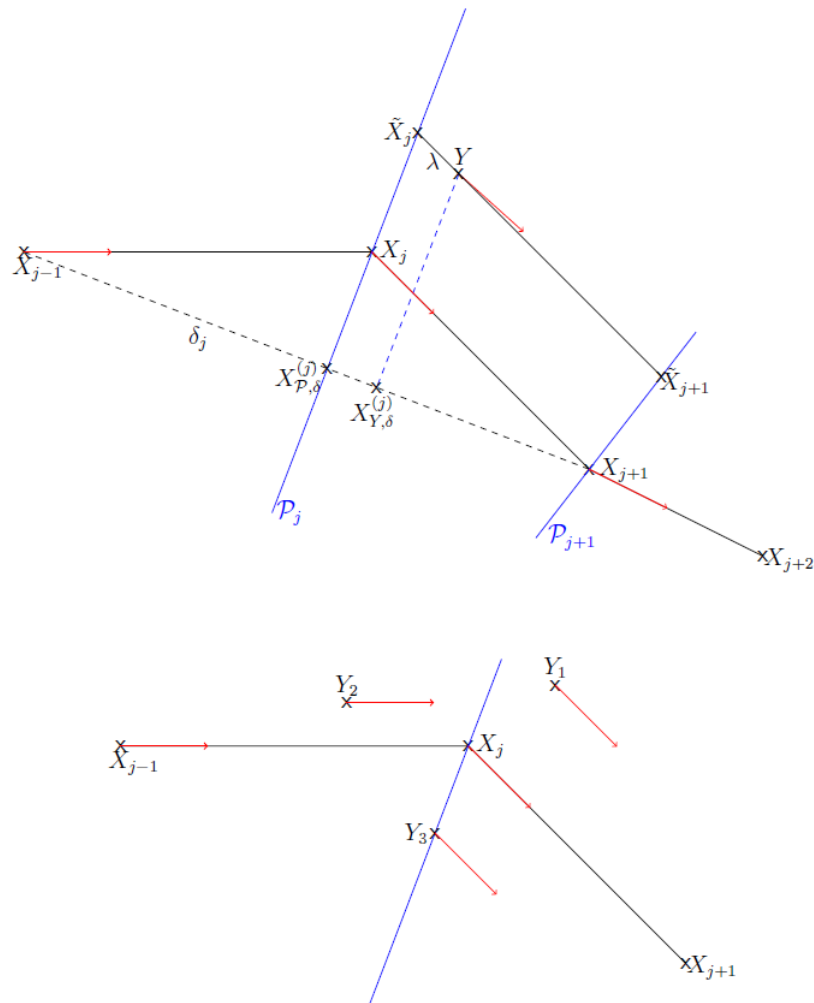


Figure 8.12: Principle of the method

8.5.2.3 Implementation of the academic facet tree

The discretisation of the current density requires the use of a facet tree. A facet tree is a simple graph, i.e. a set of vertices connected to each other by links, such that:

- two vertices are connected by a single link.
- there is no loop, i.e. there is only one path to connect two vertices.

The tree created here is an edge-facet tree: a vertex of the tree represents a mesh facet, while a link represents an edge.

This tree includes all edge facets, except one, as well as some of the internal facets. All the remaining facets form what we call the co-tree. This is a facet–element tree.

The technique consists in fixing the flux on the tree facets, and then deducing the flux value on the co-tree facets to conserve zero divergence:

$$\forall e \in \mathbf{E}, \quad \text{div } \mathbf{J}|_e = 0$$

$$\int_e \text{div } \mathbf{J} \, dv = \int_{\partial e} \mathbf{J} \cdot \mathbf{n} \, dS = 0$$

$$\sum_{f \in e} \int_f \mathbf{J} \cdot \mathbf{n}_f \, dS = 0$$

$$\sum_{f \in e} \Phi_f = 0$$

We will focus on describing how to obtain the academic tree, and how it is used to complete the values on the co-tree facets.

8.5.2.4 Obtaining the academic facet tree

In reality, the tree facets are those that are not in the co-tree, and it is the co-tree that is obtained first. The co-tree is a facet-element graph. The algorithm is based on the creation of several numbered co-trees, merged on moving through the elements.

Denoting \mathbf{F} the set of facets, the algorithm is:

Algorithm 8.1 Academic facet trees.

```

1: Entrées: Choose one facet on the edge of the domain: the root facet, corresponding to the
   valve. This facet has the number of co-tree 1; the elements separated by the facet are marked
   1.
2: for  $f_i \in \mathbf{F}$  do
3:     if the facet is internal to the domain, i.e. it is not on the boundary then
4:         obtain the elements  $e$  and  $\tilde{e}$  separated by the facet.
5:         if neither of the two elements is marked then
6:             they form a new co-tree:  $f_i$ ,  $e$  and  $\tilde{e}$  are marked  $n$ ,  $n$  is
             incremented ( $n = n + 1$ ).
7:         end if
8:         if only one of the two elements is marked then
9:             the facet is added to the co-tree of the marked element, as
             well as the unmarked element: for example,  $e$  is marked  $k$ ,
             hence  $f_i$  and  $\tilde{e}$  are marked  $k$ .
10:        end if
11:        if the elements are in different co-trees then
12:            the lowest co-tree number is assigned to all elements and all
            facets of these co-trees; facet  $f_i$  will not be in a tree.
13:        end if
14:    end if
15: end for

```

At the end of this algorithm, the co-tree facets are marked with a 1, and the tree facets with a 0.

8.5.2.5 Use of the academic facet tree

Once the facet tree has been obtained, the principle is to calculate the fluxes of the current density through the facets belonging to the tree, from the analytical vector \mathbf{J}_s ; and from this to deduce the fluxes on the co-tree facets to maintain zero divergence (see paragraph 8.5.2.3).

To obtain the fluxes on the co-tree facets, the number of facets to be completed per element is indicated, then the following algorithm is used:

Algorithm 8.2 Calculation of the fluxes on the co-tree facets.

```

1: full = FALSE
2: while ( cpt < nbElmts and full=FALSE ) do
3:     full = TRUE;
4:     for  $e_i \in \mathcal{E}$  do
5:         if only one facet  $f$  is to be completed on element  $e_i$  then
6:             sum the known fluxes as a function of the direction of the
              normal of the facet;
7:             deduce the fluxes on facet  $f$  to be completed;
8:             decrease by 1 the number of facets to be completed in ele-
              ments  $e, \tilde{e}$  separated by  $f$ .
9:         else
10:            full=FALSE
11:        end if
12:    end for
13:    counter increment (cpt=cpt+1)
14: end while

```

This algorithm has the disadvantage of systematically traversing all mesh elements of the mesh, which unlike the DFS tree is not optimal. Nevertheless, the calculation time is not noticeably increased.

8.5.2.6 The minimisation method

We have seen that the discretisation of the current density vector is performed in such a way as to guarantee zero divergence, and this through the use of a facet tree. But this technique imposes zero divergence without any control over the \mathbf{J}_s^d constructed.

The paper [Badics, Cendes 2007] shows that it is possible to impose zero divergence while minimising the error between the known current density vector and the discretised vector. We propose an adaptation of this method, known as the minimisation method, which still uses the facet tree.

We consider a mesh of domain \mathcal{D} with the following notation:

- \mathbb{N} the set of nodes;
- \mathbb{A} the set of edges;
- \mathbb{F} the set of facets;
- \mathbb{E} the set of elements.

and their respective cardinals N, A, F and E .

8.5.2.6.1 The divergence matrix

By the Green-Ostrogradski theorem:

$$\forall e \in \mathbb{E}, \quad \int_e \operatorname{div} \mathbf{J} = \sum_{f \in \partial e} \Phi_f = 0 \quad (8.29)$$

with Φ_f the flux of \mathbf{J} through facet f .

By translating this equation on each element, we obtain a matrix system :

$$D \Phi = 0 \quad (8.30)$$

Matrix D is of dimension $E \times F$, with $F > E$, with a row corresponding to a mesh element, and a column corresponding to a facet. This matrix also corresponds to the facet-element incidence matrix.

The facet tree can be used to split the set of facets into two groups: tree facets and co-tree facets. This means that matrix D is separated into two matrices C and A , the co-tree and tree matrices respectively, such that:

$$D \Phi = [C, A] \begin{bmatrix} \Phi_C \\ \Phi_A \end{bmatrix} = 0 \quad (8.31)$$

with:

- C , $E \times E$, invertible, where the columns represent the co-tree facets;
- A , $E \times (F - E)$ where the columns represent the tree facets.

This makes it possible to express the fluxes on the co-tree facets as a function of the fluxes of the tree facets:

$$\Phi_C = -C^{-1} A \Phi_A \quad (8.32)$$

Until now, we have simply calculated the fluxes on the facet tree, i.e. obtained Φ_A , and then deduced from this the fluxes on the co-tree, i.e. obtained Φ_C , using formula 8.32. We do not do that in this case, but we add a minimisation step.

8.5.2.6.2 The minimisation matrix

We saw earlier that the current density vector \mathbf{J} belongs to the space of the facet elements, hence:

$$\mathbf{J} = \sum_{f \in \mathbb{F}} \Phi_f \mathcal{F}_f \quad (8.33)$$

with:

- \mathcal{F}_f the facet functions;
- Φ_f the fluxes through the facets.

We now want to minimise the error between the analytical current density vector \mathbf{J}_s and the discrete current density vector \mathbf{J} . This means, considering the norm $L^2(D)$, minimising:

$$\varepsilon = \int_{\mathcal{D}} (\mathbf{J} - \mathbf{J}_s)^2 dv \quad (8.34)$$

Considering equation 8.34 and developing it:

$$\begin{aligned} \int_{\mathcal{D}} (\mathbf{J} - \mathbf{J}_s)^2 dv &= \int_{\mathcal{D}} \left(\sum_{f \in \mathbb{F}} \Phi_f \mathcal{F}_f - \mathbf{J}_s \right)^2 dv \\ &= \int_{\mathcal{D}} \left(\left(\sum_{f \in \mathbb{F}} \Phi_f \mathcal{F}_f \right)^2 - 2 \sum_{f \in \mathbb{F}} \Phi_f \mathcal{F}_f \cdot \mathbf{J}_s + \mathbf{J}_s^2 \right) dv \\ &= \underbrace{\int_{\mathcal{D}} \left(\sum_{f \in \mathbb{F}} \Phi_f \mathcal{F}_f \right)^2 dv}_{(1)} - 2 \sum_{f \in \mathbb{F}} \Phi_f \int_{\mathcal{D}} \mathcal{F}_f \cdot \mathbf{J}_s dv + \int_{\mathcal{D}} \mathbf{J}_s^2 dv \end{aligned} \quad (8.35)$$

Focusing on term (1) of equation 8.35:

$$\begin{aligned}
\int_{\mathcal{D}} \left(\sum_{f \in \mathbb{F}} \Phi_f \mathcal{F}_f \right)^2 dv &= \int_{\mathcal{D}} \left(\sum_{f \in \mathbb{F}} \Phi_f \mathcal{F}_f \right) \cdot \left(\sum_{f \in \mathbb{F}} \Phi_f \mathcal{F}_f \right) dv \\
&= \int_{\mathcal{D}} \left(\sum_{f \in \mathbb{F}} \Phi_f \sum_{g \in \mathbb{F}} \Phi_g \mathcal{F}_f \cdot \mathcal{F}_g \right) dv \\
&= \sum_{f \in \mathbb{F}} \sum_{g \in \mathbb{F}} \Phi_f \Phi_g \int_{\mathcal{D}} \mathcal{F}_f \cdot \mathcal{F}_g dv
\end{aligned} \tag{8.36}$$

Using equations 8.35 and 8.36, we thus seek to minimise:

$$\int_{\mathcal{D}} (\mathbf{J} - \mathbf{J}_s)^2 dv = \sum_{f \in \mathbb{F}} \sum_{g \in \mathbb{F}} \Phi_f \Phi_g \int_{\mathcal{D}} \mathcal{F}_f \cdot \mathcal{F}_g dv - 2 \sum_{f \in \mathbb{F}} \Phi_f \int_{\mathcal{D}} \mathcal{F}_f \cdot \mathbf{J}_s dv + \int_{\mathcal{D}} \mathbf{J}_s^2 dv \tag{8.37}$$

More precisely, we seek the set of $(\Phi_f)_{f \in \mathbb{F}}$ that minimises 8.37. However, seeking the minimum of a function is equivalent to seeking to cancel out its derivative; we thus differentiate 8.37 with respect to the variable Φ_{f_i} :

$$\frac{\partial \varepsilon}{\partial \Phi_{f_i}} = 2 \sum_{f \in \mathbb{F} \setminus \{f_i\}} \Phi_f \int_{\mathcal{D}} \mathcal{F}_f \cdot \mathcal{F}_{f_i} dv + 2 \Phi_{f_i} \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathcal{F}_{f_i} dv - 2 \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathbf{J}_s dv \tag{8.38}$$

Hence, we must solve $\frac{\partial \varepsilon}{\partial \Phi_{f_i}} = 0$, namely:

$$\sum_{f \in \mathbb{F}} \Phi_f \int_{\mathcal{D}} \mathcal{F}_f \cdot \mathcal{F}_{f_i} dv = \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathbf{J}_s dv \tag{8.39}$$

By considering this equation for all $f_i \in \mathbb{F}$ we obtain the following system matrix:

$$M \Phi = v \tag{8.40}$$

where:

- $M_{i,j} = \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathcal{F}_{f_j} dv$;
- $\Phi_i = \Phi_{f_i}$;
- $v_i = \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathbf{J}_s dv$

The equation can be written in matrix form:

$$\Phi^T M \Phi - 2 \Phi^T v + \int_{\mathcal{D}} \mathbf{J}_s^2 dv \tag{8.41}$$

Matrix M is conventionally called the mass matrix of the facet functions (see [Henneron 2004]).

We can separate this system as a function of Φ_C and Φ_A , as for the divergence matrix:

$$[M_C, M_A] \begin{bmatrix} \Phi_C \\ \Phi_A \end{bmatrix} = v \iff M_C \Phi_C + M_A \Phi_A = 0 \tag{8.42}$$

However, $\Phi_C = -C^{-1}A\Phi_A$, so the system 8.40 is finally written:

$$(-M_C C^{-1}A + M_A) \Phi_A = v \tag{8.43}$$

This system is of size $F \times (F - E)$ and it is overdetermined. This problem must then be solved by a least squares method, then Φ_C deduced from Φ_A using equation 8.32.

We have thus forced zero divergence while minimising the error between the exact current density vector and the approximate current density vector.

8.5.2.6.3 Taking account of boundary conditions

In the previous description of the minimisation method, we did not deal with the domain boundary conditions to avoid making the explanations too cumbersome. Following on from the previous results, we now add the information needed to process boundary conditions.

While the minimisation method does not require the fluxes to be fixed on all facets of the facet tree, it is necessary to fix the fluxes on the facets forming the domain boundary. Hence, for every facet belonging to the tree and also to the edge of the domain, the flux is fixed. The flux is calculated from the analytical current density if the facet is on a current input or output edge; otherwise it is set at 0.

Denoting Φ_B the set of fixed fluxes on the boundary and $\Phi_{\tilde{A}}$ the fluxes of tree facets that do not belong to the edges, and repeating the previous approach, starting by breaking down the divergence matrix:

$$D \Phi = 0 \iff D \Phi = [C, \tilde{A}, B] \begin{bmatrix} \Phi_C \\ \Phi_{\tilde{A}} \\ \Phi_B \end{bmatrix} = 0 \quad (8.44)$$

from which the expression for Φ_C :

$$\Phi_C = -C^{-1} (\tilde{A} \Phi_A + B \Phi_B) \quad (8.45)$$

Then the breakdown of the mass matrix:

$$M \Phi = v \iff [M_C, M_{\tilde{A}}, M_B] \begin{bmatrix} \Phi_C \\ \Phi_{\tilde{A}} \\ \Phi_B \end{bmatrix} = v \quad (8.46)$$

Making the system to be resolved:

$$(-M_C C^{-1} \tilde{A} + M_{\tilde{A}}) \Phi_A = v + (M_C C^{-1} B - M_B) \Phi_B \quad (8.47)$$

Resolving the system 8.47 by least squares thus reduces the error between the analytical current density vector and its discrete value (i.e. with a discrete vector as close as possible to the analytical), while ensuring zero divergence and exact fluxes at the edges.

Remark 8.5.1 *To reduce the size of the matrix and avoid unnecessary calculations, columns corresponding to zero-flux edges are not taken into account in matrix B (as their contribution is zero).*

Remark 8.5.2 *Matrix B represents only the current input/output edges, and hence does not appear in the case of closed inductors.*

Remark 8.5.3 *It should be noted that only one edge facet has a flux obtained by minimisation, as it does not belong to the facet tree; and it is clearly the valve.*

8.5.2.6.4 Inversion of the co-tree matrix

The use of facet trees and co-trees allows the divergence matrix to be broken down into several matrices, including matrix C , the co-tree matrix. This matrix is square, of size $E \times E$ and invertible. Our method precisely involves obtaining the inverse of this matrix. However, matrix inversion is usually very costly in computational time and best avoided. Nevertheless, due to the very particular construction of this matrix, it is possible to set up an efficient algorithm to obtain C^{-1} , the inverse matrix of C .

By the design of the facet tree and its co-tree, matrix C is made up of 1 and -1, and it has a set of rows with a single non-zero value, a set of rows with two non-zero terms, and another set with three non-zero terms, etc. This is represented in two dimensions in the figure below.

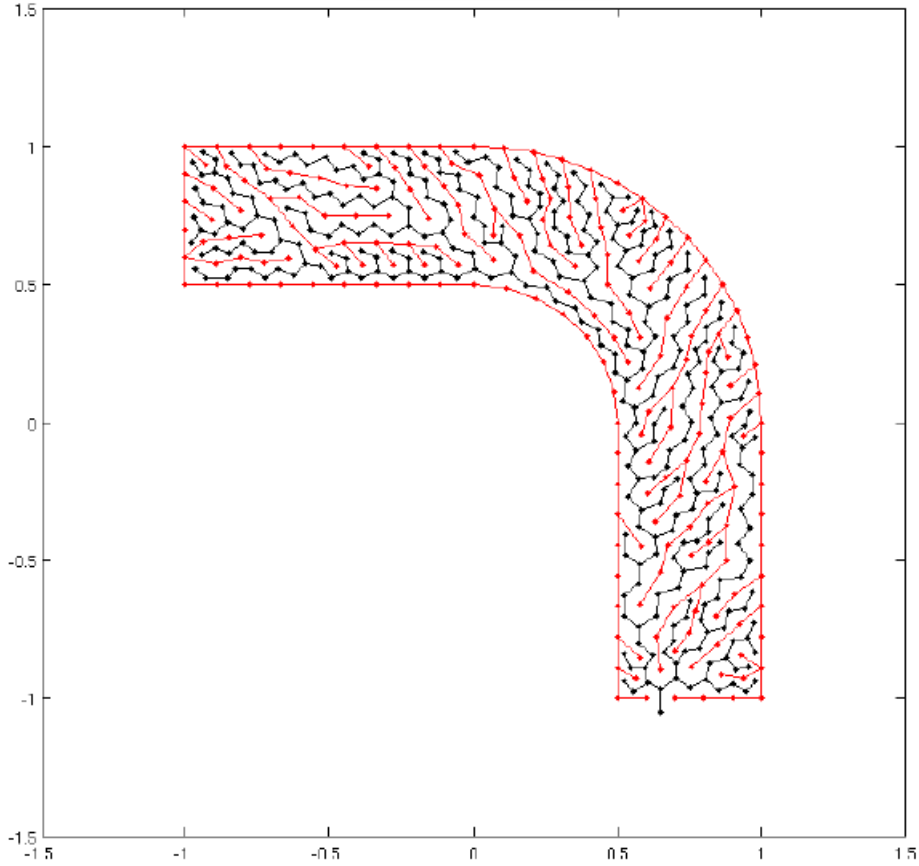


Figure 8.13: Facet tree (in red) and co-tree (in black) in two dimensions

It is this characteristic, in the same way that it allowed use of the algorithm in section 8.5.2.5 to deduce the fluxes on the co-tree, that allows inversion of matrix C .

We look for matrix C^{-1} such that if $Cx = y$ then $w = C^{-1}y$. We shall see that it is possible to express a given x_j as a linear combination of $(y_\alpha)_\alpha : x_j = \sum_\alpha \gamma_\alpha y_\alpha$, with coefficients $(\gamma_\alpha)_\alpha$ to be determined.

Consider the rows in matrix C with a single non-zero value. They verify:

$$C_{i,j} x_j = y_i \implies x_j = \frac{y_i}{C_{i,j}} \implies C_{j,i}^{-1} = \frac{1}{C_{i,j}}$$

Next, consider the rows in matrix C with two non-zero values. They verify:

$$C_{i,j} x_j + C_{i,k} x_k = y_i, \quad x_k \text{ connu} \implies x_j = \frac{y_i - C_{i,k} x_k}{C_{i,j}}$$

$$\implies C_{j,i}^{-1} = \frac{1}{C_{i,j}}; \quad C_{j,\alpha}^{-1} = -\frac{C_{i,k}}{C_{i,j}} \gamma_\alpha; \quad \forall \alpha \text{ tels que } x_h = \sum_{\alpha} \gamma_\alpha y_\alpha, \gamma_\alpha \neq 0$$

Similarly, considering the rows in matrix C with three non-zero values:

$$C_{i,j} x_j + C_{i,k} x_k + C_{i,l} x_l = y_i, \quad x_k, x_l \text{ connus} \implies x_j = \frac{y_i - C_{i,k} x_k - C_{i,l} x_l}{C_{i,j}}$$

$$\implies C_{j,i}^{-1} = \frac{1}{C_{i,j}}; \quad C_{j,\alpha}^{-1} = -\frac{C_{i,k}}{C_{i,j}} \gamma_\alpha; \quad \forall \alpha \text{ tels que } x_k = \sum_{\alpha} \gamma_\alpha y_\alpha, \gamma_\alpha \neq 0$$

$$C_{j,\beta}^{-1} = -\frac{C_{i,l}}{C_{i,j}} \gamma_\beta; \quad \forall \beta \text{ tels que } x_l = \sum_{\alpha} \gamma_\alpha y_\alpha, \gamma_\alpha \neq 0$$

It is possible to continue in this way for rows with four non-zero values, then five, six, etc. At the end, each x_j is written as a linear combination of $(y_i)_i$:

$$x_j = \sum_i \gamma_{j,i} y_i$$

the coefficients $\gamma_{j,i}$ representing the values of matrix $C_{j,i}^{-1}$.

The algorithm itself follows the method of algorithm 8.2. For each row in the inverse matrix, we only record column indices with non-zero values, as well as the associated factors:

Algorithm 8.3 Calculating the inverse of the mass matrix.

```

1: full = FALSE
2: while ( cpt < nbElmts and full = FALSE ) do
3:     full = TRUE;
4:     for  $e_i \in \mathcal{E}$  do
5:         if only one facet  $f$  is to be completed on element  $e_i$  then
6:             if only one facet is in the co-tree then
7:                 index(f) = i;
8:                 factor(f) = 1 /  $C_{i,f}$ ;
9:             else
10:                for each facet  $\tilde{f}$  of the co-tree in  $e_i$ ,  $\tilde{f} \neq f$  do
11:                    index(f) = [index(f), index( $\tilde{f}$ )];
12:                    factor(f) = [factor(f),  $-\frac{C_{i,\tilde{f}}}{C_{i,f}}$  factor( $\tilde{f}$ )]
13:                end for
14:                index(f) = i;
15:                factor(f) =  $\frac{1}{C_{i,f}}$ 
16:            end if
17:            decrease by 1 the number of facets to be completed in elements  $e$ ,  $\tilde{e}$ 
               separated by  $f$ 
18:        else
19:            full = FALSE
20:        end if
21:    end for
22:    counter increment (cpt=cpt+1)
23: end while

```

8.5.2.7 Calculation of the mass matrix and of right member

The mass matrix and the second member involve integrals, which cannot be calculated analytically. We begin by breaking down the integral on the domain into a sum of integrals on the elements:

$$\int_{\mathcal{D}} \dots = \sum_{e \in \mathcal{E}} \int_e \dots$$

we then calculate the numerical integral on the elements using quadrature formulas, as presented in the book [Dhatt, Thouzot 1984]. In the case of a tetrahedron e , we choose a formula with an order of 3 to 5 quadrature points:

$$\iiint_{\mathcal{D}} f(x, y, z) dv = \sum_{i=0}^5 w_i f(x_i, y_i, z_i)$$

with $(w_i)_i$ the weights associated with the values f at quadrature points $(x_i, y_i, z_i)_i$.

Consider the integral $\int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathcal{F}_{f_j} dv$ and focus on the facet function supports.

Facet function \mathcal{F}_{f_i} is defined on elements e_i, \tilde{e}_i , and function \mathcal{F}_{f_j} on elements e_j, \tilde{e}_j . Consider the various possible cases to calculate this integral:

$$f_i = f_j, \quad \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathcal{F}_{f_j} dv = \int_{e_i} \mathcal{F}_{f_i}^2 dv + \int_{e_j} \mathcal{F}_{f_i}^2 dv$$

$$e_j \in \{e_i, \tilde{e}_i\} \quad \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathcal{F}_{f_j} dv = \int_{e_j} \mathcal{F}_{f_i} \cdot \mathcal{F}_{f_j} dv$$

$$\tilde{e}_j \in \{e_i, \tilde{e}_i\} \quad \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathcal{F}_{f_j} dv = \int_{\tilde{e}_j} \mathcal{F}_{f_i} \cdot \mathcal{F}_{f_j} dv$$

$$e_j \notin \{e_i, \tilde{e}_i\} \text{ et } \tilde{e}_j \notin \{e_i, \tilde{e}_i\} \quad \int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathcal{F}_{f_j} dv = 0$$

Thus, the integral $\int_{\mathcal{D}} \mathcal{F}_{f_i} \cdot \mathcal{F}_{f_j} dv$ will be non-zero if $f_j \in e_i \cup \tilde{e}_i$, i.e. if the intersection of the facet function supports is other than the empty set: $\{e_i, \tilde{e}_i\} \cap \{e_j, \tilde{e}_j\} \neq \emptyset$.

8.6 Case of non-constant cross-section

Coils are wound inductors, usually consisting of a winding of copper wires. These wires are usually considered as one and the same entity, especially when it comes to modelling the current flowing through them.

In particular, some coils have wedges between the wire bundles. It is far too costly to model each bundle of wires independently, so it would be useful to model this system in one piece. Thus, although there is no such thing as a variable cross-section inductor, the wish to ignore the heterogeneity of the system (air, wedge, copper), and treat it as a single block, leads to the concept of an inductor of non-constant cross-section.

To illustrate the approach, we will construct the current density vector in a very specific 2D geometry, before correcting it using the facet tree technique. This test will very quickly prove the limitations of the facet tree for this type of problem.

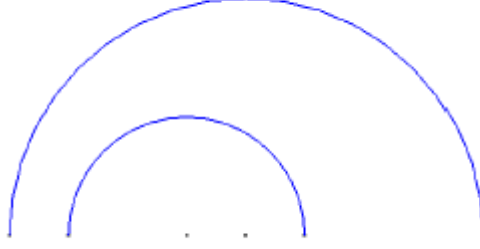


Figure 8.14: Semi-circles for a non-constant cross-section

8.6.1 Geometry

To build coils of non-constant cross-section using SALOME software, a very simple method is to create two half-cylinders with different axes, but in the same plane. For our two-dimensional tests, we consider two semi-circles whose centres are on the line formed by the ends of the arcs, illustrated below:

We introduce the following notation:

- C_{int} is the smaller semi-circle, the inner semi-circle;
- C_{ext} is the larger semi-circle, the outer semi-circle;
- $c_{int} = (0, 0)$ is the centre of C_{int} ;
- $c_{ext} = (x_{ext}, 0)$ is the centre of C_{ext} ;
- r_{int} is the radius of C_{int} ;
- r_{ext} is the radius of C_{ext} ;
- D is the domain formed by the semi-circles and the line $y = 0$;
- $s = x_{ext} + r_{ext} - r_{int}$ is the size of the final cross-section.

8.6.2 Calculation of the current density

The input current density vector is a unit vector. We construct \mathbf{J} such that its value decreases as the cross-section increases (to maintain zero divergence), such that it is tangential to the edges and its divergence is small.

Let there be a point $P = (x, y)$ in the domain. We know that if $P \in C_{int}$, P verifies:

$$x^2 + y^2 = r_{int}^2 \quad (8.48)$$

Similarly, if $P \in C_{ext}$:

$$(x - x_{ext})^2 + y^2 = r_{ext}^2 \quad (8.49)$$

For the other points of the domain, we will look for the centre $c_{var} = (x_{var}, 0) \in [c_{int}; c_{ext}]$ of a semi-circle C_{var} such that P verifies:

$$(x - x_{var})^2 + y^2 = |P - c_{var}|^2 \quad (8.50)$$

We know that if P is on a circle of centre c , then the expression for the unit current density vector at this point is:

$$\mathbf{J}_P = \|\mathbf{J}_P\| (\sin(\theta_c), -\cos(\theta_c)) \quad (8.51)$$

where θ_c is the angle between lines Ox and (c, P) .

Consider the point P in the polar coordinate system, for which c_{int} is the centre:

$$P = (\rho, \theta)$$

We know that if:

$$\rho = r_{int} \quad \text{alors} \quad P \in C_{int}$$

We look for a value of ρ such that point P belongs to circle C_{ext} ; in other words, we look for which pair of values of ρ, θ being fixed, satisfies equation 8.49:

$$(\rho \cos \theta - x_{ext})^2 + (\rho \sin \theta)^2 = r_{ext}^2$$

This amounts to solving the quadratic equation:

$$\rho^2 - \rho (2 x_{ext} \cos \theta) + x_{ext}^2 - r_{ext}^2 = 0 \quad (8.52)$$

Denoting ρ_{max} the solution of this equation. We can then obtain c_{var} as a function of ρ using the formula:

$$c_{var} = \frac{\rho - r_{int}}{\rho_{max} - r_{int}} c_{ext} \quad (8.53)$$

Denoting θ_{var} the value of the angle between the abscissa line and line (c_{var}, P) , this gives:

$$\mathbf{J}_P = \|\mathbf{J}_P\| (\sin \theta_{var}, -\cos \theta_{var}) \quad (8.54)$$

with:

$$\|\mathbf{J}_P\| = \frac{\pi - \theta}{s \pi} + \frac{\theta}{\pi} \quad (8.55)$$

8.6.3 Use of the facet tree

Once our current density vector has been calculated (see figure below), we perform a correction of \mathbf{J} to obtain zero flux on the edges, as well as a zero divergence.

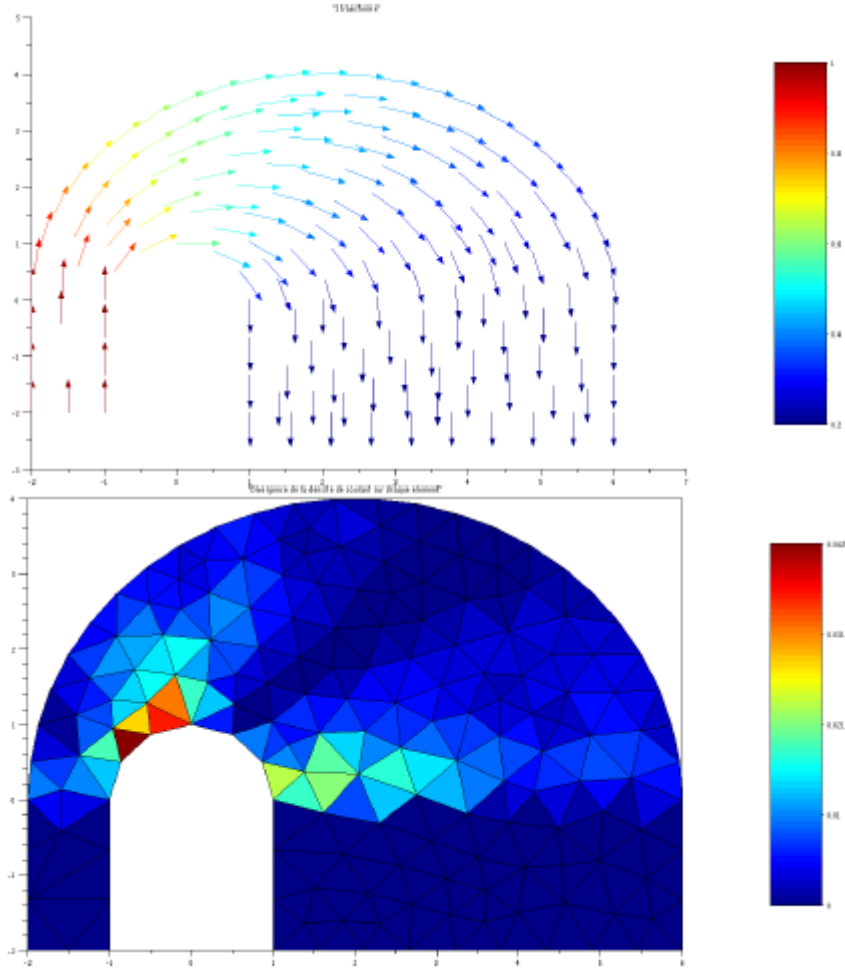
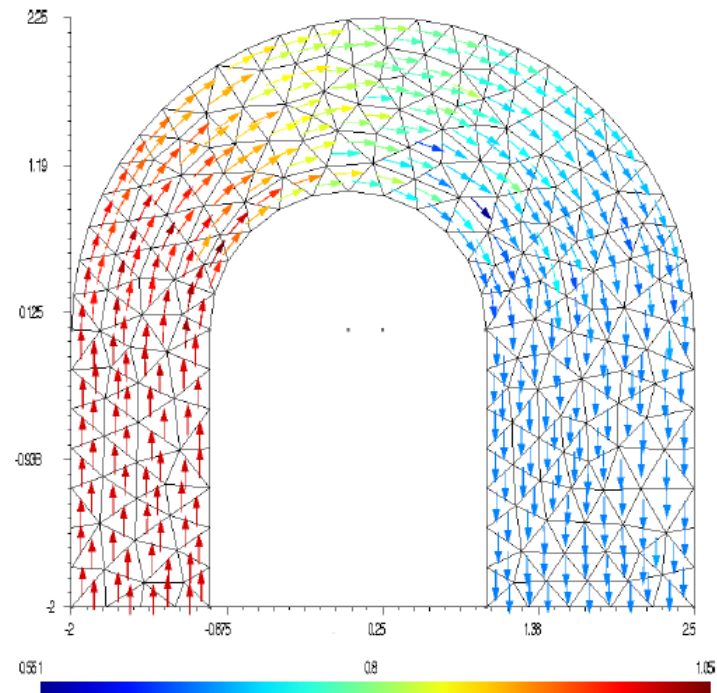


Figure 8.15: Exact \mathbf{J} in a non-constant cross-section and its divergence on the elements

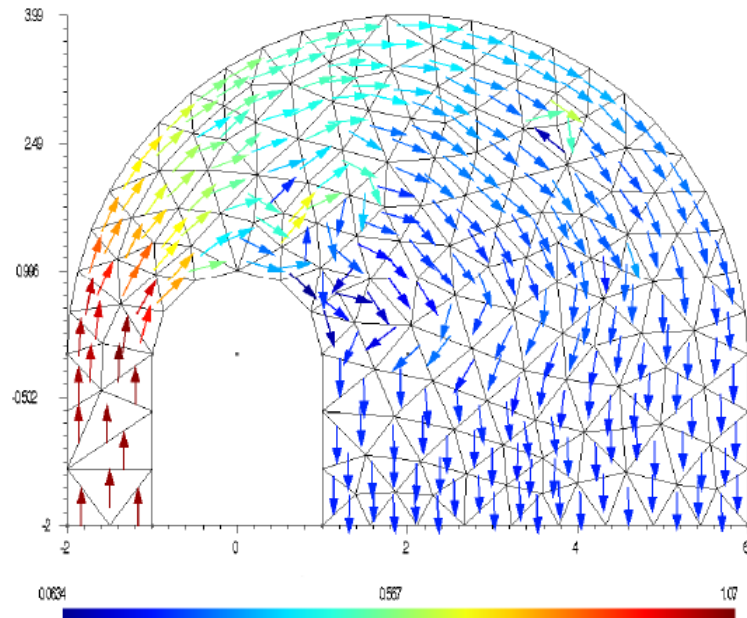
The use of the facet tree proves to be a disaster. For a small variation in cross-section variation, disturbance of the direction disturbance is acceptable, but the norm is already too heavily skewed. When the ratio between the input and output cross-sections is greater than 2, the current density vector is completely false. The facet tree is thus completely unusable in the case of a non-constant cross-section.

8.6.4 Minimisation method

While the facet tree does not work well, the minimisation method allows for cases of inductors with a limited variation in cross-section. While the previous phenomena do appear, they are mitigated. If the difference in cross-section is not too great, the current density vector is acceptable in terms of direction, but its norm already shows a wide variation (see figure below).

Figure 8.16: \mathbf{J} obtained by minimisation for a non-constant cross-section

However, as with the facet tree alone, if the variation in cross-section is too great, the current density is totally wrong (see figure below).

Figure 8.17: \mathbf{J} obtained by minimisation for a non-constant cross-section

Chapter 9

Discretisation of weak forms in code_Carmel

Abstract

We will project the different values into the discrete function spaces using simple interpolation functions. For the time-based version of code_Carmel, the time differentiation in the magneto-dynamic and circuit coupling equations will be discretised using the backward Euler method. For the spectral version of code_Carmel, a specific approach is used.

9.1 Discrete function spaces

In practice, the discretisation of continuous Hilbert spaces by the finite element method is based on a mesh of domain \mathcal{D}_h . Here, this means cutting up the domain under study into simple polyhedra respecting the different boundaries between the media. A mesh thus designates the set of volumes, faces, edges and nodes. On a given mesh, there is an infinite number of discrete sub-spaces available to approach the continuous Hilbert spaces defined earlier.

Here, it has been chosen to represent the discretised spaces using lowest-order Whitney elements. In addition to being simple, they have the advantage of associating each space class ($H^1(\mathcal{D})$, $\mathbf{H}(\mathbf{rot}, \mathcal{D})$, $\mathbf{H}(div, \mathcal{D})$ and $L^2(\mathcal{D})$) with a type of element (node, edge, facet and volume).

Although only the approximation of $\mathbf{H}(\mathbf{rot}, \mathcal{D})$ and $H^1(\mathcal{D})$ is necessary for the discretisation of the weak formulations, for the sake of completeness we present the approximation of the four Hilbert spaces $H^1(\mathcal{D})$, $\mathbf{H}(\mathbf{rot}, \mathcal{D})$, $\mathbf{H}(div, \mathcal{D})$ and $L^2(\mathcal{D})$ by the respective Whitney spaces $W^0(\mathcal{D}_h)$, $W^1(\mathcal{D}_h)$, $W^2(\mathcal{D}_h)$ and $W^3(\mathcal{D}_h)$, where \mathcal{D} is a topologically trivial domain, simply connected and without cavity, and \mathcal{D}_h is a mesh based on this domain.

9.1.1 Approximation of $H^1(\mathcal{D})$

Let n_0 be the number of nodes in mesh \mathcal{D}_h . The lowest-order Whitney space to approach $H^1(\mathcal{D})$ on \mathcal{D}_h is:

$$W^0(\mathcal{D}_h) = Vect(w_1^0(\mathbf{x}), w_2^0(\mathbf{x}), \dots, w_{n_0}^0(\mathbf{x})) \subset H^1(\mathcal{D}) \quad (9.1)$$

where the basic functions $(w_i^0(\mathbf{x}))_{i=1, \dots, n_0}$ at values in \mathbb{R} verify the following four properties:

1. denoting \mathbf{x}_j the coordinates of the j-th node, we have:

$$w_i^0(\mathbf{x}_j) = \delta_i^j, \forall (i, j) \in \{1, \dots, n_0\}^2 \quad (9.2)$$

with δ_i^j the Kronecker symbol set to 1 if $i = j$ and 0 otherwise. Thus, $w_i^0(\mathbf{x})$ is associated with node i which is why we talk about **nodal functions**.

2. $w_i^0(\mathbf{x})$, $i = (1, \dots, n_0)$ is **continue** on \mathcal{D} , and also belongs to $H^1(\mathcal{D})$.
3. the set of functions $(w_i^0(\mathbf{x}))_{i=1, \dots, n_0}$ is a **partition of unity** on \mathcal{D} :

$$\sum_{i=1}^{n_0} w_i^0(\mathbf{x}) = 1 \quad (9.3)$$

4. the i -th nodal function w_i^0 is identically zero on element K_j if this does not contain node i .

Thus, any *scalar field* $s(\mathbf{x})$ belonging to $H^1(\mathcal{D})$ will be approximated in $\mathbf{W}^0(\mathcal{D}_h)$ by:

$$s_h(\mathbf{x}) = \sum_{i=1}^{n_0} s_i w_i^0(\mathbf{x}) \quad (9.4)$$

By evaluating this expression at the mesh nodes \mathbf{x}_j and using the first property, we find that s_i corresponds to the point value at node i . Finally, it should be noted that the quantity $s_h(\mathbf{x})$ is preserved when passing from one element to another.

Remark 9.1.1 *On a triangular mesh in 2D or tetrahedral in 3D, functions $w_i^0(x, y, z)$ are Lagrange polynomials of not more than 1st order on each element. On more complex elements, there is a generalisation of this type of polynomial.*

9.1.2 Discrete approximation of $H(\mathbf{rot}, \mathcal{D})$

Let n_1 be the number of edges in mesh \mathcal{D}_h . We define the Whitney space to approach $H(\mathbf{rot}, \mathcal{D})$ on \mathcal{D}_h by:

$$\mathbf{W}^1(\mathcal{D}_h) = \text{Vec}(\mathbf{w}_1^1(\mathbf{x}), \mathbf{w}_2^1(\mathbf{x}), \dots, \mathbf{w}_{n_1}^1(\mathbf{x})) \subset H(\mathbf{rot}, \mathcal{D}) \quad (9.5)$$

where $\mathbf{w}_i^1(\mathbf{x})$ is a vector function with values in \mathbb{R}^3 , associated with the edge of index i . By analogy with property 1 for the nodal functions, they verify:

$$\int_{a_j} \mathbf{w}_i^1(\mathbf{x}) \cdot d\mathbf{l} = \delta_i^j, \forall (i, j) \in \{1, \dots, n_1\}^2 \quad (9.6)$$

where the preceding integral designates the circulation of $\mathbf{w}_i^1(\mathbf{x})$ associated with edge a_j .

Due to the properties of Whitney spaces, their expression is defined directly from that of the nodal functions. Hence, the function associated with edge i , orientated from node u to node v is:

$$\mathbf{w}_i^1 = w_v^0 \mathbf{grad} \left(\sum_{t \in \mathcal{N}(v, \bar{u})} w_t \right) - w_u^0 \left(\sum_{t \in \mathcal{N}(u, \bar{v})} w_t \right) \quad (9.7)$$

where $\mathcal{N}(u, \bar{v})$ designates the nodes on facets containing node u but not node v .

Remark 9.1.2 *For example, for the cubic element represented in Figure 9.1, $\mathcal{N}(1, \bar{2})$ contains nodes $\{1, 4, 5, 8\}$. Only facet '1485' contains node 1 and does not contain node 2.*

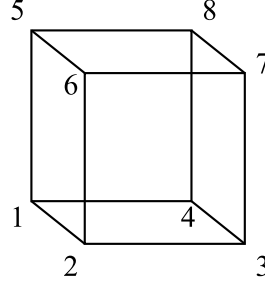


Figure 9.1: Cubic element

Thus, any *vector field* $\mathbf{V}(\mathbf{x})$ belonging to $H(\mathbf{rot}, \mathcal{D})$ will be approximated in $\mathbf{W}^1(\mathcal{D}_h)$ by:

$$\mathbf{V}_h(\mathbf{x}) = \sum_{i=1}^{n_1} V_i \mathbf{w}_i^1(\mathbf{x}) \quad (9.8)$$

where, as with the nodal functions, V_i corresponds to the circulation of \mathbf{V} on edge i .

Finally, an interesting property of this approximation is that it preserves the **continuity of the tangential trace when passing from one element to another**.

9.1.3 Discrete approximation of $H(\mathbf{div}, \mathcal{D})$

Let n_2 be the number of facets in mesh \mathcal{D}_h . We define the Whitney space to approach $H(\mathbf{div}, \mathcal{D})$ on \mathcal{D}_h by:

$$\mathbf{W}^2(\mathcal{D}_h) = \text{Vec}(\mathbf{w}_1^2(\mathbf{x}), \mathbf{w}_2^2(\mathbf{x}), \dots, \mathbf{w}_{n_2}^2(\mathbf{x})) \subset H(\mathbf{div}, \mathcal{D}) \quad (9.9)$$

where $\mathbf{w}_i^2(\mathbf{x})$ is a vector function with values in \mathbb{R}^3 , associated with the i -th facet. As before, we have:

$$\int_{f_j} \mathbf{w}_i^2(\mathbf{x}) \cdot \mathbf{d}s = \delta_{ij}^j, \quad \forall (i, j) \in \{1, \dots, n_2\}^2 \quad (9.10)$$

where the previous integral designates the flux of $\mathbf{w}_i^2(\mathbf{x})$ through facet f_j .

As with the edges, the *facet functions* are defined from the nodal functions. For conventional elements with three or four nodes per face, they are written:

$$\mathbf{w}_i^2 = a \sum_{r \in \mathcal{N}(i)} w_r^0 \left(\mathbf{grad} \sum_{t \in \mathcal{N}(r, \overline{r+1})} w_t^0 \right) \times \left(\mathbf{grad} \sum_{t \in \mathcal{N}(r, \overline{r-1})} w_t^0 \right) \quad (9.11)$$

Here, $\mathcal{N}(i)$ designates the ordered list of nodes on facet i , and a is a constant which is 2 on facets with three nodes and 1 on facets with four. Finally, the cyclic index $r+1$ in $\mathcal{N}(r, \overline{r+1})$ actually corresponds to the node at the node following r in the list $\mathcal{N}(i)$.

Remark 9.1.3 Using the example of the cubic element (see Figure 9.1), and if we wish to calculate the function associated with facet ' i ' made up of nodes '1458', $\mathcal{N}(i)$ is the ordered list $\{1, 4, 5, 8\}$ in the previous expression, it is thus a question of summing the belonging to ' $\mathcal{N}(r, \overline{r+1})$ ' and ' $\mathcal{N}(r, \overline{r-1})$ '. In practice, this means calculating $\mathcal{N}(1, \overline{4})$, $\mathcal{N}(4, \overline{5})$, $\mathcal{N}(5, \overline{8})$ and $\mathcal{N}(8, \overline{1})$ for the term ' $\mathcal{N}(r, \overline{r+1})$ ', and $\mathcal{N}(1, \overline{8})$, $\mathcal{N}(8, \overline{5})$, $\mathcal{N}(5, \overline{4})$ and $\mathcal{N}(4, \overline{1})$ for ' $\mathcal{N}(r, \overline{r-1})$ '.

Thus, any vector field $\mathbf{V}(\mathbf{x})$ belonging to $H(\mathbf{div}, \mathcal{D})$ will be approximated in $\mathbf{W}^2(\mathcal{D}_h)$ by:

$$\mathbf{V}_h(\mathbf{x}) = \sum_{i=1}^{n_2} V_i \mathbf{w}_i^2(\mathbf{x}) \quad (9.12)$$

V_i corresponds to the flux of \mathbf{V} on facet i . Here it is the **normal trace** of \mathbf{V}_h that is **preserved through the interfaces**.

9.1.4 Discrete approximation of $L^2(\mathcal{D})$

Let n_3 be the number of volume elements in mesh \mathcal{D}_h . We define the Whitney space to approach $L^2(\mathcal{D})$ on \mathcal{D}_h by:

$$\mathbf{W}^3(\mathcal{D}_h) = \text{Vec}(w_1^3(\mathbf{x}), w_2^3(\mathbf{x}), \dots, w_{n_3}^3(\mathbf{x})) \subset L^2(\mathcal{D}) \quad (9.13)$$

where $w_i^3(\mathbf{x})$ is a scalar function with values in \mathbb{R}^+ associated with element i .

By analogy with the previous calculations, they verify:

$$\int_{e_j} w_i^3(\mathbf{x}) \cdot \mathbf{d}v = \delta_{ij}^j, \quad \forall (i, j) \in \{1, \dots, n_3\}^2 \quad (9.14)$$

Here, the scalar function is integrated on element e_j .

In reality, the volume functions are constant on the element with which they are associated, and zero otherwise. For element e_i we thus have:

$$w_i^3(\mathbf{x}) = \frac{1}{\text{vol}(e_i)} \quad (9.15)$$

where $\text{vol}(e_i)$ designates the volume of element e_i .

Finally, any scalar field $s(\mathbf{x})$ in $L^2(\mathcal{D})$ will be approximated in $\mathbf{W}^3(\mathcal{D}_h)$ by a constant scalar function by parts:

$$s_h(\mathbf{x}) = \sum_{i=1}^{n_3} s_i w_i^3(\mathbf{x}) \quad (9.16)$$

where s_i corresponds, according to the previous property, to the volume of element e_i .

9.1.5 Taking account of *ad hoc* boundary conditions

The lowest-order Whitney spaces thus provide a geometric definition of the approximating spaces of: $H^1(\mathcal{D})$, $H(\mathbf{rot}, \mathcal{D})$, $H(\text{div}, \mathcal{D})$ and $L^2(\mathcal{D})$. Similarly, their sub-spaces with *ad hoc* boundary conditions on a boundary Σ are also geometrically and simply defined.

For example, $H_{0,\Sigma}^1(\mathcal{D})$ contains the functions of $H^1(\mathcal{D})$ for which the trace cancels out on Σ . The discrete space to approach it, $\mathbf{W}_{0,\Sigma}^0(\mathbf{D}_h)$, is obtained directly from $\mathbf{W}^0(\mathbf{D}_h)$ by removing the functions associated with the nodes included in Σ . Thus, by denoting \mathcal{D}_Σ the restriction of \mathbf{D}_h on the edge of Σ , we have:

$$\mathbf{W}_{0,\Sigma}^0(\mathbf{D}_h) = \mathbf{W}^0(\mathbf{D}_h) \setminus \mathbf{W}^0(\mathcal{D}_\Sigma) \quad (9.17)$$

with the conformity property conserved:

$$\mathbf{W}_{0,\Sigma}^0(\mathbf{D}_h) \subset H_{0,\Sigma}^1(\mathcal{D}) \quad (9.18)$$

Similarly, we can define $\mathbf{W}_{1,\Sigma}^1(\mathbf{D}_h)$, the space discretising $H_{0,\Sigma}(\mathbf{rot}, \mathcal{D})$, by removing from $\mathbf{W}^1(\mathbf{D}_h)$ the functions related to the edges on Σ . Hence, the sub-space of $\mathbf{W}^1(\mathbf{D}_h)$ with *ad hoc* boundary conditions on Σ is written:

$$\mathbf{W}_{0,\Sigma}^1(\mathbf{D}_h) = \mathbf{W}^1(\mathbf{D}_h) \setminus \mathbf{W}^1(\mathcal{D}_\Sigma) \quad (9.19)$$

with:

$$\mathbf{W}_{0,\Sigma}^1(\mathbf{D}_h) \subset H_{0,\Sigma}(\mathbf{rot}, \mathcal{D}) \quad (9.20)$$

In general, we can define the Whitney space with *ad hoc* boundary conditions on a boundary Σ by removing functions related to elements (nodes, edges and facets) belonging to Σ :

$$\mathbf{W}_{0,\Sigma}^k(\mathbf{D}_h) = \mathbf{W}^k(\mathbf{D}_h) \setminus \mathbf{W}^k(\mathcal{D}_\Sigma), \quad k \in \{0, 1, 2\} \quad (9.21)$$

9.2 Electrokinetic problem

9.2.1 Formulation φ with imposed voltage

The weak form of the equation is:

$$\int_{\mathcal{D}_c} \sigma \mathbf{grad} \varphi' \cdot \mathbf{grad} \varphi \, d\mathcal{D}_c + \int_{\Gamma} \varphi' (\sigma \mathbf{grad} \varphi) \cdot \mathbf{n} \, d\Gamma = - \int_{\mathcal{D}_c} \sigma \mathbf{grad} \varphi' \cdot \mathbf{grad} \alpha V \, d\mathcal{D}_c \quad (5.77)$$

Potential φ belongs to the nodal element space $\mathcal{W}_{\Gamma_b}^0$:

$$\varphi = \sum_{n \in \mathcal{N}_h} \varphi_n w_n^0 \quad (9.22)$$

As a result, the starting equation becomes:

$$\begin{aligned} \sum_{n \in \mathcal{N}_h} \varphi_n \int_{\mathcal{D}_c} \sigma \mathbf{grad} \varphi' \cdot \mathbf{grad} w_n^0 \, d\mathcal{D}_c \\ + \sum_{n \in \mathcal{N}_h} \varphi_n \int_{\Gamma} \varphi' (\sigma \mathbf{grad} w_n^0) \cdot \mathbf{n} \, d\Gamma \\ = - \int_{\mathcal{D}_c} \sigma \mathbf{grad} \varphi' \cdot \mathbf{grad} \alpha V \, d\mathcal{D}_c \end{aligned} \quad (9.23)$$

It is recalled that $\Gamma = \Gamma_h \cup \Gamma_b$. By its definition in $\mathcal{W}_{\Gamma_b}^0$, potential φ is zero on Γ_b . We thus naturally impose $\mathbf{E} \times \mathbf{n} = 0$ on Γ_b in the strong sense.

In addition, by eliminating the calculation of the surface integral on Γ_h , we impose $\mathbf{J} \cdot \mathbf{n} = 0$ in the weak sense. The integral form is thus written:

$$\sum_{n \in \mathcal{N}_h} \varphi_n \int_{\mathcal{D}_c} \sigma \mathbf{grad} \varphi' \cdot \mathbf{grad} w_n^0 \, d\mathcal{D}_c = - \int_{\mathcal{D}_c} \sigma \mathbf{grad} \varphi' \cdot \mathbf{grad} \alpha V \, d\mathcal{D}_c \quad (9.24)$$

For the test function, we thus take:

$$\varphi' = w_i^0$$

Thus equation 5.77 in its integral form becomes:

$$\forall w_i^0 \in \mathcal{W}_{\Gamma_b}^0 \quad \sum_{n \in \mathcal{N}_h} \varphi_n \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 \, d\mathcal{D}_c = - \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} \alpha V \, d\mathcal{D}_c \quad (9.25)$$

9.2.2 Formulation φ with imposed current

To impose the current with the scalar potential formulation, we saw that it was necessary to express β and \mathbf{J} as a function of α and the scalar electric potential ϕ_I . The scalar electric potential formulation with an imposed current is written:

$$\begin{aligned} \operatorname{div} \sigma \mathbf{grad} \varphi + \operatorname{div} \sigma \mathbf{grad} \alpha V &= 0 \\ \int_{\mathcal{D}_c} \mathbf{grad} \alpha \cdot \sigma \mathbf{grad} (\varphi + \alpha V) d\mathcal{D}_c &= I \end{aligned} \quad (3.28)$$

The voltage thus becomes an unknown when the current is imposed. Thus equation 3.28 in its integral form becomes:

$$\begin{aligned} \forall w_i^0 \in \mathcal{W}_{\Gamma_b}^0 \quad \sum_{n \in \mathcal{N}_h} \varphi_n \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 d\mathcal{D}_c + \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} \alpha V d\mathcal{D}_c &= 0 \\ \sum_{n \in \mathcal{N}_h} \varphi_n \int_{\mathcal{D}_c} \mathbf{grad} \alpha \cdot \sigma \mathbf{grad} (w_n^0 + \alpha V) d\mathcal{D}_c &= I \end{aligned} \quad (9.26)$$

9.2.3 Formulation \mathbf{T}

The weak form obtained is:

$$\int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathcal{U} \cdot \mathbf{rot} \mathbf{T} d\mathcal{D} = - \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathcal{U} \cdot \mathbf{rot} \mathbf{H}_s d\mathcal{D} \quad (5.81)$$

with $\mathbf{H}_s = \sum_{a \in \mathcal{A}} \mathbf{w}_a h_{a,s}$ where $h_{a,s}$ is the circulation of \mathbf{H}_s calculated on the edges of the mesh using the tree technique.

Potential \mathbf{T} is sought in $\mathcal{W}_{\Gamma_h}^1$:

$$\mathbf{T} = \sum_{a \in \mathcal{A}_h} T_a \mathbf{w}_a^1 \quad (9.27)$$

For the test function, we take:

$$\mathcal{U} = \mathbf{w}_i^1$$

Equation 5.81 in its integral form becomes:

$$\forall \mathbf{w}_i^1 \in \mathcal{W}_{\Gamma_h}^1 \quad \sum_{a \in \mathcal{A}_h} T_a \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} = - \sum_{a \in \mathcal{A}} h_{a,s} \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \quad (9.28)$$

To ensure a unique solution, it is necessary to impose a gauge condition. However, if the conjugated gradient method is used to resolve the system of equations, the problem is automatically “gauged”, as in the case of the vector magnetic potential formulation [Ren 1996][Ren 1996b].

9.3 Magnetostatic problem

9.3.1 Projection in space only

9.3.1.1 Formulation A

The weak form of the equation is:

$$\int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{A}' \cdot \mathbf{rot} \mathbf{A} \, d\mathcal{D} - \int_{\Gamma} \mathbf{A}' \cdot \left(\mathbf{n} \times \frac{1}{\mu} \mathbf{rot} \mathbf{A} \right) d\Gamma = \int_{\mathcal{D}} \mathbf{A}' \cdot \mathbf{J}_s \, d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{rot} \mathbf{A}' \, d\mathcal{D} \quad (5.61)$$

The vector magnetic potential \mathbf{A} belongs to the edge element space. Its discrete form is thus written:

$$\mathbf{A} = \sum_{a \in \mathcal{A}} \mathbf{w}_a^1 a_a \quad \mathbf{A} \in \mathcal{W}_{\Gamma_b}^1 \quad (9.29)$$

where a_a is the circulation of the vector potential \mathbf{A} on edge 'a'.

The integral form of the formulation to be solved is therefore (see equation 5.61) taking as its test function $\mathbf{w}_i^1 \in \mathcal{W}_{\Gamma_b}^1$:

$$\int_{\mathcal{D}} \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{A} \, d\mathcal{D} - \int_{\Gamma} \mathbf{w}_i^1 \cdot \left(\mathbf{n} \times \frac{1}{\mu} \mathbf{rot} \mathbf{A} \right) d\Gamma = \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot \mathbf{J}_s \, d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{B}_r \, d\mathcal{D} \quad (9.30)$$

with:

$$J_s^d = \sum_{f \in \mathcal{F}} \mathbf{w}_f^2 j_s^d$$

The current density is discretised in the facet element space.

The integral on Γ breaks down as before into two terms. The first on Γ_b is eliminated naturally, which leads to $\mathbf{B} \cdot \mathbf{n} = 0$ imposed in the strong sense. By eliminating the second (on Γ_h) we impose $\mathbf{H} \times \mathbf{n} = \mathbf{0}$ in the weak sense. Equation 9.30 thus becomes:

$$\int_{\mathcal{D}} \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{A} \, d\mathcal{D} = \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot \mathbf{J}_s^d \, d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{B}_r \, d\mathcal{D} \quad (9.31)$$

This leads to the final integral formulation:

$$\forall \mathbf{w}_i^1 \in \mathcal{W}_{\Gamma_b}^1 \quad \sum_{a \in \mathcal{A}} a_a \int_{\mathcal{D}} \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 \, d\mathcal{D} = \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}_i^1 \, d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{B}_r \, d\mathcal{D} \quad (9.32)$$

In this case, resolution by the conjugated gradient method leads to an automatically gauged system [Ren 1996b]. Under these conditions, the use of a gauge of type $\mathbf{A} \cdot \mathbf{w}$ is no longer necessary.

9.3.1.2 Formulation Ω

The weak form of the formulation is:

$$\int_{\mathcal{D}} \mu (\mathbf{grad} \Omega' \cdot \mathbf{grad} \Omega - \mathbf{grad} \Omega' \cdot \mathbf{H}_s) \, d\mathcal{D} + \int_{\Gamma} \Omega' (\mu \mathbf{grad} \Omega) \, d\gamma = - \int_{\mathcal{D}} \Omega' \operatorname{div} \mathbf{B}_r \, d\mathcal{D} \quad (5.67)$$

The scalar potential Ω belongs to the nodal element space \mathcal{W}_h^0 , hence it can be written as follows:

$$\Omega = \sum_{n \in \mathcal{N}_h} w_n^0 \Omega_n \quad (9.33)$$

Hence we take the test function $w_i^0 \in \mathcal{W}_h^0$, so:

$$\int_{\mathcal{D}} \mu (\mathbf{grad} w_i^0 \cdot \mathbf{grad} \Omega - \mathbf{grad} w_i^0 \cdot \mathbf{H}_s) d\mathcal{D} + \int_{\Gamma} w_i^0 (\mu \mathbf{grad} \Omega) d\gamma = - \int_{\mathcal{D}} w_i^0 \operatorname{div} \mathbf{B}_r d\mathcal{D} \quad (9.34)$$

where, H_s which represents the source field, is calculated from J_0^d and broken down in space \mathcal{W}_h^1 .

For the surface integral Γ , on Γ_h we have $w_i^0 = 0$, which imposes $\mathbf{H} \times \mathbf{n} = 0$ in the strong sense. However, by eliminating the integral on Γ_b , the condition $\mathbf{B} \cdot \mathbf{n} = 0$ is imposed in the weak sense. Under these conditions, the preceding equation is written:

$$\forall w_i^0 \in \mathcal{W}_{\Gamma_h}^0 \quad \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 d\mathcal{D} = \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{H}_s d\mathcal{D} - \int_{\mathcal{D}} w_i^0 \operatorname{div} \mathbf{B}_r d\mathcal{D} \quad (9.35)$$

9.3.2 Projection in space and time

This case is not detailed here as it is dealt with as a special case for magnetodynamic problems in paragraph 9.4.2.

9.4 Magnetodynamic problem

9.4.1 Projection in space only

9.4.1.1 Formulation $\mathbf{A} - \varphi$

The weak form of this formulation is given by the following expressions:

$$\int_{\mathcal{D}} \left[\frac{1}{\mu} \operatorname{rot} \mathbf{A}' \cdot \operatorname{rot} \mathbf{A} + \sigma \mathbf{A}' \cdot \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \right] d\mathcal{D} = \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{A}' d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \operatorname{rot} \mathbf{A}' d\mathcal{D} \quad (5.29)$$

$$\int_{\mathcal{D}} \sigma \mathbf{grad} \varphi' \cdot \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) d\mathcal{D} = 0 \quad (9.36)$$

This formulation has two unknowns: the vector magnetic potential \mathbf{A} and scalar electric potential φ defined in the Whitney element space as follows:

$$\varphi = \sum_{n \in \mathcal{N}_h} w_n^0 \varphi_n \quad \varphi \in \mathcal{W}_{\Gamma_b}^0 \quad (9.37)$$

$$\mathbf{A} = \sum_{a \in \mathcal{A}_h} \mathbf{w}_a^1 a_a \quad \mathbf{A} \in \mathcal{W}_{\Gamma_b}^1 \quad (9.38)$$

We thus take:

$$\mathbf{A}' = \mathbf{w}_i^1$$

and

$$\varphi' = w_i^0$$

The system of equations 5.29 is thus written:

$$\int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{A} + \sigma \mathbf{w}_i^1 \cdot \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \right] d\mathcal{D} = \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}_i^1 d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{rot} \mathbf{w}_i^1 d\mathcal{D} \quad (9.39)$$

$$\int_{\mathcal{D}} \sigma \mathbf{grad} w_i^0 \cdot \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) d\mathcal{D} = 0 \quad (9.40)$$

The first equation corresponds to Ampère's circuital law and the second to the conservation of the current density flux.

As already noted above, the surface integrals disappear. This amounts to strongly imposing boundary conditions on Γ_b ($\mathbf{E} \times \mathbf{n} = 0$ and $\mathbf{B} \cdot \mathbf{n} = 0$) and weakly on Γ_h ($\mathbf{H} \times \mathbf{n} = 0$ and $\mathbf{J} \cdot \mathbf{n} = 0$).

Unlike the expressions in the preceding paragraphs, the weak formulation here shows time differentiations. They are dealt with in paragraph 9.5.

9.4.1.2 Formulation T-Ω

The weak formulation is written:

$$\int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \cdot \mathbf{rot} \mathbf{T}' + \mathbf{T}' \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{T}' d\gamma = \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} \mathbf{T}' + \mathbf{T}' \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \quad (5.49)$$

$$\int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{grad} \Omega' d\gamma = \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \quad (5.50)$$

For this formulation, based on the reasoning above, the surface integrals on Γ disappear. The boundary conditions are then imposed in the weak sense on Γ_b ($\mathbf{E} \times \mathbf{n} = 0$ and $\mathbf{B} \cdot \mathbf{n} = 0$) and in the strong sense on Γ_h ($\mathbf{H} \times \mathbf{n} = 0$ and $\mathbf{J} \cdot \mathbf{n} = 0$).

This formulation also has two unknowns: the vector electric potential \mathbf{T} and scalar magnetic potential Ω defined in the Whitney element space as follows:

$$\Omega = \sum_{n \in \mathcal{N}_h} w_n^0 \Omega_n \quad \Omega \in \mathcal{W}_h^0 \quad (9.41)$$

$$\mathbf{T} = \sum_{a \in \mathcal{A}_h} \mathbf{w}_a^1 t_a \quad \mathbf{T} \in \mathcal{W}_h^1 \quad (9.42)$$

For the test function, we take:

$$\mathbf{T}' = \mathbf{w}_i^1$$

and:

$$\Omega' = w_i^0$$

Equations 5.49 and 5.50 thus become:

$$\begin{aligned} \sum_{a \in \mathcal{A}_h} t_a \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_a^1 \cdot \mathbf{rot} \mathbf{w}_i^1 d\mathcal{D} \\ + \sum_{a \in \mathcal{A}_h} t_a \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot \frac{\partial}{\partial t} \mu \mathbf{w}_a^1 d\mathcal{D} - \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot \frac{\partial}{\partial t} \mu \mathbf{grad} w_n^0 d\mathcal{D} = \\ \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} \mathbf{w}_i^1 + \mathbf{w}_i^1 \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \quad (9.43) \end{aligned}$$

$$\begin{aligned} \sum_{a \in \mathcal{A}_h} t_a \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot \frac{\partial}{\partial t} \mu \mathbf{w}_a^1 d\mathcal{D} - \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot \frac{\partial}{\partial t} \mu \mathbf{grad} w_n^0 d\mathcal{D} = \\ \int_{\mathcal{D}} \left[\mathbf{grad} w_i^0 \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \quad (9.44) \end{aligned}$$

By integrating this last equation in time, we obtain:

$$\begin{aligned} \sum_{a \in \mathcal{A}_h} t_a \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot \mu \mathbf{w}_a^1 d\mathcal{D} - \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot \mu \mathbf{grad} w_n^0 d\mathcal{D} = \\ \int_{\mathcal{D}} [\mathbf{grad} w_i^0 \cdot (\mu \mathbf{H}_s + \mathbf{B}_r)] d\mathcal{D} \quad (9.45) \end{aligned}$$

We can discretise fields \mathbf{H}_s and \mathbf{B}_r :

$$\mathbf{H}_s(\mathbf{x}, t) = \sum_l H_{s_l} \mathbf{w}_l^1(\mathbf{x}) \quad (9.46)$$

$$\mathbf{B}_r(\mathbf{x}, t) = \sum_l B_{r_l} (\mathbf{w}_l^2(\mathbf{x}) \times \mathbf{n}) \quad (9.47)$$

The weak form integral is written with these considerations:

$$\begin{aligned}
& \sum_{a \in \mathcal{A}_h} t_a \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_a^1 \cdot \mathbf{rot} \mathbf{w}_i^1 d\mathcal{D} \\
& + \sum_{a \in \mathcal{A}_h} t_a \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot \frac{\partial}{\partial t} \mu \mathbf{w}_a^1 d\mathcal{D} - \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot \frac{\partial}{\partial t} \mu \mathbf{grad} w_n^0 d\mathcal{D} = \\
& \sum_l H_{sl} \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_l^1 \cdot \mathbf{rot} \mathbf{w}_i^1 d\mathcal{D} + \sum_l H_{sl} \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot \frac{\partial}{\partial t} \mu \mathbf{w}_l^1 d\mathcal{D} \\
& + \sum_l B r_l \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot (\mathbf{w}_l^2 \times \mathbf{n}) d\mathcal{D} \quad (9.48)
\end{aligned}$$

$$\begin{aligned}
& \sum_{a \in \mathcal{A}_h} t_a \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot \mu \mathbf{w}_a^1 d\mathcal{D} - \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot \mu \mathbf{grad} w_n^0 d\mathcal{D} = \\
& \sum_l H_{sl} \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot \mu \mathbf{w}_l^1 d\mathcal{D} + \sum_l B r_l \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot (\mathbf{w}_l^2 \times \mathbf{n}) d\mathcal{D} \quad (9.49)
\end{aligned}$$

Unlike the expressions in the preceding paragraphs, the weak formulation here shows time differentiations. They are dealt with in paragraph 9.5.

9.4.2 Projection in space and time

One way to obtain steady state without calculating the transient state, when the power source is multi-harmonic and sinusoidal, is to use the *Harmonic Balance Method*. This is a Fourier-type spectral approach that provides a spectral representation (Fourier series) of the solution sought when the values of the system under study are periodic. When the values are not periodic, the Fourier basis is no longer suitable.

We therefore propose to develop spectral approaches in which the discretisation bases of the time dimension are suited to the properties (periodicity, regularity and continuity) of the electromagnetic values. We introduce the finite dimension space $\mathcal{C} = (\psi_i)_{i=1}^{n_t}$ containing the continuous scalar functions defined in the interval \mathcal{T} . The N^t elements of \mathcal{C} form a basis relative to the scalar product of $L_w^2(\mathcal{T})$, where w is a positive function on \mathcal{T} , i.e.:

$$\int_{\mathcal{T}} \psi_i(t) \psi_j(t) w(t) dt = \delta_{ij}; \quad 1 \leq i, j \leq n_t \quad (9.50)$$

with δ_{ij} the Kronecker product.

To simplify the notation, the weighted integral 9.50 is written:

$$\int_{\mathcal{T}_w} \psi_i \psi_j$$

We will now describe two variants of the spectral approach for the calculation of the solution sought $\mathbf{X}(t)$ in the form:

$$\mathbf{X}(t) = \sum_{i=1}^{N^t} \mathbf{Y}_i \psi_i(t) \quad (9.51)$$

where the spectral coefficients \mathbf{Y}_i are vectors containing all the spatial degrees of freedom to be determined.

For example, the vector of the unknowns of the vector potential is written:

$$\mathbf{A}(t) = \sum_{i=1}^{N^t} \mathbf{A}_i \psi_i(t) \quad (9.52)$$

where \mathbf{A}_i is the i -th spectral vector of size n_1 (total of the spatial unknowns of the vector magnetic potential).

9.4.2.1 Differentiation in the spectral domain

The interest of spectral methods in relation to transient techniques (time methods) lies in the fact that, for any order, derivatives can be easily linked to their primitives.

As a result, unknowns A_{ij}^∂ , for $i = 1$ to $i = N^t$, are linked to unknowns A_{ij} by the differentiation matrix \mathbf{D} as follows:

$$\begin{bmatrix} A_{1j}^\partial \\ \vdots \\ A_{N^t j}^\partial \end{bmatrix} = \mathbf{D} \begin{bmatrix} A_{1j} \\ \vdots \\ A_{N^t j} \end{bmatrix} \quad (9.53)$$

Matrix \mathbf{D} is a square matrix of size $N^t \times N^t$ that depends on the adopted discretisation basis \mathcal{C} . It is explained in annex **N** for the Fourier basis and the Legendre and Chebyshev polynomial bases.

Hence, we link vector \mathbf{X}^{A^∂} and \mathbf{X}^A by:

$$\begin{pmatrix} \begin{bmatrix} A_{11}^\partial \\ \vdots \\ A_{1n_1}^\partial \\ \vdots \\ A_{N^t 1}^\partial \\ \vdots \\ A_{N^t n_1}^\partial \end{bmatrix} \end{pmatrix} = \begin{pmatrix} \begin{bmatrix} D_{11} & & 0 \\ & \ddots & \\ 0 & & D_{11} \end{bmatrix} & \cdots & \begin{bmatrix} D_{1N^t} & & 0 \\ & \ddots & \\ 0 & & D_{1N^t} \end{bmatrix} \\ \vdots & & \vdots \\ \begin{bmatrix} D_{N^t 1} & & 0 \\ & \ddots & \\ 0 & & D_{N^t 1} \end{bmatrix} & \cdots & \begin{bmatrix} D_{N^t N^t} & & 0 \\ & \ddots & \\ 0 & & D_{N^t N^t} \end{bmatrix} \end{pmatrix} \begin{pmatrix} \begin{bmatrix} A_{11} \\ \vdots \\ A_{1n_1} \\ \vdots \\ A_{N^t 1} \\ \vdots \\ A_{N^t n_1} \end{bmatrix} \end{pmatrix} \quad (9.54)$$

In other words, by denoting \mathbf{I}^{n_1} the identity matrix of size $n_1 \times n_1$, we write:

$$\mathbf{X}^{A^\partial} = (\mathbf{D} \otimes \mathbf{I}^{n_1}) \mathbf{X}^A \quad (9.55)$$

where \otimes is the Kronecker product described in annex **O**.

9.4.2.2 Formulation \mathbf{A} - φ

The weak form of the equation obtained above is:

$$\begin{aligned} \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\mu^{-1} \mathbf{rot} \mathbf{A} \cdot \mathbf{rot} \mathbf{A}' + \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathbf{A}' \right] d\mathcal{D} = \\ \int_{\mathcal{T}} \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{A}' d\mathcal{D} + \int_{\mathcal{T}} \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{rot} \mathbf{A}' d\mathcal{D} + \int_{\mathcal{T}} \int_{\Gamma} (\mathbf{H}^\Gamma \times \mathbf{n}) \cdot \mathbf{A}' d\gamma \\ \int_{\mathcal{T}} \int_{\mathcal{D}} \sigma \left(\frac{\mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathbf{grad} \varphi' d\mathcal{D} = 0 \end{aligned} \quad (5.40)$$

Remark 9.4.1 In multi-harmonic code_Carmel, before applying the Galerkin method, we assume that the non-linear constitutive relation can be written in the general form:

$$\mathbf{H} = \mathcal{K}(\mathbf{B}) = \mathcal{K}(\mathbf{rot} \mathbf{A}) \quad (9.56)$$

The non-linear relation is then written:

$$H(\mathbf{x}, t) = \bar{\nu}(\mathbf{x}) B(\mathbf{x}, t) + \mathcal{K}(\mathbf{x}, t) \quad (9.57)$$

Given the previous remark expressing the relation between \mathbf{H} and \mathbf{B} , the weak form of equation 5.40 becomes:

$$\begin{aligned} \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\mathcal{K}(\mathbf{rot} \mathbf{A}) \cdot \mathbf{rot} \mathbf{A}' + \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathbf{A}' \right] d\mathcal{D} = \\ \int_{\mathcal{T}} \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{A}' d\mathcal{D} + \int_{\mathcal{T}} \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{rot} \mathbf{A}' d\mathcal{D} + \int_{\mathcal{T}} \int_{\Gamma} (\mathbf{H}^{\Gamma} \times \mathbf{n}) \cdot \mathbf{A}' d\gamma \\ \int_{\mathcal{T}} \int_{\mathcal{D}} \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \cdot \mathbf{grad} \varphi' d\mathcal{D} = 0 \end{aligned} \quad (9.58)$$

One way to solve system 9.58 is to apply the Galerkin method. The discrete weak form of the magnetodynamic problem can be obtained by applying the weighted residuals and the Galerkin method to system of equations 9.58 or by discretising the dependent degrees of freedom of time on \mathcal{C} , i.e. by writing the values \mathbf{A} , φ and \mathbf{J}_0 as linear combinations of space functions (belonging to W_1 , W_1 or W_2) and time functions (the indices of $\mathbf{A}(\mathbf{x}, t)$ and $\varphi(\mathbf{x}, t)$ such that we find the notation chosen previously for \mathbf{X}^A and \mathbf{X}^{φ}):

$$\mathbf{A}(\mathbf{x}, t) = \sum_{s,i} A_{s,i} \mathbf{w}_i^1(\mathbf{x}) \psi_s(t) \quad (9.59)$$

$$\partial_t \mathbf{A}(\mathbf{x}, t) = \sum_{s,i} A_{s,i}^{\partial} \mathbf{w}_i^1(\mathbf{x}) \psi_s(t) \quad (9.60)$$

$$\varphi(\mathbf{x}, t) = \sum_{s,j} \varphi_{s,j} w_j^0(\mathbf{x}) \psi_s(t) \quad (9.61)$$

$$\mathbf{J}_0(\mathbf{x}, t) = \sum_{s,l} J_{s,l} \mathbf{w}_l^2(\mathbf{x}) \psi_s(t) \quad (9.62)$$

$$\mathbf{B}_r(\mathbf{x}, t) = \sum_{s,l} B_{s,l} \mathbf{w}_l^2(\mathbf{x}) \psi_s(t) \quad (9.63)$$

$$\mathbf{H}^{\Gamma}(\mathbf{x}, t) = \sum_{s,l} H_{s,l}^{\Gamma} (\mathbf{w}_l^1(\mathbf{x}) \times \mathbf{n}) \psi_s(t) \quad (9.64)$$

By introducing the preceding equations into 9.58 the discrete weak form of the problem is written:

$$\begin{aligned}
& \int_{\mathcal{T}} \int_{\mathcal{D}} \mathcal{K}(\mathbf{rot} \mathbf{A}) \cdot \mathbf{rot} \mathbf{u} + \sum_s \int_{\mathcal{T}} \psi_s(t) \left[\sum_i A_{si}^{\partial} \int_{\mathcal{D}_c} \sigma \mathbf{w}_i^1 \cdot \mathbf{u} + \sum_j \varphi_{sj} \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_j^0 \cdot \mathbf{u} \right] = \\
& \sum_s \int_{\mathcal{T}} \psi_s(t) \left[\sum_l J_{sl} \int_{\mathcal{D}} \mathbf{w}_l^2 \cdot \mathbf{u} + \sum_l \frac{1}{\mu} B_{sl} \int_{\mathcal{D}} \mathbf{w}_l^2 \cdot \mathbf{rot} \mathbf{u} + \sum_l H_{sl}^{\Gamma} \int_{\Gamma_H} (\mathbf{w}_l^1 \times \mathbf{n}) \cdot \mathbf{u} \right] \\
& \sum_s \int_{\mathcal{T}} \psi_s(t) \left[\sum_i A_{si}^{\partial} \int_{\mathcal{D}_c} \sigma \mathbf{w}_i^1 \cdot \mathbf{grad} v + \sum_j \varphi_{sj} \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_j^0 \cdot \mathbf{grad} v \right] = \\
& \sum_s \int_{\mathcal{T}} \psi_s(t) \left[\sum_l J_{sl}^{\Gamma} \int_{\Gamma_H} (\mathbf{w}_l^2 \times \mathbf{n}) v \right]
\end{aligned} \tag{9.65}$$

We apply the Galerkin method, with the test function:

$$\mathbf{u} = \mathbf{w}_f^1 \psi_p$$

et

$$v = w_g^0 \psi_p$$

to finally obtain the following system of equations:

$$\left\{ \begin{aligned} & \int_{\mathcal{T}_w} \psi_p \int_{\mathcal{D}} \mathcal{K}(\mathbf{rot} \mathbf{A}) \cdot \mathbf{rot} \mathbf{w}_f^1 + \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_i A_{si}^{\partial} \int_{\mathcal{D}_c} \sigma \mathbf{w}_i^1 \cdot \mathbf{w}_f^1 + \sum_j \varphi_{sj} \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_j^0 \cdot \mathbf{w}_f^1 \right] \\ & = \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_l J_{sl}^0 \int_{\mathcal{D}} \mathbf{w}_l^2 \cdot \mathbf{w}_f^1 + \sum_l \frac{1}{\mu} B_{sl} \int_{\mathcal{D}} \mathbf{w}_l^2 \cdot \mathbf{rot} \mathbf{w}_f^1 + \sum_l H_{sl}^{\Gamma} \int_{\Gamma_H} (\mathbf{w}_l^1 \times \mathbf{n}) \cdot \mathbf{w}_f^1 \right] \\ & \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_i A_{si}^{\partial} \int_{\mathcal{D}_c} \sigma \mathbf{w}_i^1 \cdot \mathbf{grad} w_g^0 + \sum_j \varphi_{sj} \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_j^0 \cdot \mathbf{grad} w_g^0 \right] = \\ & \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_l J_{sl}^{\Gamma} \int_{\Gamma_H} (\mathbf{w}_l^2 \times \mathbf{n}) w_g^0 \right] \end{aligned} \right. = \tag{9.66}$$

with:

$$1 \leq f \leq n_1, \quad 1 \leq g \leq n_0, \quad 1 \leq s, p \leq N_t$$

By considering the decomposition hypothesis of the non-linear magnetic constitutive relation,

$$\mathbf{H} = \nu^{pf} \mathbf{rot} \mathbf{A} + \mathcal{K}^{nl}(\mathbf{rot} \mathbf{A})$$

we obtain:

$$\left\{ \begin{aligned} & \int_{\mathcal{T}_w} \psi_p \int_{\mathcal{D}} \mathcal{K}^{nl}(\mathbf{rot} \mathbf{A}) \cdot \mathbf{rot} \mathbf{w}_f^1 + \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_i A_{si} \int_{\mathcal{D}} \nu_{pf} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_f^1 + \sum_i A_{si}^\partial \int_{\mathcal{D}_c} \sigma \mathbf{w}_i^1 \cdot \mathbf{w}_f^1 \right. \\ & + \sum_j \varphi_{sj} \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_j^0 \cdot \mathbf{w}_f^1 \left. \right] = \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_l J_{sl}^0 \int_{\mathcal{D}} \mathbf{w}_l^2 \cdot \mathbf{w}_f^1 + \sum_l \frac{1}{\mu} B_{sl} \int_{\mathcal{D}} \mathbf{w}_l^2 \cdot \mathbf{rot} \mathbf{w}_f^1 \right. \\ & + \sum_l H_{sl}^\Gamma \int_{\Gamma_H} (\mathbf{w}_l^1 \times \mathbf{n}) \cdot \mathbf{w}_f^1 \left. \right] \\ & \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_i A_{si}^\partial \int_{\mathcal{D}_c} \sigma \mathbf{w}_i^1 \cdot \mathbf{grad} w_g^0 + \sum_j \varphi_{sj} \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_j^0 \cdot \mathbf{grad} w_g^0 \right] = \\ & \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_l J_{sl}^\Gamma \int_{\Gamma_H} (\mathbf{w}_l^2 \times \mathbf{n}) w_g^0 \right] \end{aligned} \right. \quad (9.67)$$

9.4.2.3 Formulation $\mathbf{T} - \Omega$

The weak formulation is written:

$$\begin{aligned} \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \cdot \mathbf{rot} \mathbf{T}' + \mathbf{T}' \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{T}' d\gamma = \\ \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} \mathbf{T}' + \mathbf{T}' \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \quad (5.53) \end{aligned}$$

$$\begin{aligned} \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{grad} \Omega' d\gamma = \\ \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \quad (5.54) \end{aligned}$$

In this formulation, the relation between the magnetic field \mathbf{H} and the flux density \mathbf{B} is linear. As a result, the magnetic permeability μ is constant. This changes the previous expressions:

$$\begin{aligned} \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \cdot \mathbf{rot} \mathbf{T}' + \mathbf{T}' \cdot \mu \left(\frac{\partial}{\partial t} \mathbf{T} - \mathbf{grad} \frac{\partial}{\partial t} \Omega \right) \right] d\mathcal{D} - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{T}' d\gamma = \\ \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} \mathbf{T}' + \mathbf{T}' \cdot \left(\mu \frac{\partial}{\partial t} \mathbf{H}_s + \frac{\partial}{\partial t} \mathbf{B}_r \right) \right] d\mathcal{D} \quad (9.68) \end{aligned}$$

$$\begin{aligned} \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \mu \left(\frac{\partial}{\partial t} \mathbf{T} - \mathbf{grad} \frac{\partial}{\partial t} \Omega \right) \right] d\mathcal{D} - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{grad} \Omega' d\gamma = \\ \int_{\mathcal{T}} \int_{\mathcal{D}} \left[\mathbf{grad} \Omega' \cdot \left(\mu \frac{\partial}{\partial t} \mathbf{H}_s + \frac{\partial}{\partial t} \mathbf{B}_r \right) \right] d\mathcal{D} \quad (9.69) \end{aligned}$$

One way to solve the system of equations 9.68 and 9.69 is to apply the Galerkin method. The discrete weak form of the magnetodynamic problem can be obtained by applying the weighted residuals and the Galerkin method to previous system of equations or by discretising the dependent degrees of freedom of time on \mathcal{C} , i.e. by writing the values \mathbf{T} , Ω , \mathbf{H}_s and \mathbf{B}_r as linear combinations of space functions (belonging to W_1 , W_1 or W_2) and time functions (the indices of $\mathbf{T}(\mathbf{x}, t)$ and $\Omega(\mathbf{x}, t)$ such that we find the notation chosen previously for \mathbf{X}^T and \mathbf{X}^Ω):

$$\mathbf{T}(\mathbf{x}, t) = \sum_{s,i} T_{s,i} \mathbf{w}_i^1(\mathbf{x}) \psi_s(t) \quad (9.70)$$

$$\partial_t \mathbf{T}(\mathbf{x}, t) = \sum_{s,i} T_{s,i}^\partial \mathbf{w}_i^1(\mathbf{x}) \psi_s(t) \quad (9.71)$$

$$\Omega(\mathbf{x}, t) = \sum_{s,j} \Omega_{s,j} w_j^0(\mathbf{x}) \psi_s(t) \quad (9.72)$$

$$\frac{\partial}{\partial t} \Omega(\mathbf{x}, t) = \sum_{s,j} \Omega_{s,j}^\partial w_j^0(\mathbf{x}) \psi_s(t) \quad (9.73)$$

$$\mathbf{H}_s(\mathbf{x}, t) = \sum_{s,l} H_{s,l} \mathbf{w}_l^1(\mathbf{x}) \psi_s(t) \quad (9.74)$$

$$\frac{\partial}{\partial t} \mathbf{H}_s(\mathbf{x}, t) = \sum_{s,l} H_{s,l}^\partial \mathbf{w}_l^1(\mathbf{x}) \psi_s(t) \quad (9.75)$$

$$\mathbf{B}_r(\mathbf{x}, t) = \sum_{s,l} B_{r,s,l} (\mathbf{w}_l^2(\mathbf{x}) \times \mathbf{n}) \psi_s(t) \quad (9.76)$$

$$\frac{\partial}{\partial t} \mathbf{B}_r(\mathbf{x}, t) = \sum_{s,l} B_{r,s,l}^\partial (\mathbf{w}_l^2(\mathbf{x}) \times \mathbf{n}) \psi_s(t) \quad (9.77)$$

By introducing expressions 9.70 to 9.77 into 9.68 and 9.69, the discrete weak form of the problem is written:

$$\begin{aligned} & \sum_s \int_{\mathcal{T}} \psi_s(t) \sum_i T_{s,i} \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_i^1(\mathbf{x}) \cdot \mathbf{rot} \mathbf{T}' d\mathcal{D} \\ & + \sum_s \int_{\mathcal{T}} \psi_s(t) \sum_i T_{s,i}^\partial \int_{\mathcal{D}} \mu \mathbf{T}' \cdot \mathbf{w}_i^1(\mathbf{x}) d\mathcal{D} \\ & - \sum_s \int_{\mathcal{T}} \psi_s(t) \sum_j \Omega_{s,j}^\partial \int_{\mathcal{D}} \mu \mathbf{T}' \cdot \mathbf{grad} w_j^0(\mathbf{x}) d\mathcal{D} \\ & - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{T}' d\gamma = \\ & \sum_s \int_{\mathcal{T}} \psi_s(t) \sum_l H_{s,l}^\partial \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_l^1 \cdot \mathbf{rot} \mathbf{T}' d\mathcal{D} \\ & + \sum_s \int_{\mathcal{T}} \psi_s(t) \sum_l H_{s,l}^\partial \int_{\mathcal{D}} \mu \mathbf{T}' \cdot \mathbf{w}_l^1 d\mathcal{D} \\ & + \sum_s \int_{\mathcal{T}} \psi_s(t) \sum_l B_{r,s,l}^\partial \int_{\mathcal{D}} \mathbf{T}' \cdot \mathbf{w}_l^2 d\mathcal{D} \quad (9.78) \end{aligned}$$

$$\begin{aligned} & \sum_s \int_{\mathcal{T}} \psi_s(t) \sum_i T_{s,i}^\partial \int_{\mathcal{D}} \mu \mathbf{grad} \Omega' \cdot \mathbf{w}_i^1(\mathbf{x}) d\mathcal{D} \\ & - \sum_s \int_{\mathcal{T}} \psi_s(t) \sum_j \Omega_{s,j}^\partial \int_{\mathcal{D}} \mu \mathbf{grad} \Omega' \cdot \mathbf{grad} w_j^0(\mathbf{x}) d\mathcal{D} - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{grad} \Omega' d\gamma = \\ & \sum_s \int_{\mathcal{T}} \psi_s(t) \sum_l H_{s,l}^\partial \int_{\mathcal{D}} \mathbf{grad} \Omega' \cdot \mu \mathbf{w}_l^1(\mathbf{x}) d\mathcal{D} \\ & + \sum_s \int_{\mathcal{T}} \psi_s(t) \sum_l B_{r,s,l}^\partial \int_{\mathcal{D}} \mathbf{grad} \Omega' \cdot (\mathbf{w}_l^2(\mathbf{x}) \times \mathbf{n}) d\mathcal{D} \quad (9.79) \end{aligned}$$

We apply the Galerkin method, with the test function:

$$\mathbf{T}' = \mathbf{w}_f^1 \psi_p$$

and

$$\Omega' = w_g^0 \psi_p$$

to finally obtain the following system of equations:

$$\begin{aligned} & \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_i T_{s,i} \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_i^1(\mathbf{x}) \cdot \mathbf{rot} \mathbf{w}_f^1 d\mathcal{D} \\ & + \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_i T_{s,i}^\partial \int_{\mathcal{D}} \mu \mathbf{w}_f^1 \cdot \mathbf{w}_i^1(\mathbf{x}) d\mathcal{D} \\ & - \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_j \Omega_{s,j}^\partial \int_{\mathcal{D}} \mu \mathbf{w}_f^1 \cdot \mathbf{grad} w_j^0(\mathbf{x}) d\mathcal{D} \\ & - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{T}' d\gamma = \\ & \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_l H s_{s,l}^\partial \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_l^1 \cdot \mathbf{rot} \mathbf{w}_f^1 d\mathcal{D} \\ & + \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_l H s_{s,l}^\partial \int_{\mathcal{D}} \mu \mathbf{w}_f^1 \cdot \mathbf{w}_l^1 d\mathcal{D} \\ & + \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_l B r_{s,l}^\partial \int_{\mathcal{D}} \mathbf{w}_f^1 \cdot \mathbf{w}_l^2 d\mathcal{D} \quad (9.80) \end{aligned}$$

$$\begin{aligned} & \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_i T_{s,i}^\partial \int_{\mathcal{D}} \mu \mathbf{grad} w_g^0 \cdot \mathbf{w}_i^1(\mathbf{x}) d\mathcal{D} \\ & - \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_j \Omega_{s,j}^\partial \int_{\mathcal{D}} \mu \mathbf{grad} w_g^0 \cdot \mathbf{grad} w_j^0(\mathbf{x}) d\mathcal{D} - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{grad} \Omega' d\gamma = \\ & \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_l H s_{s,l}^\partial \int_{\mathcal{D}} \mathbf{grad} w_g^0 \cdot \mu \mathbf{w}_l^1(\mathbf{x}) d\mathcal{D} \\ & + \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_l B r_{s,l}^\partial \int_{\mathcal{D}} \mathbf{grad} w_g^0 \cdot (\mathbf{w}_l^2(\mathbf{x}) \times \mathbf{n}) d\mathcal{D} \quad (9.81) \end{aligned}$$

9.5 Time discretisation

9.5.1 Weak form discretisation

The backward Euler method is used in the time-based version of code_Carmel. It consists in writing, for a variable U [Dhatt, Thouzot 1984]:

$$\dot{U}_{t+\Delta t} \simeq \frac{1}{\Delta t} (U_{t+\Delta t} - U_t) \quad (9.82)$$

If we index the variable:

$$\begin{aligned} n & \rightarrow t \\ n+1 & \rightarrow t + \Delta t \end{aligned}$$

then the previous expression becomes:

$$\dot{U}_{n+1} = \frac{1}{\Delta t} (U_{n+1} - U_n) \quad (9.83)$$

9.5.2 Magnétodynamique

9.5.2.1 Formulation $\mathbf{A} - \varphi$

The expression obtained previously for the weak integral form of vector magnetic potential and scalar electric potential is:

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{w}'_a \cdot \mathbf{rot} \mathbf{A} + \sigma \mathbf{w}'_a \cdot \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) \right] d\mathcal{D} &= \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}'_a d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{w}'_a d\mathcal{D} \\ \int_{\mathcal{D}} \sigma \mathbf{grad} w'_n \left(\frac{\partial \mathbf{A}}{\partial t} + \mathbf{grad} \varphi \right) d\mathcal{D} &= 0 \end{aligned} \quad (9.39)$$

These expressions are rewritten at time $i + 1$ for the time-dependent variables:

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{w}'_a \cdot \mathbf{rot} \mathbf{A}_{(i+1)} + \sigma \mathbf{w}'_a \cdot \left(\frac{\partial \mathbf{A}}{\partial t}_{(i+1)} + \mathbf{grad} \varphi_{(i+1)} \right) \right] d\mathcal{D} &= \int_{\mathcal{D}} \mathbf{J}_{s(i+1)} \cdot \mathbf{w}'_a d\mathcal{D} \\ &+ \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{w}'_a d\mathcal{D} \\ \int_{\mathcal{D}} \sigma \mathbf{grad} w'_n \left(\frac{\partial \mathbf{A}}{\partial t}_{(i+1)} + \mathbf{grad} \varphi_{(i+1)} \right) d\mathcal{D} &= 0 \end{aligned} \quad (9.84)$$

By applying the backward Euler method, the system of equations becomes:

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{w}'_a \cdot \mathbf{rot} \mathbf{A}_{(i+1)} + \sigma \mathbf{w}'_a \cdot \left(\frac{\mathbf{A}_{(i+1)}}{\Delta t} + \mathbf{grad} \varphi_{(i+1)} \right) \right] d\mathcal{D} &= \int_{\mathcal{D}} \mathbf{J}_{s(i+1)} \cdot \mathbf{w}'_a d\mathcal{D} \\ &+ \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{w}'_a d\mathcal{D} + \int_{\mathcal{D}} \sigma \mathbf{w}'_a \cdot \frac{\mathbf{A}_{(i)}}{\Delta t} d\mathcal{D} \\ \int_{\mathcal{D}} \sigma \mathbf{grad} w'_n \left(\frac{\mathbf{A}_{(i+1)}}{\Delta t} + \mathbf{grad} \varphi_{(i+1)} \right) d\mathcal{D} &= \int_{\mathcal{D}} \sigma \mathbf{grad} w'_n \cdot \frac{\mathbf{A}_{(i)}}{\Delta t} d\mathcal{D} \end{aligned} \quad (9.85)$$

We know that we can write:

$$\begin{aligned} \mathbf{A}_{(i+1)} &= \sum_{a=1}^{n_1} a_a(i+1) \mathbf{w}_a^1 \\ \mathbf{A}_{(i)} &= \sum_{a=1}^{n_1} a_a(i) \mathbf{w}_a^1 \\ \varphi_{(i+1)} &= \sum_{n=1}^{n_0} \phi_n(i+1) w_n^0 \end{aligned} \quad (9.86)$$

The system of equations thus becomes:

$$\begin{aligned}
& \sum_{a=1}^{n_1} a_a(i+1) \left[\int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{w}'_a \cdot \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} + \frac{1}{\Delta t} \int_{\mathcal{D}} \sigma \mathbf{w}'_a \cdot \mathbf{w}_a^1 d\mathcal{D} \right] + \\
& \sum_{n=1}^{n_0} \phi_n(i+1) \int_{\mathcal{D}} \sigma \mathbf{w}'_a w_n^0 d\mathcal{D} = \int_{\mathcal{D}} \mathbf{J}_{s(i+1)} \cdot \mathbf{w}'_a d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{w}'_a d\mathcal{D} + \\
& \sum_{a=1}^{n_1} a_a(i) \frac{1}{\Delta t} \int_{\mathcal{D}} \sigma \mathbf{w}'_a \cdot \mathbf{w}_a^1 d\mathcal{D}
\end{aligned} \tag{9.87}$$

$$\begin{aligned}
& \sum_{a=1}^{n_1} a_a(i+1) \frac{1}{\Delta t} \int_{\mathcal{D}} \sigma \mathbf{grad} w_n^0 \mathbf{w}_a^1 d\mathcal{D} + \sum_{n=1}^{n_0} \phi_n(i+1) \int_{\mathcal{D}} \sigma \mathbf{grad} w_n^0 \mathbf{grad} w_n^0 d\mathcal{D} = \\
& + \sum_{a=1}^{n_1} a_a(i) \frac{1}{\Delta t} \int_{\mathcal{D}} \sigma \mathbf{grad} w_n^0 \mathbf{w}_a^1 d\mathcal{D}
\end{aligned} \tag{9.88}$$

9.5.2.2 Formulation T-Ω

The expression obtained previously for the weak integral form of vector electric potential and scalar magnetic potential is:

$$\begin{aligned}
& \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T} \cdot \mathbf{rot} \mathbf{w}'_a + \mathbf{w}'_a \cdot \frac{\partial}{\partial t} \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} = \\
& \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} \mathbf{w}'_a + \mathbf{w}'_a \cdot \frac{\partial}{\partial t} (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \tag{9.48}
\end{aligned}$$

$$\int_{\mathcal{D}} \left[\mathbf{grad} w_n^0 \cdot \mu (\mathbf{T} - \mathbf{grad} \Omega) \right] d\mathcal{D} = \int_{\mathcal{D}} \left[\mathbf{grad} w_n^0 \cdot (\mu \mathbf{H}_s + \mathbf{B}_r) \right] d\mathcal{D} \tag{9.49}$$

Only the first equation is explicitly time dependent. Given the backward Euler method discretisation, it becomes:

$$\begin{aligned}
& \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T}_{(i+1)} \cdot \mathbf{rot} \mathbf{w}'_a + \mathbf{w}'_a \cdot \frac{1}{\Delta t} \mu (\mathbf{T}_{(i+1)} - \mathbf{T}_{(i)} - \mathbf{grad}_{(i+1)} + \mathbf{grad}_{(i)} \Omega) \right] d\mathcal{D} = \\
& \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_s \cdot \mathbf{rot} \mathbf{w}'_a + \mathbf{w}'_a \cdot \frac{1}{\Delta t} (\mu \mathbf{H}_{s(i+1)} - \mathbf{H}_{s(i)} + \mathbf{B}_{r(i+1)} - \mathbf{B}_{r(i)}) \right] d\mathcal{D} \tag{9.89}
\end{aligned}$$

and further:

$$\begin{aligned}
& \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{T}_{(i+1)} \cdot \mathbf{rot} \mathbf{w}'_a + \mathbf{w}'_a \cdot \frac{1}{\Delta t} \mu (\mathbf{T}_{(i+1)} - \mathbf{grad} \Omega_{(i+1)}) \right] d\mathcal{D} = \\
& \int_{\mathcal{D}} \left[\frac{1}{\sigma} \mathbf{rot} \mathbf{H}_{s(i+1)} \cdot \mathbf{rot} \mathbf{w}'_a + \mathbf{w}'_a \cdot \frac{1}{\Delta t} (\mu \mathbf{H}_{s(i+1)} - \mu \mathbf{H}_{s(i)} + \mathbf{B}_{r(i+1)} - \mathbf{B}_{r(i)}) \right] d\mathcal{D} \\
& + \int_{\mathcal{D}} \mathbf{w}'_a \cdot \frac{1}{\Delta t} \mu (\mathbf{T}_{(i)} - \mathbf{grad} \Omega_{(i)}) d\mathcal{D} \tag{9.90}
\end{aligned}$$

9.6 Equations with overall values

9.6.1 Case of an imposed voltage on a wound conductor

We have established an additional equation 3.43 for imposed voltage on a wound conductor:

$$\frac{d}{dt} \int_{\mathcal{D}} \mathbf{A} \cdot \mathbf{N} d\mathcal{D} + Ri = V \quad (9.91)$$

As before this equation is discretised using the backward Euler method:

$$\int_{\mathcal{D}} \frac{\mathbf{A}_{(i+1)}}{\Delta t} \cdot \mathbf{N} d\mathcal{D} - \int_{\mathcal{D}} \frac{\mathbf{A}_{(i)}}{\Delta t} \cdot \mathbf{N} d\mathcal{D} + Ri = V \quad (9.92)$$

We know that we can write:

$$\begin{aligned} \mathbf{A}_{(i+1)} &= \sum_{a=1}^{n_1} a_a(i+1) \mathbf{w}_a^1 \\ \mathbf{A}_{(i)} &= \sum_{a=1}^{n_1} a_a(i) \mathbf{w}_a^1 \end{aligned}$$

This leads to the additional equation:

$$\sum_{a=1}^{n_1} a_a(i+1) \frac{1}{\Delta t} \int_{\mathcal{D}} \mathbf{w}_a^1 \cdot \mathbf{N} d\mathcal{D} + Ri = \sum_{a=1}^{n_1} a_a(i) \frac{1}{\Delta t} \int_{\mathcal{D}} \mathbf{w}_a^1 \cdot \mathbf{N} d\mathcal{D} \quad (9.93)$$

9.6.2 Case of a surface insulator

This property is only found in the spectral version.

To simplify the notation, consider a completely conductive domain of study cut into two parts, denoted D_1 and D_2 by an insulating surface denoted S . Let M be the mesh of volume finite elements of the entire domain, M_1 the volume sub-mesh associated with sub-domain D_1 and M_2 that of sub-domain D_2 . The unknowns in both sub-meshes M_1 and M_2 are numbered without considering contact between the two meshes, i.e. the edges and nodes belonging to the insulating surface have split and different unknowns depending on the sub-mesh considered.

The matrix assembled in the insulation is constructed by considering fictitious volumes. The electric elementary matrix of a prism is associated with the triangle-type surface finite elements, and the electric elementary matrix of a hexahedron is associated with the quadrangular surface elements.

For the magnetic part, only the continuity condition of the magnetic values between the two isolated sub-domains is considered. This continuity condition is represented mathematically by:

$$\begin{pmatrix} L_{11} & \dots & L_{1n} \\ \vdots & \ddots & \vdots \\ L_{n1} & \dots & L_{nn} \end{pmatrix} \mathbf{A}_S^{D_1} = \mathbf{A}_S^{D_2} \quad (9.94)$$

Where $L_{ij} = \delta_{ij}$, with δ_{ij} the Kronecker symbol. Vector $\mathbf{A}_S^{D_1}$ corresponds to the magnetic unknowns contained in the insulating surface and viewed from medium D_1 . The finite element matrix assembled on the virtual elements associated with the insulator is thus written:

$$\begin{pmatrix} \mathbf{L} & \mathbf{0} \\ \mathbf{0} & \mathbf{grad} \end{pmatrix} \quad (9.95)$$

Finally, the total matrix system to be solved in the presence of an insulator is written:

$$\begin{pmatrix} \mathbf{RotRot} + \mathbf{WW} & \mathbf{WGrad}^T \\ \mathbf{WGrad} & \mathbf{GradGrad} \end{pmatrix} \begin{pmatrix} \mathbf{A} \\ \boldsymbol{\varphi} \end{pmatrix} + \begin{pmatrix} \mathbf{L} & \mathbf{0} \\ \mathbf{0} & \mathbf{grad} \end{pmatrix} \begin{pmatrix} \mathbf{A}_{iso} \\ \boldsymbol{\varphi}_{iso} \end{pmatrix} = \begin{pmatrix} \mathbf{J} \\ \mathbf{0} \end{pmatrix} \quad (9.96)$$

Where \mathbf{A}_{iso} and φ_{iso} are the magnetic and electrical unknowns respectively on both sides of the surface insulation.

9.7 Resolution of discrete problems

9.7.1 Generic matrix notation

In the preceding paragraphs, we have seen that the modelling of electrotechnical devices can generate a number of different problems, depending on the formulation used and whether or not electrical or mechanical coupling is taken into account. Using the approaches described above, all these models can be represented by the following generic problem:

Find $\mathbf{X}(t) \in \mathbb{R}^N$ such that:

$$\mathbf{K} \frac{d\mathbf{X}(t)}{dt} + (\mathbf{M}_\theta(\theta) + \mathbf{M}(\mathbf{X})) \mathbf{X}(t) = \mathbf{C} \mathbf{U}(t), \quad \forall t \in [0, T], \quad (9.97)$$

and find $\theta(t) \in \mathbb{R}$ such that:

$$J_M \frac{d^2\theta(t)}{dt^2} + f_M \frac{d\theta(t)}{dt} = \Gamma_B(\mathbf{X}) + \Gamma_M(t) \quad (9.98)$$

9.7.2 Time discretisation

To solve equations 9.97 and 9.98, the time domain $[0, T]$ is discretised in N_t regular intervals separated by a time step $\tau = \frac{T}{N_t}$. The choice of this time step is not insignificant: it should be small enough to identify the different dynamics of the problem (electrical, magnetic or mechanical). Hence, we will only solve the problem for N_t time $t^k = k\tau$, $k = 1, \dots, N_t$, with the initial conditions imposed for $t^0 = 0$. We thus define the notation

$$\mathbf{X}(t^k) = \mathbf{X}^k, \quad k = 0, \dots, N_t \quad (9.99)$$

The next step is to express the time derivatives, namely $\frac{d\mathbf{X}}{dt}(t^k)$ and $\frac{d^2\theta}{dt^2}(t^k)$.

9.7.2.1 Time discretisation of the magnetic equation

To discretise the magnetic equation, we are interested in the expression of the time derivative of $\mathbf{X}(t)$. Here we use a backward Euler method that has the advantage of being both stable and easily to implement. In this case, the latter is written:

$$\frac{d\mathbf{X}}{dt}(t^k) \simeq \frac{\mathbf{X}^k - \mathbf{X}^{k-1}}{\tau} \quad k = 1, \dots, N_t \quad (9.100)$$

By putting this expression into the magnetic equation, we write:

$$\left(\frac{\mathbf{K}}{\tau} + \mathbf{M}_\theta(\theta) + \mathbf{M}(\mathbf{X}^k) \right) \mathbf{X}^k = \mathbf{C} \mathbf{U}^k + \frac{\mathbf{K}}{\tau} \mathbf{X}^{k-1}, \quad k = 1, \dots, N_t \quad (9.101)$$

9.7.2.2 Time discretisation of the mechanical equation

For the second order mechanical equation in time, we break it down into two first order equations by introducing $\Omega = \frac{d\theta}{dt}$. We thus write:

$$\begin{cases} \frac{d\Omega}{dt}(t) &= (J_M)^{-1} (-f_M \Omega(t) + \Gamma_B(\mathbf{X}(t)) + \Gamma_M) \\ \frac{d\theta}{dt}(t) &= \Omega(t) \end{cases} \quad (9.102)$$

To solve this expression, a forward Euler method is used for the first and a backward Euler method for the second. The use of a forward method is consistent, as the time characteristic of the mechanical equation τ_M in electrotechnical applications is very large compared with that of the magnetic problem τ_B ($\tau_B \ll \tau_M$). Thus, the time discretisation step τ is chosen to be small compared with τ_B and hence very much smaller than τ_M . Thus, the time discretisation error due to the use of a forward method on the mechanical equation is very low. We thus write:

$$\frac{d\Omega}{dt}(t^k) = \left(\frac{d\Omega}{dt}\right)^k \simeq \frac{\Omega^{k+1} - \Omega^k}{\tau} \quad (9.103)$$

Although a forward method could be used on the second equation for the same reasons, a backward method is preferred because it results in a lower numerical error for an equivalent complexity of implementation. Thus, we have:

$$\frac{d\theta}{dt}(t^k) = \left(\frac{d\theta}{dt}\right)^k \simeq \frac{\theta^k - \theta^{k-1}}{\tau} \quad (9.104)$$

The discretisation of these two equations is thus written:

$$\begin{cases} \Omega^k &= \left(1 - \frac{\tau f_M}{J_M}\right) \Omega^{k-1} + \frac{\tau}{J_M} (\Gamma_B(\mathbf{X}^{k-1}) + \Gamma_M) \\ \theta^k &= \theta^{k-1} + \tau \Omega^k \end{cases} \quad (9.105)$$

9.7.2.3 Time discretisation of the generic problem

Finally, time discretisation of the generic problem is written:

Find $\mathbf{X}^k(t) \in \mathbb{R}^N$ such that:

$$\left(\frac{\mathbf{K}}{\tau} + \mathbf{M}_\theta(\theta) + \mathbf{M}(\mathbf{X}^k)\right) \mathbf{X}^k = \mathbf{C} \mathbf{U}^k + \frac{\mathbf{K}}{\tau} \mathbf{X}^{k-1}, \quad k = 1, \dots, N^t \quad (9.106)$$

and find $(\theta^{k+1}, \Omega^{k+1}) \in \mathbb{R}^2$ such that:

$$\begin{cases} \Omega^{k+1} &= \left(1 - \frac{\tau f_M}{J_M}\right) \Omega^k + \frac{\tau}{J_M} (\Gamma_B(\mathbf{X}^k) + \Gamma_M) \\ \theta^{k+1} &= \theta^k + \tau \Omega^{k+1} \end{cases}, \quad k = 0, \dots, N_t - 1 \quad (9.107)$$

Remark 9.7.1 To explain the chaining of the two models, the mechanical equation has been written at time step $k+1$. By knowing θ^k , equation 9.106 allows calculation of \mathbf{X}^k . And by knowing \mathbf{X}^k , the set of mechanical equations 9.107 allows calculation of θ^{k+1} making it possible to obtain \mathbf{X}^{k+1} and so on.

Part III

Construction of the matrix system

Chapter 10

Implementation of Finite Element Method in code_Carmel

Résumé

In [Nédélec 1992] or [Henneron 2004] and in Chapter 2 of [Girault 2006] we find a description of the mixed finite elements to be used to discretise variational formulations $\mathbf{A} - \phi$ and $\mathbf{T} - \Omega$. In code_Carmel, we use the following finite elements:

- scalar of class \mathbf{H}_{grad} (nodal finite elements);
- vector of class \mathbf{H}_{rot} (edge finite elements or Nedélec finite elements);
- vector of class \mathbf{H}_{div} (facet finite elements or Raviart-Thomas finite elements).

These elements have been interpreted geometrically as Whitney elements by Alain Bossavit, and we refer to [Bossavit, Vérté 1983] for a detailed presentation of this aspect. Here, we follow [Girault 2006].

10.1 Finite elements used

A finite element is defined by:

- a geometric element K : in code_Carmel, the geometric element belongs to \mathbb{R}^3 and can be a tetrahedron (T), a prism (Pr), a hexahedron (H), a pyramid (Py);
- a vector space of dimension N of scalar or vector functions defined on K denoted P_K . In code_Carmel, the approximating spaces are polynomial spaces, either scalar or vector. A basis of the approximating space is called a basis function;
- a set of N linear shapes on the space of scalar or vector functions defined on K : the degrees of freedom.

For each geometric element, we define the finite element of lowest degree of class H^1 , the finite element of lowest degree of class \mathbf{H}_{rot} and the finite element of lowest degree of class \mathbf{H}_{div} .

For each geometric element, a special representative is introduced: the reference element (\widehat{T} , \widehat{Pr} , \widehat{H} or \widehat{Py}). The basis functions defined on this element will be transformed to construct the basis functions of any element of the triangulation. The coordinates in an orthonormal coordinate system are denoted (u, v, w) .

10.2 Reference elements and shape functions used

10.2.1 Case of the tetrahedron

The geometric element T is a tetrahedron defined by its 4 vertices $(s^i)_{i=1,4}$. There are 6 edges $(a^i)_{i=1,6}$ and 4 facets $(f^i)_{i=1,4}$.

The reference tetrahedron \hat{T} has vertices:

$$s^1 = (0, 0, 0), s^2 = (1, 0, 0), s^3 = (0, 1, 0), s^4 = (0, 0, 1).$$

The numbering of nodes, edges and facets of \hat{T} is shown in Figure 10.1.

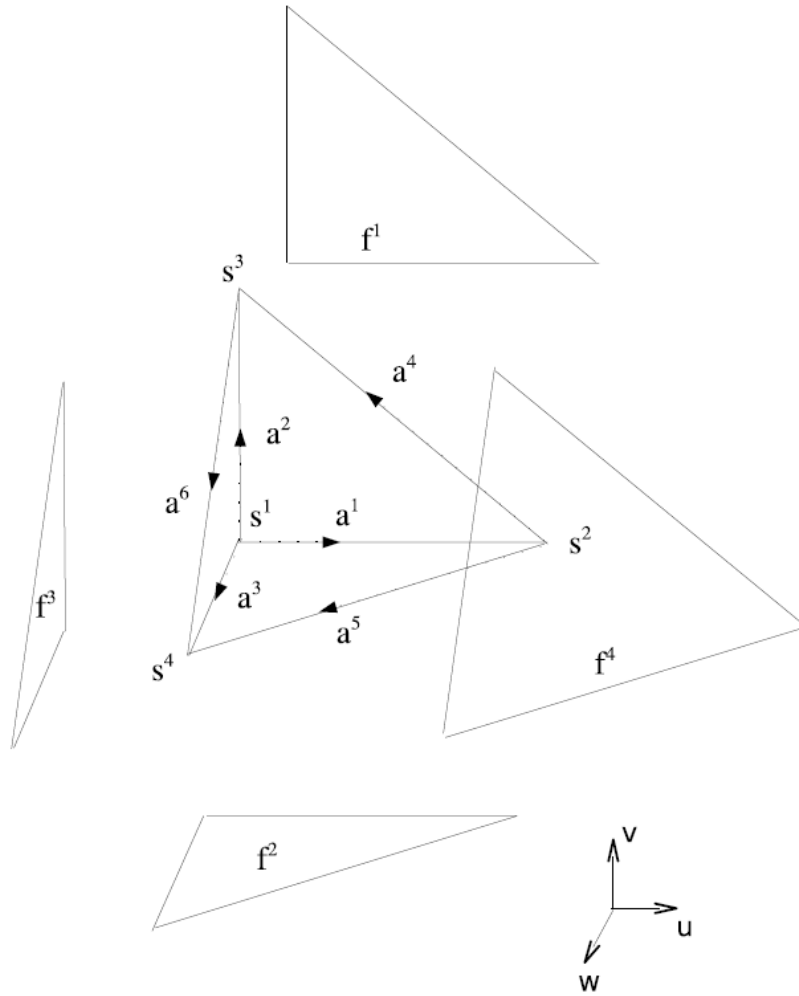


Figure 10.1: Illustration of the reference tetrahedron

10.2.1.1 Finite element P_1 de classe H^1

The approximating space P_T is the space P_1 of polynomials with 3 real variables, of real value:

$$P_T = P_1 = \{p, p(\mathbf{x}) = c_o + \mathbf{c}_1 \cdot \mathbf{x}, c_o \in \mathbb{R}, \mathbf{c}_1 \in \mathbb{R}^3\}$$

This space is of dimension 4.

The degrees of freedom are the values at the vertices of the tetrahedron, hence the name of nodal finite elements:

$$\Sigma_T = \{\sigma_i/\sigma_i(f) = f(s^i), i = 1, 4\}$$

and the basis functions are the barycentric coordinates $(\lambda_i)_{i=1,4}$ defined at any point \mathbf{x} by:

$$\sum_{j=1}^4 s^j \lambda_j(x) = x, \quad \sum_{j=1}^4 \lambda_j(x) = 1$$

We denote w_n^0 the basis function associated with vertex n , i.e. $w_n^0 = \lambda_n$.

For the reference tetrahedron, the basis functions are as follows:

$$\begin{aligned} w_1^0(u, v, w) &= 1 - u - v - w \\ w_2^0(u, v, w) &= u \\ w_3^0(u, v, w) &= v \\ w_4^0(u, v, w) &= w \end{aligned} \tag{10.1}$$

10.2.1.2 Finite element of class \mathbf{H}_{rot}

The approximating space \mathbf{P}_T is included in the space \mathbf{P}_1 of polynomials with 3 real variables, with a value in \mathbb{R}^3 :

$$\mathbf{P}_1 = (P_1)^3$$

More precisely, \mathbf{P}_T is the dimension 6 space defined by:

$$\mathbf{P}_T = \{\mathbf{p} \in (P_1)^3, \mathbf{p}(\mathbf{x}) = \mathbf{c}_0 + \mathbf{c}_1 \wedge \mathbf{x}, \mathbf{c}_0, \mathbf{c}_1 \in \mathbb{R}^3\}$$

The degrees of freedom are the circulations on the edges of T :

$$\Sigma_T = \left\{ \sigma_i/\sigma_i(\mathbf{f}) = \int_{a^i} \mathbf{f} ds, i = 1, 4 \right\}$$

Denoting T the signed volume of the tetrahedron. We have:

$$T = \frac{1}{6}(\mathbf{a}_1 \wedge \mathbf{a}_2) \cdot \mathbf{a}^3$$

Remark 10.2.1 For a tetrahedron orientated like the reference tetrahedron, the mixed product $(\mathbf{a}_1 \wedge \mathbf{a}_2) \cdot \mathbf{a}^3$ is positive.

The basis function associated with edge i is denoted w_i^1 . We have [Nédélec 1992]:

$$\begin{aligned} w_1^1(x) &= \frac{a^6 \wedge (x - s^3)}{6T}, & w_2^1(x) &= -\frac{a^5 \wedge (x - s^2)}{6T} \\ w_3^1(x) &= \frac{a^4 \wedge (x - s^2)}{6T}, & w_4^1(x) &= \frac{a^3 \wedge (x - s^1)}{6T} \\ w_5^1(x) &= -\frac{a^2 \wedge (x - s^1)}{6T}, & w_6^1(x) &= \frac{a^1 \wedge (x - s^1)}{6T} \end{aligned} \tag{10.2}$$

Another expression of the basis function relative to the edge connecting s^i and s^j is:

$$\lambda_i \cdot \mathbf{grad}(\lambda_j) - \lambda_j \cdot \mathbf{grad}(\lambda_i) \quad (10.3)$$

We obtain the basis functions for the reference tetrahedron \hat{T} from 10.3:

$$\begin{aligned} w_1^1(u, v, w) &= \begin{pmatrix} 1-v-w \\ u \\ u \end{pmatrix} & w_2^1(u, v, w) &= \begin{pmatrix} v \\ 1-v-w \\ v \end{pmatrix} \\ w_3^1(u, v, w) &= \begin{pmatrix} w \\ w \\ 1-v-w \end{pmatrix} & w_4^1(u, v, w) &= \begin{pmatrix} -v \\ u \\ 0 \end{pmatrix} \\ w_5^1(u, v, w) &= \begin{pmatrix} -w \\ 0 \\ u \end{pmatrix} & w_6^1(u, v, w) &= \begin{pmatrix} 0 \\ -w \\ v \end{pmatrix} \end{aligned} \quad (10.4)$$

The curls of the basis functions $(\mathbf{w}_i^1)_{i=1,6}$ are constant in each tetrahedron. In the reference tetrahedron, they are given by:

$$\begin{aligned} \mathbf{rot}(w_1^1)(u, v, w) &= 2 \begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix} & \mathbf{rot}(w_2^1)(u, v, w) &= 2 \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} \\ \mathbf{rot}(w_3^1)(u, v, w) &= 2 \begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix} & \mathbf{rot}(w_4^1)(u, v, w) &= 2 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \\ \mathbf{rot}(w_5^1)(u, v, w) &= 2 \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix} & \mathbf{rot}(w_6^1)(u, v, w) &= 2 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \end{aligned} \quad (10.5)$$

10.2.1.3 Finite element of class \mathbf{H}_{div}

The approximating space \mathbf{P}_T is a sub-space of dimension 4 of space \mathbf{P}_1 .

$$\mathbf{P}_T = \{\mathbf{p} \in P_1, \mathbf{p}(\mathbf{x}) = \mathbf{c}_0 + c_1 \mathbf{x}, \mathbf{c}_0 \in \mathbb{R}^3, c_1 \in \mathbb{R}\}$$

The degrees of freedom are the fluxes through the facets f_i of T :

$$\Sigma_T = \left\{ \sigma_i / \sigma_i(\mathbf{g}) = \int_{f_i} (\mathbf{g} \cdot \mathbf{n} ds), i = 1, 4 \right\}$$

The basis functions associated with the faces of the tetrahedron are given by:

$$\begin{aligned} w_1^2(u, v, w) &= \frac{s^4 - x}{3|T|} & w_2^2(u, v, w) &= \frac{s^3 - x}{3|T|} \\ w_3^2(u, v, w) &= \frac{s^2 - x}{3|T|} & w_4^2(u, v, w) &= \frac{s^1 - x}{3|T|} \end{aligned} \quad (10.6)$$

In code_Carmel, the general formula is used for the basis function relating to face i,j,k ([Geuzaine 2001] p.41)

$$2(w_i^0 \mathbf{grad} w_j^0 \wedge \mathbf{grad} w_k^0 + w_j^0 \mathbf{grad} w_k^0 \wedge \mathbf{grad} w_i^0 + w_k^0 \mathbf{grad} w_i^0 \wedge \mathbf{grad} w_j^0)$$

We thus have the following expressions for the basis functions relative to the facets of tetrahedron \hat{T} :

$$\begin{aligned}
w_1^2(u, v, w) &= 2(w_1^0 \mathbf{grad} w_2^0 \wedge \mathbf{grad} w_3^0 + w_2^0 \mathbf{grad} w_3^0 \wedge \mathbf{grad} w_1^0 \\
&\quad + w_3^0 \mathbf{grad} w_1^0 \wedge \mathbf{grad} w_2^0) \\
w_2^2(u, v, w) &= 2(w_1^0 \mathbf{grad} w_2^0 \wedge \mathbf{grad} w_4^0 + w_2^0 \mathbf{grad} w_4^0 \wedge \mathbf{grad} w_1^0 \\
&\quad + w_4^0 \mathbf{grad} w_1^0 \wedge \mathbf{grad} w_2^0) \\
w_3^2(u, v, w) &= 2(w_1^0 \mathbf{grad} w_3^0 \wedge \mathbf{grad} w_4^0 + w_3^0 \mathbf{grad} w_4^0 \wedge \mathbf{grad} w_1^0 \\
&\quad + w_4^0 \mathbf{grad} w_1^0 \wedge \mathbf{grad} w_3^0) \\
w_4^2(u, v, w) &= 2(w_2^0 \mathbf{grad} w_3^0 \wedge \mathbf{grad} w_4^0 + w_3^0 \mathbf{grad} w_4^0 \wedge \mathbf{grad} w_2^0 \\
&\quad + w_4^0 \mathbf{grad} w_2^0 \wedge \mathbf{grad} w_3^0)
\end{aligned} \tag{10.7}$$

Remark 10.2.2 According to [Deliège 2003] (p. 183), the expression of the basis functions can also be used directly.

$$\begin{aligned}
w_1^2(u, v, w) &= 2 \begin{pmatrix} u \\ v \\ -1 + w \end{pmatrix} \\
w_2^2(u, v, w) &= 2 \begin{pmatrix} u \\ -1 + v \\ w \end{pmatrix} \\
w_3^2(u, v, w) &= 2 \begin{pmatrix} -1 + u \\ v \\ w \end{pmatrix} \\
w_4^2(u, v, w) &= 2 \begin{pmatrix} u \\ v \\ w \end{pmatrix}
\end{aligned} \tag{10.8}$$

10.2.1.3.1 Case of the prism

The geometric element Pr is a right prism defined by its 6 vertices $(s^i)_{i=1,6}$. There are 9 edges $(a^i)_{i=1,9}$ and 4 facets $(f^i)_{i=1,5}$ (3 rectangular and 2 triangular).

The reference prism \widehat{Pr} has vertices:

$$\begin{aligned}
s^1 &= (0, 0, -1), & s^2 &= (1, 0, -1), & s^3 &= (0, 1, -1), \\
s^4 &= (0, 0, 1), & s^5 &= (1, 0, 1), & s^6 &= (0, 1, 1).
\end{aligned}$$

The numbering of nodes, edges and facets of \widehat{Pr} is shown in Figure 10.2.

Finite element of class H^1 We consider the reference prism: the triangular face is in the plane (u, v) . The approximating space P_P is a space of dimension 6. This is the set of polynomials of 3 real variables, of degree 1 in (u, v) and degree 1 in w .

$$P_P = \{p : (u, v, w) \mapsto p(u, v, w) = q(u, v) r(w), q \in P_1(u, v), r \in P_1(w)\}$$

Remark 10.2.3 This space is included in that of polynomials of degree 2 (and not 1).

The degrees of freedom are the values at the vertices of the prism:

$$\Sigma_P = \{\sigma_i / \sigma_i(f) = f(s^i), i = 1, 6\}$$

The basis functions w^0 for the reference prism are given by:

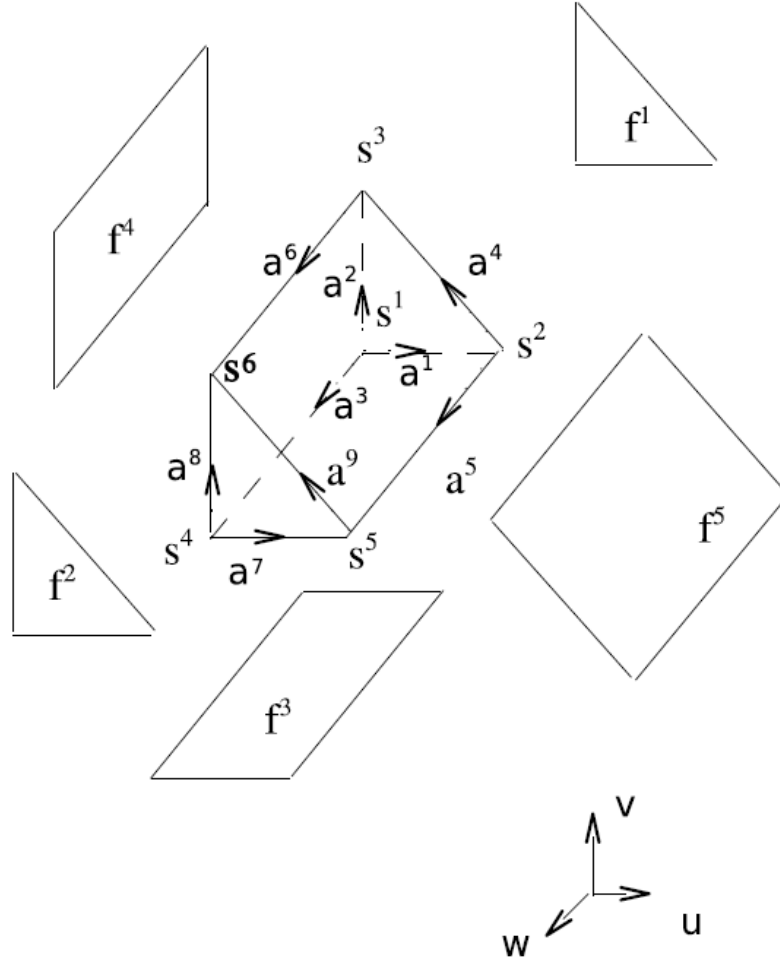


Figure 10.2: Illustration of the reference prism

$$\begin{aligned}
 w_1^0(u, v, w) &= \frac{1}{2} (1 - u - v) (1 - w) \\
 w_2^0(u, v, w) &= \frac{1}{2} u (1 - w) \\
 w_3^0(u, v, w) &= \frac{1}{2} v (1 - w) \\
 w_4^0(u, v, w) &= \frac{1}{2} (1 - u - v) (1 + w) \\
 w_5^0(u, v, w) &= \frac{1}{2} u (1 + w) \\
 w_6^0(u, v, w) &= \frac{1}{2} v (1 + w)
 \end{aligned} \tag{10.9}$$

Finite element of class \mathbf{H}_{rot} The approximating space \mathbf{P}_P is a sub-space of \mathbf{P}_1 of dimension 9:

$$\mathbf{P}_P = \left\{ \mathbf{p} : (u, v, w) \mapsto \mathbf{p}(u, v, w) = \begin{pmatrix} \alpha_1 + \beta v + \gamma_1 w + \delta v w \\ \alpha_2 - \beta v + \gamma_2 w - \delta u w \\ \alpha_3 + \epsilon_1 u + \epsilon_2 v \end{pmatrix} \right\}$$

where: $\alpha_1, \alpha_2, \alpha_3, \beta, \gamma_1, \gamma_2, \delta, \epsilon_1, \epsilon_2$ are real coefficients.

The degrees of freedom are the circulations on the edges of P :

$$\Sigma_T = \left\{ \sigma_i / \sigma_i(\mathbf{f}) = \int_{a^i} \mathbf{f} ds, i = 1, 9 \right\}$$

The basis function relative to the edge joining node i and node j is given by the formula ([Geuzaine 2001] p. 41):

$$w_j^0 \mathbf{grad} \sum_{r \in \mathcal{N}(j, \bar{i})} w_r^0 - w_i^0 \mathbf{grad} \sum_{r \in \mathcal{N}(i, \bar{j})} w_r^0 \quad (10.10)$$

where $\mathcal{N}(m, \bar{n})$ is the set of nodes on the face that contains node m and not node n .

The basis functions associated with the edges of reference prism \hat{P} are:

$$\begin{aligned} w_1^1(u, v, w) &= \frac{1}{2}(1-w) \begin{pmatrix} 1-v \\ u \\ 0 \end{pmatrix} & w_2^1(u, v, w) &= \frac{1}{2}(1-w) \begin{pmatrix} v \\ 1-u \\ 0 \end{pmatrix} \\ w_3^1(u, v, w) &= \frac{1}{2} \begin{pmatrix} 0 \\ 0 \\ 1-u-v \end{pmatrix} & w_4^1(u, v, w) &= \frac{1}{2}(1-w) \begin{pmatrix} -v \\ u \\ 0 \end{pmatrix} \\ w_5^1(u, v, w) &= \frac{1}{2} \begin{pmatrix} 0 \\ 0 \\ u \end{pmatrix} & w_6^1(u, v, w) &= \frac{1}{2} \begin{pmatrix} 0 \\ 0 \\ v \end{pmatrix} \\ w_7^1(u, v, w) &= \frac{1}{2}(1+w) \begin{pmatrix} 1-v \\ 1-u \\ 0 \end{pmatrix} & w_8^1(u, v, w) &= \frac{1}{2}(1+w) \begin{pmatrix} v \\ 1-u \\ 0 \end{pmatrix} \\ w_9^1(u, v, w) &= \frac{1}{2}(1+w) \begin{pmatrix} -v \\ u \\ 0 \end{pmatrix} \end{aligned} \quad (10.11)$$

Finite element of class \mathbf{H}_{div} The basis function relative to facet f containing nodes i, j, k (and l if it is a quadrangular facet) is obtained by applying the general formula ([Geuzaine 2001] p.41).

$$\mathbf{w}^2 = a \sum_{q \in \mathcal{N}(f)} w_q^0 \mathbf{grad} \left(\sum_{r \in \mathcal{N}(q)(\bar{q}+1)} w_r^0 \right) \wedge \mathbf{grad} \left(\sum_{r \in \mathcal{N}(q)(\bar{q}-1)} w_r^0 \right) \quad (10.12)$$

where a is equal to 2 if f is triangular and 1 if f is quadrangular.

The basis functions relative to the facets of reference prism \hat{T} are:

$$\begin{aligned}
w_1^2(u, v, w) &= 2 [w_1^0 \mathbf{grad}(w_2^0 + w_5^0) \wedge \mathbf{grad}(w_3^0 + w_6^0) \\
&\quad + w_2^0 \mathbf{grad}(w_3^0 + w_6^0) \wedge \mathbf{grad}(w_1^0 + w_4^0) \\
&\quad + w_3^0 \mathbf{grad}(w_1^0 + w_4^0) \wedge \mathbf{grad}(w_2^0 + w_5^0)] \\
w_2^2(u, v, w) &= 2 [w_4^0 \mathbf{grad}(w_2^0 + w_5^0) \wedge \mathbf{grad}(w_3^0 + w_6^0) \\
&\quad + w_5^0 \mathbf{grad}(w_3^0 + w_6^0) \wedge \mathbf{grad}(w_1^0 + w_4^0) \\
&\quad + w_6^0 \mathbf{grad}(w_1^0 + w_4^0) \wedge \mathbf{grad}(w_2^0 + w_5^0)] \\
w_3^2(u, v, w) &= w_1^0 \mathbf{grad}(w_2^0 + w_5^0) \wedge \mathbf{grad}(w_4^0 + w_5^0 + w_6^0) \\
&\quad + w_2^0 \mathbf{grad}(w_4^0 + w_5^0 + w_6^0) \wedge \mathbf{grad}(w_1^0 + w_4^0) \\
&\quad + w_4^0 \mathbf{grad}(w_1^0 + w_2^0 + w_3^0) \wedge \mathbf{grad}(w_2^0 + w_5^0) \\
&\quad + w_5^0 \mathbf{grad}(w_1^0 + w_4^0) \wedge \mathbf{grad}(w_1^0 + w_2^0 + w_3^0) \\
w_4^2(u, v, w) &= w_2^0 \mathbf{grad}(w_3^0 + w_6^0) \wedge \mathbf{grad}(w_4^0 + w_5^0 + w_6^0) \\
&\quad + w_3^0 \mathbf{grad}(w_4^0 + w_5^0 + w_6^0) \wedge \mathbf{grad}(w_1^0 + w_4^0) \\
&\quad + w_5^0 \mathbf{grad}(w_1^0 + w_2^0 + w_3^0) \wedge \mathbf{grad}(w_3^0 + w_6^0) \\
&\quad + w_6^0 \mathbf{grad}(w_2^0 + w_5^0) \wedge \mathbf{grad}(w_1^0 + w_2^0 + w_3^0)
\end{aligned} \tag{10.13}$$

10.2.1.3.2 Case of the hexahedron

The geometric element H is a right prism defined by its 8 vertices $(s^i)_{i=1,8}$. There are 12 edges $(a^i)_{i=1,12}$ and 6 facets $(f^i)_{i=1,6}$.

The reference hexahedron \hat{P} has vertices:

$$\begin{aligned}
s^1 &= (-1, -1, -1), & s^2 &= (1, -1, -1), & s^3 &= (1, 1, -1), & s^4 &= (-1, 1, 1), \\
s^5 &= (-1, -1, 1), & s^6 &= (1, -1, 1), & s^7 &= (1, 1, 1), & s^8 &= (-1, 1, 1),
\end{aligned}$$

The numbering of nodes, edges and facets of \hat{H} is shown in Figure 10.3.

Finite elements Q_1 of class H^1 The approximating space P_H is the space of polynomials of degree 1 in each of the variables u, v, w . It is a space of dimension 8.

$$P_H = Q_1 = \{p : (u, v, w) \mapsto p(u, v, w) = q(u) r(v) s(w), q \in P_1(u), r \in P_1(v), s \in P_1(w)\}$$

The degrees of freedom are the values at the vertices of the hexahedron:

$$\Sigma_H = \{\sigma_i / \sigma_i(f) = f(s^i), i = 1, 8\}$$

The basis functions w^0 for reference hexahedron \hat{H} are given by:

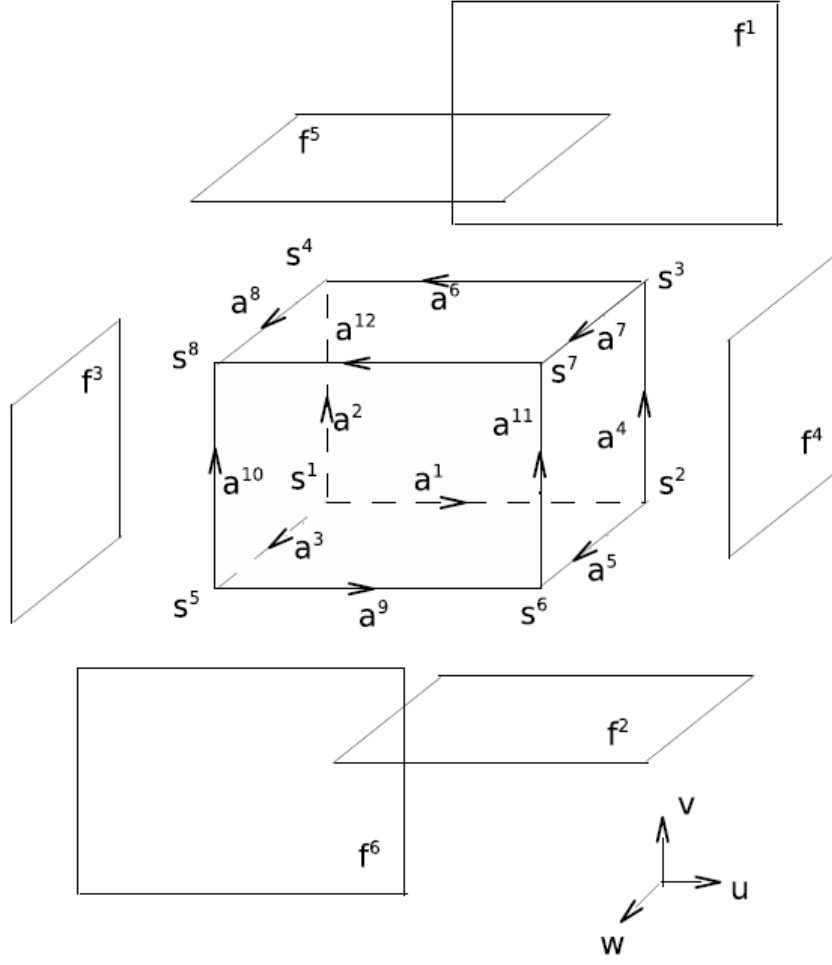


Figure 10.3: Illustration of the reference hexahedron

$$\begin{aligned}
 w_1^0(u, v, w) &= \frac{1}{8} (1 - u) (1 - v) (1 - w) \\
 w_2^0(u, v, w) &= \frac{1}{8} (1 + u) (1 - v) (1 - w) \\
 w_3^0(u, v, w) &= \frac{1}{8} (1 + u) (1 + v) (1 - w) \\
 w_4^0(u, v, w) &= \frac{1}{8} (1 - u) (1 + v) (1 - w) \\
 w_5^0(u, v, w) &= \frac{1}{8} (1 - u) (1 - v) (1 + w) \\
 w_6^0(u, v, w) &= \frac{1}{8} (1 + u) (1 - v) (1 + w) \\
 w_7^0(u, v, w) &= \frac{1}{8} (1 + u) (1 + v) (1 + w) \\
 w_8^0(u, v, w) &= \frac{1}{8} (1 - u) (1 + v) (1 + w)
 \end{aligned} \tag{10.14}$$

Finite element of class \mathbf{H}_{rot} The approximating space \mathbf{P}_H is a sub-space of polynomials of degree 2 of dimension 12:

$$\mathbf{P}_P = \left\{ \mathbf{p} : (u, v, w) \mapsto \mathbf{p}(u, v, w) = \begin{pmatrix} \alpha_1 + \beta_1 v + \gamma_1 w + \delta_1 v w \\ \alpha_2 + \beta_2 u + \gamma_2 w + \delta_2 u w \\ \alpha_3 + \beta_3 u + \gamma_3 v + \delta_3 u v \end{pmatrix} \right\}$$

where: $\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3, \gamma_1, \gamma_2, \gamma_3, \delta_1, \delta_2, \delta_3$ are real coefficients.

The degrees of freedom are the circulations on the edges of H :

$$\Sigma_P = \left\{ \sigma_i / \sigma_i(\mathbf{f}) = \int_{a^i} \mathbf{f} ds, i = 1, 12 \right\}$$

The basis functions associated with the edges of reference hexahedron \hat{H} are calculated from [10.10](#):

$$\begin{aligned} w_1^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} (1-v)(1-w) \\ 0 \\ 0 \end{pmatrix} & w_2^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} 0 \\ (1-u)(1-w) \\ 0 \end{pmatrix} \\ w_3^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} 0 \\ 0 \\ (1-u)(1-v) \end{pmatrix} & w_4^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} 0 \\ (1+u)(1-w) \\ 0 \end{pmatrix} \\ w_5^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} 0 \\ 0 \\ (1+u)(1-v) \end{pmatrix} & w_6^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} (1+v)(1+w) \\ 0 \\ 0 \end{pmatrix} \\ w_7^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} 0 \\ 0 \\ (1-u)(1+v) \end{pmatrix} & w_8^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} 0 \\ 0 \\ (1+u)(1+v) \end{pmatrix} \\ w_9^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} (1-v)(1+w) \\ 0 \\ 0 \end{pmatrix} & w_{10}^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} 0 \\ (1-u)(1+w) \\ 0 \end{pmatrix} \\ w_{11}^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} 0 \\ (1+u)(1+w) \\ 0 \end{pmatrix} & w_{12}^1(u, v, w) &= \frac{1}{8} \begin{pmatrix} (1+v)(1+w) \\ 0 \\ 0 \end{pmatrix} \end{aligned} \quad (10.15)$$

Finite element of class \mathbf{H}_{div} The basis functions relative to the facets of reference prism \hat{T} are obtained by applying the general formula [10.12](#):

$$\begin{aligned}
w_1^2(u, v, w) &= w_1^0 \mathbf{grad}(w_1^0 + w_4^0 + w_5^0 + w_8^0) \wedge \mathbf{grad}(w_1^0 + w_2^0 + w_5^0 + w_6^0) \\
&\quad + w_2^0 \mathbf{grad}(w_1^0 + w_2^0 + w_5^0 + w_6^0) \wedge \mathbf{grad}(w_2^0 + w_3^0 + w_6^0 + w_7^0) \\
&\quad + w_3^0 \mathbf{grad}(w_2^0 + w_3^0 + w_6^0 + w_7^0) \wedge \mathbf{grad}(w_3^0 + w_4^0 + w_7^0 + w_8^0) \\
&\quad + w_4^0 \mathbf{grad}(w_3^0 + w_4^0 + w_7^0 + w_8^0) \wedge \mathbf{grad}(w_1^0 + w_4^0 + w_5^0 + w_8^0) \\
w_2^2(u, v, w) &= w_1^0 \mathbf{grad}(w_1^0 + w_4^0 + w_5^0 + w_8^0) \wedge \mathbf{grad}(w_1^0 + w_2^0 + w_3^0 + w_4^0) \\
&\quad + w_2^0 \mathbf{grad}(w_1^0 + w_2^0 + w_3^0 + w_4^0) \wedge \mathbf{grad}(w_2^0 + w_3^0 + w_6^0 + w_7^0) \\
&\quad + w_5^0 \mathbf{grad}(w_5^0 + w_6^0 + w_7^0 + w_8^0) \wedge \mathbf{grad}(w_1^0 + w_4^0 + w_5^0 + w_8^0) \\
&\quad + w_6^0 \mathbf{grad}(w_2^0 + w_3^0 + w_6^0 + w_7^0) \wedge \mathbf{grad}(w_5^0 + w_6^0 + w_7^0 + w_8^0) \\
w_3^2(u, v, w) &= w_1^0 \mathbf{grad}(w_1^0 + w_2^0 + w_5^0 + w_6^0) \wedge \mathbf{grad}(w_1^0 + w_2^0 + w_3^0 + w_4^0) \\
&\quad + w_4^0 \mathbf{grad}(w_1^0 + w_2^0 + w_3^0 + w_4^0) \wedge \mathbf{grad}(w_3^0 + w_4^0 + w_7^0 + w_8^0) \\
&\quad + w_5^0 \mathbf{grad}(w_5^0 + w_6^0 + w_7^0 + w_8^0) \wedge \mathbf{grad}(w_1^0 + w_2^0 + w_5^0 + w_6^0) \\
&\quad + w_8^0 \mathbf{grad}(w_3^0 + w_4^0 + w_7^0 + w_8^0) \wedge \mathbf{grad}(w_5^0 + w_6^0 + w_7^0 + w_8^0) \\
w_4^2(u, v, w) &= w_2^0 \mathbf{grad}(w_1^0 + w_2^0 + w_5^0 + w_6^0) \wedge \mathbf{grad}(w_1^0 + w_2^0 + w_3^0 + w_4^0) \\
&\quad + w_3^0 \mathbf{grad}(w_1^0 + w_2^0 + w_3^0 + w_4^0) \wedge \mathbf{grad}(w_3^0 + w_4^0 + w_7^0 + w_8^0) \\
&\quad + w_6^0 \mathbf{grad}(w_5^0 + w_6^0 + w_7^0 + w_8^0) \wedge \mathbf{grad}(w_1^0 + w_2^0 + w_5^0 + w_6^0) \\
&\quad + w_7^0 \mathbf{grad}(w_3^0 + w_4^0 + w_7^0 + w_8^0) \wedge \mathbf{grad}(w_5^0 + w_6^0 + w_7^0 + w_8^0) \\
w_5^2(u, v, w) &= w_3^0 \mathbf{grad}(w_2^0 + w_3^0 + w_6^0 + w_7^0) \wedge \mathbf{grad}(w_1^0 + w_2^0 + w_3^0 + w_4^0) \\
&\quad + w_4^0 \mathbf{grad}(w_1^0 + w_2^0 + w_3^0 + w_4^0) \wedge \mathbf{grad}(w_1^0 + w_4^0 + w_5^0 + w_8^0) \\
&\quad + w_7^0 \mathbf{grad}(w_5^0 + w_6^0 + w_7^0 + w_8^0) \wedge \mathbf{grad}(w_2^0 + w_3^0 + w_6^0 + w_7^0) \\
&\quad + w_8^0 \mathbf{grad}(w_1^0 + w_4^0 + w_5^0 + w_8^0) \wedge \mathbf{grad}(w_5^0 + w_6^0 + w_7^0 + w_8^0) \\
w_6^2(u, v, w) &= w_5^0 \mathbf{grad}(w_1^0 + w_4^0 + w_5^0 + w_8^0) \wedge \mathbf{grad}(w_1^0 + w_2^0 + w_5^0 + w_6^0) \\
&\quad + w_6^0 \mathbf{grad}(w_1^0 + w_2^0 + w_5^0 + w_6^0) \wedge \mathbf{grad}(w_2^0 + w_3^0 + w_6^0 + w_7^0) \\
&\quad + w_7^0 \mathbf{grad}(w_2^0 + w_3^0 + w_6^0 + w_7^0) \wedge \mathbf{grad}(w_3^0 + w_4^0 + w_7^0 + w_8^0) \\
&\quad + w_8^0 \mathbf{grad}(w_3^0 + w_4^0 + w_7^0 + w_8^0) \wedge \mathbf{grad}(w_1^0 + w_4^0 + w_5^0 + w_8^0)
\end{aligned} \tag{10.16}$$

10.2.2 Case of the pyramid

Unlike the prism, hexahedron or tetrahedron, the pyramid is a slightly trickier element because of its particular connectivity: unlike the classic 3 elements where each node is connected to 3 edges, the vertex of the pyramid is connected to 4 edges. The result is less regular shape functions.

The pyramid element \widehat{Py} and associated shape functions are derived from the excellent paper by Gradinaru and Hiptmair [Gradinaru 1999] (which nevertheless contains an error for edge functions 6 and 7), where the shape functions are determined by cutting the pyramid into two tetrahedra. Their expression can also be found using Whitney formulas. It should be noted that the facet functions differ in each case, which we will examine more closely. In other versions of Carmel in which pyramids have been implemented, functions from the paper by Hiptmair have been used.

The reference pyramid is shown in Figure 10.4. The rather unusual numbering is due to integration with code_Carmel (we will return to this later). Hence, the square base is numbered to approximate to the numbering of the hexahedra. The edges are defined in order of ascending index: e_{ij} where $i < j$, and are orientated from node n_i to n_j .

The faces are also named in order of ascending index: f_{ijk} with $i < j < k$ and f_{1234} for

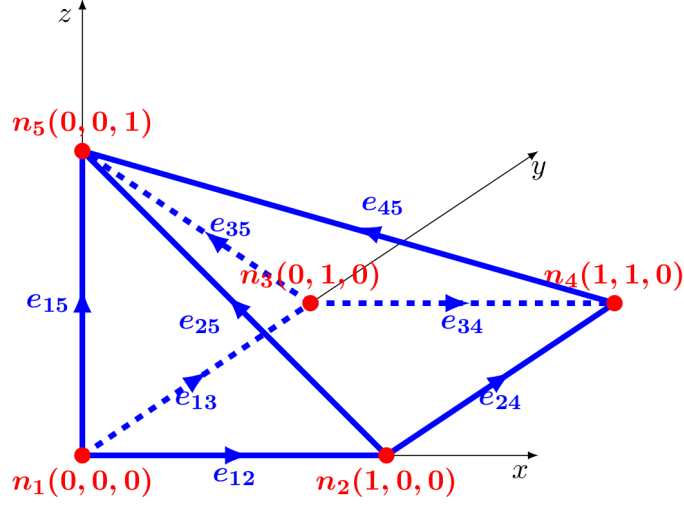


Figure 10.4: Reference pyramid

the square base. To simplify the problem, we will consider that all faces are initially orientated outwards. We will then see how to adjust them according to the rather unusual orientation in Carmel. Figure 10.5 shows the definition of the 5 faces of the reference element.

We will now present the shape functions used.

10.2.2.1 Nodal shape functions

The nodal functions are used to discretise elements belonging to $(H^1(\Omega))^3$. The nodal function associated with a node is 1 on that node, and 0 on all other nodes:

$$\int_{\{n_j\}} w_i^n \cdot \delta_{n_j} = \delta_i^j \quad (10.17)$$

where δ_{n_j} is the Dirac distribution associated with node j , and δ_i^j , the Kronecker symbol. The 5 nodal functions are:

$$\begin{aligned} w_1^n(x, y, z) &= \frac{(1-x-z)(1-y-z)}{1-z}; \\ w_2^n(x, y, z) &= \frac{x(1-y-z)}{1-z}; \\ w_3^n(x, y, z) &= \frac{(1-x-z)y}{1-z}; \\ w_4^n(x, y, z) &= \frac{xy}{1-z}; \\ w_5^n(x, y, z) &= z; \end{aligned}$$

It can be seen that they form a partition of the unit on the element.

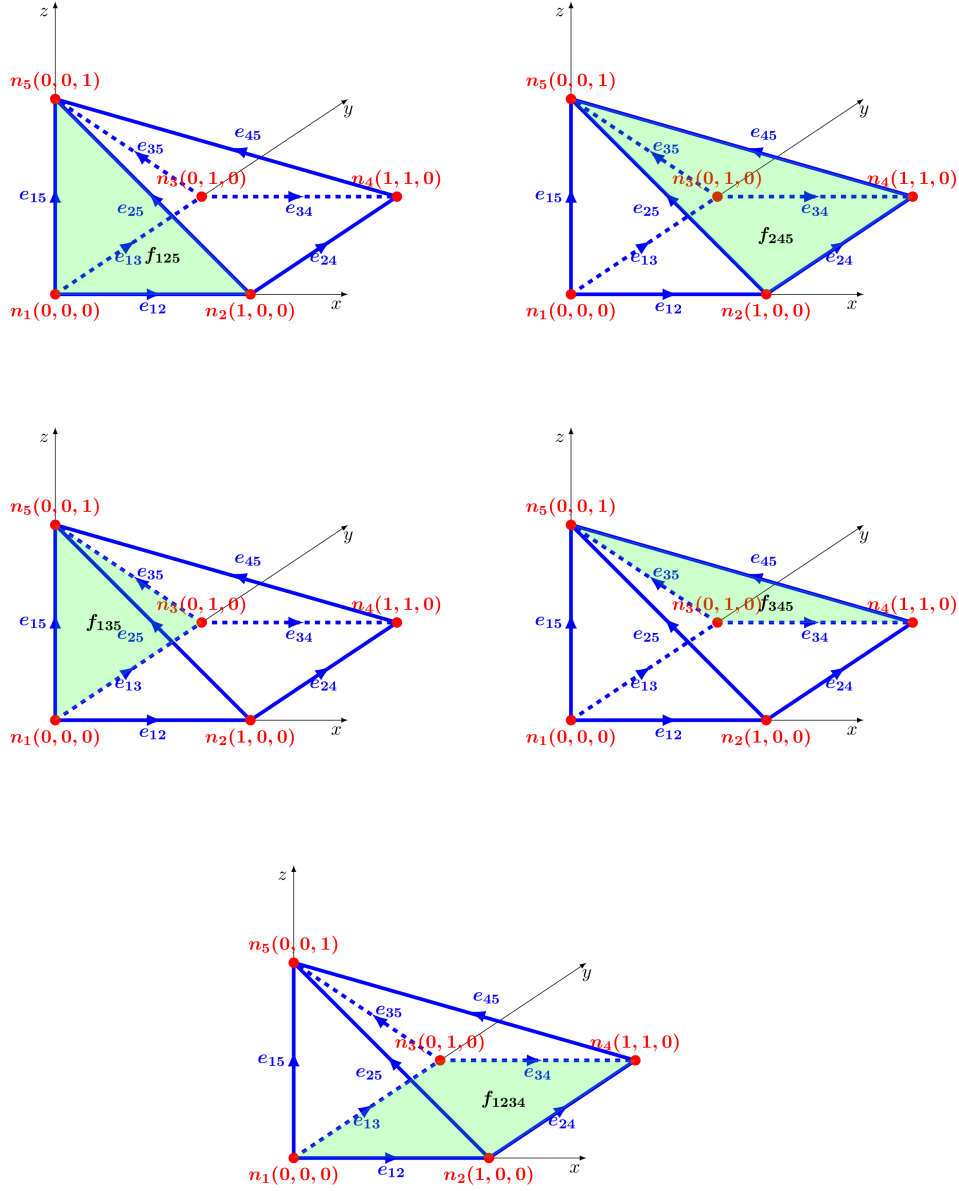


Figure 10.5: Reference pyramid

10.2.2.2 Edge shape functions

The “edge” functions are used to discretise elements belonging to $H(rot, \Omega)$. They are referred to as edge functions because their circulation is equal to 1 on the edge with which they are associated, and 0 otherwise. They thus verify the following property:

$$\int_{e_j} \mathbf{w}_i^e \cdot d\mathbf{l} = \delta_{ij} \quad (10.18)$$

Their expression is detailed in the reference paper (which does, however, contain an error for the component in z on the 6^{eme} and 7^{eme}). They can also be determined using the following Whitney formula [Geuzaine 2001]. Hence, for the function associated with edge \mathbf{w}_{ij}^e , orientated

from i to j , we have:

$$\mathbf{w}_{ij}^e = w_j^n \sum_{r \in \mathcal{N}(j, \bar{i})} \nabla w_r - w_i^n \sum_{r \in \mathcal{N}(i, \bar{j})} \nabla w_r \quad (10.19)$$

where $\mathcal{N}(i, \bar{j})$ are the nodes belonging to the faces that contain node i but not node j . For example, the set $\mathcal{N}(1, \bar{2})$ represents the nodes belonging to facet f_{135} , and thus we have $\mathcal{N}(1, \bar{2}) = \{1, 3, 5\}$. On the other hand, we have $\mathcal{N}(5, \bar{1})$, the indices of nodes belonging to facets f_{245} and f_{345} , from which $\mathcal{N}(5, \bar{1}) = \{2, 3, 4, 5\}$. Finally, the expressions for the edge functions are:

$$\begin{aligned} \mathbf{w}_{12}^e &= \begin{pmatrix} 1 - z - y \\ 0 \\ x - \frac{xy}{1 - z} \end{pmatrix}, & \mathbf{w}_{13}^e &= \begin{pmatrix} 0 \\ 1 - z - x \\ y - \frac{xy}{1 - z} \end{pmatrix}, & \mathbf{w}_{24}^e &= \begin{pmatrix} 0 \\ x \\ \frac{xy}{1 - z} \end{pmatrix}, & \mathbf{w}_{34}^e &= \begin{pmatrix} y \\ 0 \\ \frac{xy}{1 - z} \end{pmatrix} \\ \mathbf{w}_{15}^e &= \begin{pmatrix} z - \frac{yz}{1 - z} \\ z - \frac{xz}{1 - z} \\ 1 - x - y + \frac{xy}{1 - z} - \frac{xyz}{(1 - z)^2} \end{pmatrix}, & \mathbf{w}_{25}^e &= \begin{pmatrix} -z + \frac{yz}{1 - z} \\ \frac{xz}{1 - z} \\ x - \frac{xy}{1 - z} + \frac{xyz}{(1 - z)^2} \end{pmatrix} \\ \mathbf{w}_{35}^e &= \begin{pmatrix} \frac{yz}{1 - z} \\ -z + \frac{xz}{1 - z} \\ y - \frac{xy}{1 - z} + \frac{xyz}{(1 - z)^2} \end{pmatrix}, & \mathbf{w}_{45}^e &= \begin{pmatrix} -\frac{yz}{1 - z} \\ -\frac{xz}{1 - z} \\ \frac{xy}{1 - z} - \frac{xyz}{(1 - z)^2} \end{pmatrix} \end{aligned}$$

10.2.2.3 Facet shape functions

The shape functions associated with the facets are used to discretise the elements of $H(\text{div}, \Omega)$. Their flux is 1 on the facet with which they are associated, and 0 otherwise. They thus verify the following relation:

$$\int_{f_j} \mathbf{w}_i^f \cdot \mathbf{dn} = \delta_{ij} \quad (10.20)$$

This time, the method developed by Hiptmair and the use of Whitney formulas result in different expressions. Both nevertheless appear permissible. In both cases, the facet functions are defined so that their normal is directed towards the outside of the element. When implementing in code `Carmel`, care should be taken to modify their direction according to the orientation defined in the data structure.

10.2.2.3.1 Hiptmair approach

The facet functions presented by Hiptmair are as follows:

$$\begin{aligned}
\mathbf{w}_{125}^f &= \begin{pmatrix} -\frac{xz}{1-z} \\ -2+y+\frac{z}{1-z} \\ z \end{pmatrix}, & \mathbf{w}_{135}^f &= \begin{pmatrix} -2+x+\frac{x}{1-z} \\ -\frac{yz}{1-z} \\ z \end{pmatrix} \\
\mathbf{w}_{245}^f &= \begin{pmatrix} x+\frac{x}{1-z} \\ -\frac{yz}{1-z} \\ z \end{pmatrix}, & \mathbf{w}_{345}^f &= \begin{pmatrix} -\frac{xz}{1-z} \\ y+\frac{y}{1-z} \\ z \end{pmatrix}, & \mathbf{w}_{1234}^f &= \begin{pmatrix} x \\ y \\ z-1 \end{pmatrix}
\end{aligned}$$

10.2.2.3.2 Whitney approach

By analogy with the edge elements, the facet functions can be determined from the nodal functions. Thus, for facet \mathcal{F} consisting of nodes $\{i, j, k\}$ or $\{i, j, k, l\}$, we have:

$$\mathbf{w}_{\mathcal{F}}^f = a \sum_{q \in \mathcal{N}(\mathcal{F})} w_q^n \left(\sum_{r \in \mathcal{N}(\mathcal{F}, q, \overline{q+1})} \nabla w_r \right) \times \left(\sum_{r \in \mathcal{N}(\mathcal{F}, q, \overline{q-1})} \nabla w_r \right) \quad (10.21)$$

a is here a numerical coefficient equal to 2 if the facet contains 3 nodes, and 1 if it contains 4. $\mathcal{N}(\mathcal{F}, q, \overline{q+1})$ are the nodes belonging to the faces that contain the q^{eme} node of facet \mathcal{F} , but not the $(q+1)^{eme}$ (where $q+1$ is the next *cyclique* index). For example, for facet f_{125} made up of nodes $\{1, 2, 5\}$, we will have to calculate the following 3×2 quantities, where q will traverse the elements of f_{125} (the left-hand column corresponds to the terms $\mathcal{N}(f_{125}, q, \overline{q+1})$ while that on the right represents $\mathcal{N}(f_{125}, q, \overline{q-1})$):

$$\mathcal{N}(f_{125}, 1, \overline{2}) = \{1, 3, 5\}, \quad \mathcal{N}(f_{125}, 1, \overline{5}) = \{1, 2, 3, 4\}, \quad \text{avec } q=1$$

$$\mathcal{N}(f_{125}, 2, \overline{5}) = \{1, 2, 3, 4\}, \quad \mathcal{N}(f_{125}, 2, \overline{1}) = \{2, 4, 5\}, \quad \text{avec } q=2$$

$$\mathcal{N}(f_{125}, 5, \overline{1}) = \{2, 3, 4, 5\}, \quad \mathcal{N}(f_{125}, 5, \overline{2}) = \{1, 3, 4, 5\}, \quad \text{avec } q=5$$

The shape functions obtained are as follows:

$$\begin{aligned}
\mathbf{w}_{125}^{f2} &= \begin{pmatrix} \frac{2xz(z+y-1)}{(1-z)^2} \\ \frac{2(z+y-1)}{1-z} \\ -\frac{2z(z+y-1)}{1-z} \end{pmatrix}, & \mathbf{w}_{135}^{f2} &= \begin{pmatrix} \frac{2(z+x-1)}{1-z} \\ \frac{2yz(z+x-1)}{(1-z)^2} \\ -\frac{2z(z+x-1)}{1-z} \end{pmatrix} \\
\mathbf{w}_{245}^{f2} &= \begin{pmatrix} 2x \\ -\frac{2xyz}{(1-z)^2} \\ \frac{2xz}{1-z} \end{pmatrix}, & \mathbf{w}_{345}^{f2} &= \begin{pmatrix} -\frac{2xyz}{(1-z)^2} \\ 2y \\ \frac{2yz}{1-z} \end{pmatrix}, & \mathbf{w}_{1234}^{f2} &= \begin{pmatrix} x \\ y \\ z-1 \end{pmatrix}
\end{aligned}$$

10.2.2.3.3 Comparison of the two types of function

Although in a different form, both types of function are permissible, i.e. they verify equation (10.20). The shape functions from Whitney's formalism cancel out on the opposite facet while those from Hiptmair's paper change their orientation to remain permissible.

To adopt the same approach as used in other versions of Carmel, we will use the functions developed in the reference paper. Moreover, these functions seemed to provide better results. This may be due to the fact that the functions developed by Hiptmair are more regular, and that the error resulting from Gauss integration is thus less.

10.2.3 Transformation of the reference element into a real element (Calculating the integral)

The Gauss quadrature method [Dhatt, Thouzot 1984] is a widely used numerical integration method in which the parameters are determined in such a way as to exactly integrate the polynomials.

If we take a polynomial function $y(\xi)$, we replace the integral of this function with a linear combination of its r values at the integration points ξ_i :

$$\int_{-1}^1 y(\xi) d\xi = w_1 y(\xi_1) + w_2 y(\xi_2) + \dots + w_i y(\xi_i) + \dots + w_r y(\xi_r) \quad (10.22)$$

We seek to determine the $2r$ coefficients (w_i and ξ_i) for the following polynomial:

$$y(\xi) = a_1 + a_2 \xi + \dots + a_{2r} \xi^{2r-1}$$

The reader can follow the development of this calculation on page 281 of [Dhatt, Thouzot 1984]. Above all, it should be remembered that the abscissae ξ_i are also the roots of the Legendre polynomial of order r :

$$P_r(\xi) = 0$$

defined by the recurrence formula:

$$\begin{aligned}
P_0(\xi) &= 1 \\
P_1(\xi) &= \xi \\
\cdots &\quad \cdots \quad \cdots \\
P_k(\xi) &= \frac{2k-1}{k} \xi P_{k-1}(\xi) - \frac{k-1}{k} P_{k-2}(\xi); \quad k = 2, 3, \dots, r
\end{aligned} \tag{10.23}$$

The weights w_i are written:

$$w_i = \frac{2(4 - \xi_i^2)}{[r P_{r-1}(\xi)]^2}; \quad i = 1, 2, \dots, r \tag{10.24}$$

The integration error is of the form:

$$e = \frac{2^{2r+1} (r!)^4}{(2r+1) [(2r)!]^3} \frac{d^{2r} y}{d\xi^{2r}} \tag{10.25}$$

10.2.4 Calculation of elementary integrals by the Gauss method

10.2.4.1 Case of triangles

A direct method of integration consists in writing:

$$\int_0^1 \int_0^{1-\xi} y(\xi, \eta) d\xi d\eta \simeq \sum_{i=1}^r w_i y(\xi_i, \eta_i) \tag{10.26}$$

An interpolation of order 4 with 6 points is used in a reference triangle.

The six Gauss points are:

$$\begin{aligned}
\mathbf{p}_1 &= \begin{pmatrix} a \\ a \\ -1 \end{pmatrix}, \quad \mathbf{p}_2 = \begin{pmatrix} 1-2a \\ a \\ -1 \end{pmatrix}, \quad \mathbf{p}_3 = \begin{pmatrix} a \\ 1-2a \\ -1 \end{pmatrix}, \\
\mathbf{p}_4 &= \begin{pmatrix} b \\ b \\ -1 \end{pmatrix}, \quad \mathbf{p}_5 = \begin{pmatrix} 1-2b \\ b \\ -1 \end{pmatrix}, \quad \mathbf{p}_6 = \begin{pmatrix} b \\ 1-2b \\ -1 \end{pmatrix}
\end{aligned} \tag{10.27}$$

with:

$$\begin{aligned}
a &= 0.445948490915965D0 \\
b &= 0.091576213509771D0
\end{aligned}$$

The two weights used are w_1 for $\mathbf{p}_1, \mathbf{p}_2$ and \mathbf{p}_3 , and w_2 for $\mathbf{p}_4, \mathbf{p}_5$ and \mathbf{p}_6 with:

$$\begin{aligned}
w_1 &= 0.111690794839005D0 \\
w_2 &= 0.054975871827661D0
\end{aligned}$$

10.2.4.2 Case of rectangles

For numerical integration in two dimensions, numerical integration in one dimension is used in each of the directions ξ and η . The “product” method results in:

$$\int_{-1}^1 \int_{-1}^1 y(\xi, \eta) d\xi d\eta = \sum_{i=1}^{r_1} \sum_{j=1}^{r_2} w_i w_j y(\xi_i, \eta_j) \tag{10.28}$$

where:

- w_i, w_j are the coefficients of the integration method;
- ξ_i, η_j are the coordinates of the corresponding integration points.

The Gauss points are defined for an interpolation of order 5 with 7 points in a reference rectangle.

The 7 Gauss points used are the following.

$$\begin{aligned} \mathbf{p}_1 = \begin{pmatrix} 0 \\ 0,5 \\ 0 \end{pmatrix}, \mathbf{p}_2 = \begin{pmatrix} 0 \\ 0,5 \\ a \end{pmatrix}, \mathbf{p}_3 = \begin{pmatrix} 0 \\ 0,5 \\ -a \end{pmatrix}, \mathbf{p}_4 = \begin{pmatrix} 0 \\ \frac{b}{2} + 0,5 \\ b \end{pmatrix}, \\ \mathbf{p}_5 = \begin{pmatrix} 0 \\ \frac{b}{2} + 0,5 \\ -b \end{pmatrix}, \mathbf{p}_6 = \begin{pmatrix} 0 \\ -\frac{b}{2} + 0,5 \\ b \end{pmatrix}, \mathbf{p}_7 = \begin{pmatrix} 0 \\ -\frac{b}{2} + 0,5 \\ -b \end{pmatrix} \end{aligned} \quad (10.29)$$

with:

$$\begin{aligned} a &= \sqrt{\frac{14}{15}} \\ b &= \sqrt{\frac{3}{5}} \end{aligned}$$

The three weights used are p for \mathbf{p}_1 , q for $\mathbf{p}_2, \mathbf{p}_3$, r for $\mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_7$, with:

$$\begin{aligned} p &= \frac{8}{7} \\ q &= \frac{20}{63} \\ r &= \frac{20}{36} \end{aligned}$$

10.2.4.3 Case of tetrahedra

The formula for direct integration on a tetrahedron is given by:

$$\int_0^1 \int_0^{1-\xi} \int_0^{1-\xi-\eta} y(\xi, \eta, \zeta) d\xi d\eta d\zeta = \sum_{i=1}^r w_i y(\xi_i, \eta_i, \zeta_i) \quad (10.30)$$

Three different cases are possible.

In the first case, Gauss points can be defined here for an interpolation of order 3 with 5 points in a reference tetrahedron.

The 5 Gauss points used are:

$$\mathbf{p}_1 = \begin{pmatrix} a \\ a \\ a \end{pmatrix}, \mathbf{p}_2 = \begin{pmatrix} b \\ b \\ b \end{pmatrix}, \mathbf{p}_3 = \begin{pmatrix} b \\ b \\ c \end{pmatrix}, \mathbf{p}_4 = \begin{pmatrix} b \\ c \\ b \end{pmatrix}, \mathbf{p}_5 = \begin{pmatrix} c \\ b \\ b \end{pmatrix} \quad (10.31)$$

with

$$\begin{aligned} a &= \frac{1}{4} \\ b &= \frac{1}{6} \\ c &= \frac{1}{2} \end{aligned}$$

The three weights used are p for \mathbf{p}_1 , and q for $\mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_5$, with:

$$\begin{aligned} p &= -\frac{2}{15} \\ q &= \frac{3}{40} \end{aligned}$$

In the second case, Gauss points can be defined here for an interpolation of order 2 with 4 points in a reference tetrahedron.

The 4 Gauss points used are:

$$\mathbf{p}_1 = \begin{pmatrix} a \\ a \\ a \end{pmatrix}, \mathbf{p}_2 = \begin{pmatrix} a \\ a \\ b \end{pmatrix}, \mathbf{p}_3 = \begin{pmatrix} a \\ b \\ a \end{pmatrix}, \mathbf{p}_4 = \begin{pmatrix} b \\ a \\ a \end{pmatrix} \quad (10.32)$$

with

$$\begin{aligned} a &= \frac{5 - \sqrt{5}}{20} \\ b &= \frac{5 + 3\sqrt{5}}{20} \end{aligned}$$

The four weights used are p for $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4$, with:

$$p = \frac{1}{24}$$

In the third case, Gauss points can be defined here for an interpolation of order 5 with 15 points in a reference tetrahedron.

The 15 Gauss points used are:

$$\begin{aligned} \mathbf{p}_1 &= \begin{pmatrix} a \\ a \\ a \end{pmatrix}, \mathbf{p}_2 = \begin{pmatrix} b_1 \\ b_1 \\ b_1 \end{pmatrix}, \mathbf{p}_3 = \begin{pmatrix} b_1 \\ b_1 \\ c_1 \end{pmatrix}, \mathbf{p}_4 = \begin{pmatrix} b_1 \\ c_1 \\ b_1 \end{pmatrix}, \mathbf{p}_5 = \begin{pmatrix} c_1 \\ b_1 \\ b_1 \end{pmatrix}, \mathbf{p}_6 = \begin{pmatrix} b_2 \\ b_2 \\ b_2 \end{pmatrix} \\ \mathbf{p}_7 &= \begin{pmatrix} b_2 \\ b_2 \\ c_2 \end{pmatrix}, \mathbf{p}_8 = \begin{pmatrix} b_2 \\ c_2 \\ b_2 \end{pmatrix}, \mathbf{p}_9 = \begin{pmatrix} c_2 \\ b_2 \\ b_2 \end{pmatrix}, \mathbf{p}_{10} = \begin{pmatrix} d \\ d \\ e \end{pmatrix}, \mathbf{p}_{11} = \begin{pmatrix} e \\ d \\ d \end{pmatrix}, \mathbf{p}_{12} = \begin{pmatrix} e \\ d \\ d \end{pmatrix} \\ \mathbf{p}_{13} &= \begin{pmatrix} d \\ e \\ e \end{pmatrix}, \mathbf{p}_{14} = \begin{pmatrix} e \\ d \\ e \end{pmatrix}, \mathbf{p}_{15} = \begin{pmatrix} e \\ e \\ d \end{pmatrix} \end{aligned} \quad (10.33)$$

with

$$\begin{aligned} a &= \frac{1}{4} \\ b &= \frac{1}{6} \\ c &= \frac{1}{2} \end{aligned}$$

The three weights used are p for \mathbf{p}_1 , and q for $\mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_5$, with:

$$\begin{aligned} p &= -\frac{2}{15} \\ q &= \frac{3}{40} \end{aligned}$$

10.2.4.4 Case of prisms

The 6 Gauss points used are:

$$\mathbf{p}_1 = \begin{pmatrix} a \\ a \\ b \end{pmatrix}, \mathbf{p}_2 = \begin{pmatrix} a \\ 0 \\ b \end{pmatrix}; \mathbf{p}_3 = \begin{pmatrix} 0 \\ a \\ b \end{pmatrix}, \mathbf{p}_4 = \begin{pmatrix} a \\ a \\ -b \end{pmatrix}, \mathbf{p}_5 = \begin{pmatrix} a \\ 0 \\ -b \end{pmatrix}, \mathbf{p}_6 = \begin{pmatrix} 0 \\ a \\ -b \end{pmatrix} \quad (10.34)$$

with

$$\begin{aligned} a &= \frac{1}{2} \\ b &= \frac{1}{\sqrt{3}} \end{aligned}$$

The weight used is p for $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_5$ and \mathbf{p}_6 , with:

$$p = \frac{1}{6}$$

10.2.4.5 Case of hexahedra

The “product” method is written:

$$\int_{-1}^1 \int_{-1}^1 \int_{-1}^1 y(\xi, \eta, \zeta) d\xi d\eta d\zeta = \sum_{i=1}^{r_1} \sum_{j=1}^{r_2} \sum_{k=1}^{r_3} w_i w_j w_k y(\xi_i, \eta_j, \zeta_k) \quad (10.35)$$

where:

- w_i, w_j, w_k are the coefficients of the integration method;
- ξ_i, η_j and ζ_k are the coordinates of the corresponding integration points.

A direct method consists in writing:

$$\int_{-1}^1 \int_{-1}^1 \int_{-1}^1 y(\xi, \eta, \zeta) d\xi d\eta d\zeta = \sum_{i=1}^{r_1} w_i y(\xi_i, \eta_i, \zeta_i) \quad (10.36)$$

We prefer to use Gauss points here for an interpolation of order 5 with 14 points in a reference hexahedron.

The 14 Gauss points used are:

$$\begin{aligned}
 \mathbf{p}_1 &= \begin{pmatrix} a \\ 0 \\ 0 \end{pmatrix}, \mathbf{p}_2 = \begin{pmatrix} -a \\ 0 \\ 0 \end{pmatrix}, \mathbf{p}_3 = \begin{pmatrix} 0 \\ a \\ 0 \end{pmatrix}, \mathbf{p}_4 = \begin{pmatrix} 0 \\ -a \\ 0 \end{pmatrix}, \mathbf{p}_5 = \begin{pmatrix} 0 \\ 0 \\ a \end{pmatrix}, \\
 \mathbf{p}_6 &= \begin{pmatrix} 0 \\ 0 \\ -a \end{pmatrix}, \mathbf{p}_7 = \begin{pmatrix} b \\ b \\ b \end{pmatrix}, \mathbf{p}_8 = \begin{pmatrix} -b \\ b \\ b \end{pmatrix}, \mathbf{p}_9 = \begin{pmatrix} b \\ -b \\ b \end{pmatrix}, \mathbf{p}_{10} = \begin{pmatrix} b \\ b \\ -b \end{pmatrix}, \\
 \mathbf{p}_{11} &= \begin{pmatrix} -b \\ -b \\ b \end{pmatrix}, \mathbf{p}_{12} = \begin{pmatrix} -b \\ b \\ -b \end{pmatrix}, \mathbf{p}_{13} = \begin{pmatrix} b \\ -b \\ -b \end{pmatrix}, \mathbf{p}_{14} = \begin{pmatrix} b \\ -b \\ b \end{pmatrix} \quad (10.37)
 \end{aligned}$$

with

$$\begin{aligned}
 a &= \sqrt{\frac{19}{30}} \\
 b &= \sqrt{\frac{19}{33}}
 \end{aligned}$$

The weights used are p for $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_5$ and \mathbf{p}_6 , and q for $\mathbf{p}_7, \mathbf{p}_8, \mathbf{p}_9, \mathbf{p}_{10}, \mathbf{p}_{11}, \mathbf{p}_{12}, \mathbf{p}_{13}$ and \mathbf{p}_{14} , with:

$$\begin{aligned}
 p &= \frac{320}{361} \\
 q &= \frac{121}{361}
 \end{aligned}$$

It is also possible to use an interpolation method of order 3 with 6 points in a reference hexahedron.

Remark 10.2.4 *The results with this interpolation method are less precise.*

The 6 Gauss points used are:

$$\mathbf{p}_1 = \begin{pmatrix} a \\ b \\ -c \end{pmatrix}, \mathbf{p}_2 = \begin{pmatrix} a \\ -b \\ -c \end{pmatrix}, \mathbf{p}_3 = \begin{pmatrix} -a \\ b \\ c \end{pmatrix}, \mathbf{p}_4 = \begin{pmatrix} -a \\ -b \\ c \end{pmatrix}, \mathbf{p}_5 = \begin{pmatrix} -d \\ 0 \\ -c \end{pmatrix}, \mathbf{p}_6 = \begin{pmatrix} d \\ 0 \\ c \end{pmatrix} \quad (10.38)$$

with:

$$\begin{aligned}
 a &= \frac{1}{\sqrt{6}} \\
 b &= \frac{1}{\sqrt{2}} \\
 c &= \frac{1}{\sqrt{3}} \\
 d &= \sqrt{\frac{2}{3}}
 \end{aligned}$$

The weight used is p for $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6$ with:

$$p = \frac{4}{3}$$

Finally, the interpolation method of code_Aster with 8 points can be used.

$$\begin{aligned} \mathbf{p}_1 = \begin{pmatrix} a \\ a \\ a \end{pmatrix}, \mathbf{p}_2 = \begin{pmatrix} a \\ a \\ -a \end{pmatrix}, \mathbf{p}_3 = \begin{pmatrix} a \\ -a \\ a \end{pmatrix}, \mathbf{p}_4 = \begin{pmatrix} a \\ -a \\ -a \end{pmatrix}, \\ \mathbf{p}_5 = \begin{pmatrix} -a \\ a \\ a \end{pmatrix}, \mathbf{p}_6 = \begin{pmatrix} -a \\ a \\ -a \end{pmatrix}, \mathbf{p}_7 = \begin{pmatrix} -a \\ -a \\ a \end{pmatrix}, \mathbf{p}_8 = \begin{pmatrix} -a \\ -a \\ -a \end{pmatrix} \end{aligned} \quad (10.39)$$

with:

$$a = \frac{1}{\sqrt{3}}$$

The weight used is p for $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_7, \mathbf{p}_8$, with:

$$p = 1$$

10.2.4.6 Case of pyramids

The Gauss points used come from the spectral version of code_Carmel.

The 8 Gauss points used are:

$$\mathbf{p}_1 = \begin{pmatrix} a_1 \\ a_1 \\ h_1 \end{pmatrix}, \quad \mathbf{p}_2 = \begin{pmatrix} a_1 \\ b_1 \\ h_1 \end{pmatrix}, \quad \mathbf{p}_3 = \begin{pmatrix} b_1 \\ a_1 \\ h_1 \end{pmatrix}, \quad \mathbf{p}_4 = \begin{pmatrix} b_1 \\ b_1 \\ h_1 \end{pmatrix} \quad (10.40)$$

(10.41)

$$\mathbf{p}_5 = \begin{pmatrix} a_2 \\ a_2 \\ h_2 \end{pmatrix}, \quad \mathbf{p}_6 = \begin{pmatrix} a_2 \\ b_2 \\ h_2 \end{pmatrix}, \quad \mathbf{p}_7 = \begin{pmatrix} b_2 \\ a_2 \\ h_2 \end{pmatrix}, \quad \mathbf{p}_8 = \begin{pmatrix} b_2 \\ b_2 \\ h_2 \end{pmatrix} \quad (10.42)$$

with:

$$a_1 = 0.18543444$$

$$b_1 = 0.69205074$$

$$a_2 = 0.09633205$$

$$b_2 = 0.35951611$$

$$h_1 = 0.12251482$$

$$h_2 = 0.54415184$$

The two weights used are w_1 for $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$ and \mathbf{p}_4 , and w_2 for $\mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_7$ and \mathbf{p}_8 , with:

$$w_1 = 0.05813686$$

$$w_2 = 0.02519647$$

We can verify that the sum of the 8 weights is indeed equal to $1/3$, the area of the reference pyramid.

Chapter 11

Taking motion into account

Abstract

The purpose of this chapter is to describe the methods used in code `_Carmel` to take into account the rotational motion of one part in relation to another. Two methods are possible in the time-based version: blocked step and overlapping. A method specific to the spectral version has been implemented. Finally, in the time-based version, it is possible to have a mechanical load and hence a speed resulting from a kinematic equation.

11.1 General principle

To simulate motion in an electromagnetic system (e.g. the motion of the rotor in an electrical machine), when modelling with the finite element method, various numerical strategies or techniques may be considered. To this end, there are two types of description: Eulerian and Lagrangian. The first consists in establishing a fixed baseline from which the various quantities can be observed, while the second follows a moving baseline. Figure 11.1 shows a mesh on which the motion of the red sub-domain is calculated using both types of description.

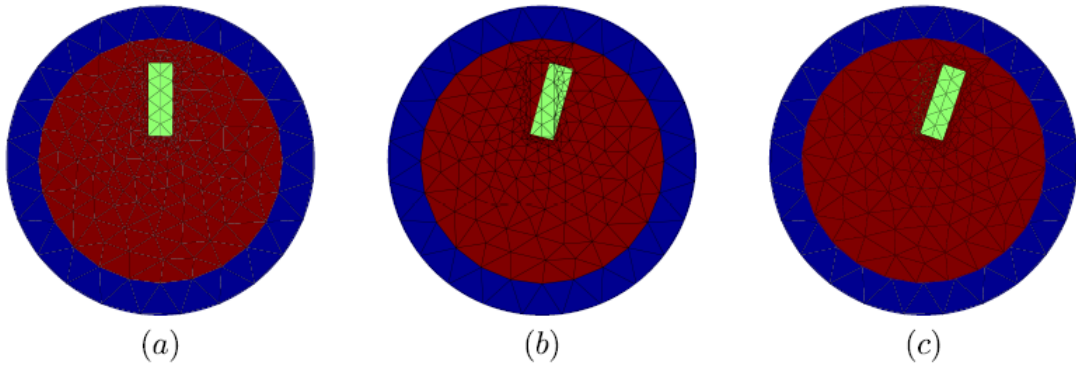


Figure 11.1: Taking motion into account with the Lagrangian and Eulerian descriptions (a): Initial position of the mesh. (b): Rotation of the red sub-domain with the Eulerian description. (c): Rotation of the red sub-domain with the Lagrangian description. The notch is properly discretised, but the elements no longer coincide at the interface between the red and blue sub-domains.

In the case of finite element modelling, the Eulerian approach thus consists in fixing the mesh of the rotor and dragging the various media and fields over time. Although it is attractive because it

does not require re-meshing or change of connectivity, dragging the boundaries between media can be complicated to take into account, as can be seen at the green notch in Figure 11.1 (b). Because the boundaries of the sub-domains no longer correspond to the boundaries of the elements, it can be difficult to take account of the discontinuities of the fields.

By contrast, the Lagrangian approach consists in turning the mesh of the rotor, with respect to that of the stator, in rigid body motion. Hence, the boundaries between media at the rotor and stator are naturally preserved as shown in Figure 11.1 (c). The real issue here is how to connect the rotor mesh with the stator mesh during motion. Finally, Maxwell's equations remain invariant even for non-rectilinear uniform motion with this type of description. It is thus the Lagrangian approach that will be used in what follows.

In this context, there are several methods. They differ in particular in their complexity and their ability or otherwise to take account of any rotational motion. We can nevertheless classify them into two categories, shown in Figure 11.2.

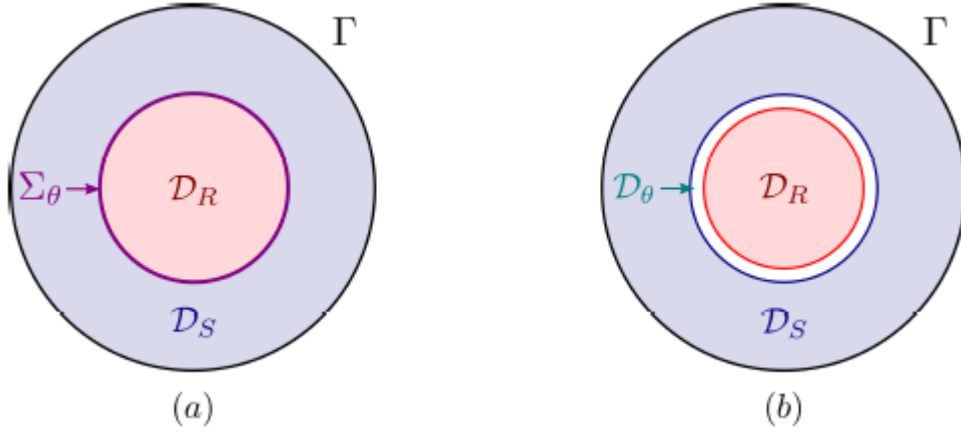


Figure 11.2: Taking motion into account with the Lagrangian description (a): Motion calculated on an interface Σ_θ . (b): Motion calculated in a domain \mathcal{D}_θ .

The first category includes approaches where motion is considered along a 2D linear and 3D surface interface Σ_θ , as shown in Figure 11.2 (a).

The approaches can be classified as follows [Gasmi 1996], [Boukari 2000], [Rapetti 2000]:

- Introduction of a transport term in $\mathbf{v} \wedge \mathbf{B}$ (where \mathbf{v} represents the speed of movement of the moving parts) [Maréchal 1991]. This solution can be used under certain conditions and imposes constraints on the matrix structure of the system to be resolved.
- Modification of the mesh; local re-meshing or mesh deformation in an area incorporating the boundary between the fixed part and the moving part. Some examples include: the blocked step method [Preston et al 1988], [Boualem 1997], the motion strip [Vassent 1990], [Bossavit 1993], [Sadowski 1993], [Ren 1996], and the overlapping method [Tsukerman 1992].
- Coupling the finite element method with another numerical resolution method. In this case, we define a sub-domain incorporating the boundary between the fixed part and the moving part. In the fixed and moving parts, with the exception of the sub-domain reserved for motion, the equations to be solved are discretised using the finite element method. In the sub-domain, we can use the macro-element that consists in searching for an analytical solution in part of the air gap [Féliachi 1981], [Razek et al 1982] or a boundary integral method that brings the space discretisation back to the boundary of the sub-domain, thus allowing coupling with the finite element method [Féliachi 1981], [Goby 1987].

- Recombining the meshes at the interface between the fixed part and the moving part; in this case we are faced with two so-called “non-compliant” meshes at the sliding surface. To recombine the two meshes, we can impose the continuity of the unknown value using interpolation methods [Perrin-Bit 1992], [Dreher et al 1996], [Boukari 2000] or, using the attached elements method (Mortar) [Rapetti et al 2000], [Rapetti 2000], [Antunes et al 2005] or Lagrange operators [Rodger et al 1990]. For the last two methods, recombination of the two meshes is achieved by imposing the continuity of a physical value on the recombination surface.

Among the methods proposed above, the method based on the introduction of a transport term is of limited use. In addition, the use of the macro-element significantly increases the computation time and, like the boundary integrals method, leads to the addition, to the stiffness matrix, of a full matrix that links all the boundary terms. This leads to a greater storage requirement and relatively long computation time [Gasmi 1996], [Boukari 2000].

As such, for code_Carmel the blocked step method and the overlapping method have been adopted.

11.2 Blocked step method

It appears that the first work on the “blocked step” method was presented in 1988 by [Preston et al 1988]. A 3D extension was introduced in 1995 [Kawase et al 1995] and subsequently followed up [Boualem, Piriou 1998], [Boualem 1997], [Boualem, Piriou 1998b].

11.2.1 Mesh layout with the blocked step method

For the blocked step method, we consider two independent meshes $\mathcal{M}_{\mathcal{D}_R}$ and $\mathcal{M}_{\mathcal{D}_S}$ that we will seek to recombine on interface Σ_θ , as shown in Figure 11.2 (a). To do this, it is possible to mesh domain $\mathcal{D} = \mathcal{D}_R \cup \mathcal{D}_S$ normally and virtually duplicate the N_θ unknowns located on $\Sigma_\theta = \mathcal{D}_R \cap \mathcal{D}_S$.

Finally, this method requires that the mesh should be *réglé* on Σ_θ . This means that there is a periodic structure of the mesh on Σ_θ by angle rotation $\Delta\theta$ as shown in Figure 11.3 on a sample 2D mesh.

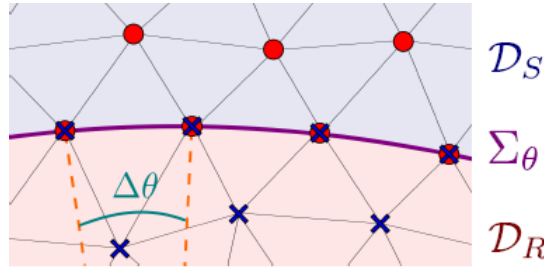


Figure 11.3: Mesh layout with the blocked step method. The rotor unknowns (blue cross) are virtually duplicated on Σ_θ

Intuitively, we understand that with this layout it will be possible to take account of the angle rotations $\theta_k = k \Delta\theta$ with $k \in \mathbb{Z}$ by permutation of unknowns along Σ_θ .

11.2.2 Finite element problem on \mathcal{D}_R et \mathcal{D}_S

Having virtually duplicated the unknowns on interface Σ_θ , the linear magnetostatic problem is written *indépendamment* on \mathcal{D}_R and \mathcal{D}_S as:

$$\begin{pmatrix} M_{rr}^R & 0 \\ 0 & M_{rr}^S \end{pmatrix} \begin{pmatrix} \mathbf{X}^R \\ \mathbf{X}^S \end{pmatrix} = \begin{pmatrix} \mathbf{F}^R \\ \mathbf{F}^S \end{pmatrix} \quad (11.1)$$

where the index R or S denotes the quantities defined on the rotor and stator mesh respectively. At this point, the problem is not properly set out, as the unknowns are virtually duplicated on Σ_θ . Hence, the system of equations 11.1 is not invertible. However, the motion equation will lead to a well-posed problem, while also taking the rotation into account.

11.2.3 Motion equation for \mathcal{D}_R et \mathcal{D}_S

This means finding the bijection linking $\mathbf{X}_\Sigma^R \in \mathbb{R}^{N_\theta}$ to $\mathbf{X}_\Sigma^S \in \mathbb{R}^{N_\theta}$ during motion, where these two vectors represent the components of \mathbf{X}^R and \mathbf{X}^S respectively, whose corresponding unknowns belong to Σ_θ . In the initial stage we can assume that:

$$\mathbf{X}_\Sigma^R = \mathbf{X}_\Sigma^S \quad (11.2)$$

Because of the periodic structure, there is a **permutation matrix** $\mathbf{R}(\theta_k) \in \mathbb{R}^{N_\theta \times N_\theta}$ that represents the angle rotation θ_k by:

$$\mathbf{X}_\Sigma^R = \mathbf{R}(\theta_k) \mathbf{X}_\Sigma^S \quad (11.3)$$

Matrix $\mathbf{R}(\theta_k)$ is obtained directly from the unit permutation matrix $\mathbf{P} = \mathbf{R}(\Delta_\theta)$ which allows permutation of the indices of the unknowns after an angle rotation $\theta = \Delta_\theta$. We thus have:

$$\mathbf{R}(\theta_k) = \mathbf{P}^{k-1} \quad (11.4)$$

where $\mathbf{R}(\theta_k)$ verifies:

$$\mathbf{R}(\theta_0) = \mathbf{R}(\theta_{N_\theta}) = I_{N_\theta} \quad (11.5)$$

where $I_{N_\theta} \in \mathbb{R}^{N_\theta \times N_\theta}$ is the identity matrix of size N_θ .

11.2.4 Notation of the total system with the blocked step method

We have yet to take advantage of motion equation 11.3 to properly set out problem 11.1. This means *eliminer* the N_θ virtual unknowns on Σ_θ . To do this, we introduce rectangular matrix $\mathbf{T}(\theta_k)$:

$$\mathbf{T}(\theta_k) = \begin{pmatrix} \mathbf{I} & 0 & 0 \\ 0 & \mathbf{I} & 0 \\ 0 & 0 & \mathbf{I} \\ 0 & \mathbf{R}(\theta_k) & 0 \end{pmatrix} \quad (11.6)$$

The equation to eliminate the N_θ unknowns according to 11.3 is:

$$\begin{pmatrix} \mathbf{X}^R \\ \mathbf{X}^S \end{pmatrix} = \begin{pmatrix} \mathbf{X}_D^R \\ \mathbf{X}_\Sigma^R \\ \mathbf{X}_D^S \\ \mathbf{X}_\Sigma^S \end{pmatrix} = \mathbf{T}(\theta_k) \begin{pmatrix} \mathbf{X}_D^R \\ \mathbf{X}_\Sigma^R \\ \mathbf{X}_D^S \\ \mathbf{X}_D^S \end{pmatrix} \quad (11.7)$$

where \mathbf{X}_D^R and \mathbf{X}_D^S represent the unknowns of the mesh at the rotor and stator respectively, and which do not belong to Σ_θ . By replacing the expression of the unknown vector in the initial system, we have:

$$\begin{pmatrix} \mathbf{M}_{rr}^R & 0 \\ 0 & \mathbf{M}_{rr}^S \end{pmatrix} \mathbf{T}(\theta_k) \begin{pmatrix} \mathbf{X}_D^R \\ \mathbf{X}_\Sigma^R \\ \mathbf{X}_D^S \end{pmatrix} = \begin{pmatrix} \mathbf{F}_D^R \\ 0 \\ \mathbf{F}_D^S \\ 0 \end{pmatrix} \quad (11.8)$$

The unusual form of the second term is due to the fact that the field sources are not in contact with Σ_θ . As the previous system has N_θ more equations than there are unknowns, it is then a question of eliminating the equations by summing the contributions on Σ_θ . In practice, this is simply done by multiplying system 11.8 by $\mathbf{T}^t(\theta_k)$ [Antunes et al 2006]. Hence, the square of the total system is written:

$$\mathbf{T}^t(\theta_k) \begin{pmatrix} \mathbf{M}_{rr}^R & 0 \\ 0 & \mathbf{M}_{rr}^S \end{pmatrix} \mathbf{T}(\theta_k) \begin{pmatrix} \mathbf{X}_D^R \\ \mathbf{X}_\Sigma^R \\ \mathbf{X}_D^S \end{pmatrix} = \mathbf{T}^t(\theta_k) \begin{pmatrix} \mathbf{F}_D^R \\ 0 \\ \mathbf{F}_D^S \\ 0 \end{pmatrix} \quad (11.9)$$

By denoting \mathbf{X} the new unknown vector, we can show that the previous system is reduced to:

$$(\mathbf{M}_{rr} + \mathbf{M}_{pf}(\theta_k)) \mathbf{X} = \mathbf{F} \quad (11.10)$$

Here \mathbf{M}_{rr} is the invariant part by angle rotation θ_k . In practical terms, it represents the interactions resulting from elements that do not touch Σ_θ . By contrast, $\mathbf{M}_{pf}(\theta_k)$ is the matrix that varies with each rotation. However, the latter has a low number of non-zero terms because it is derived from the assembly of elements adjacent to Σ_θ . Finally, since the source vector is not adjacent to Σ_θ , $\mathbf{F} = \mathbf{T}^t(\theta_k) (\mathbf{F}_D^R; 0; \mathbf{F}_D^S; 0)$ defined in 11.9 does not depend on θ .

11.2.5 Conclusion

In the case of the blocked step method, permutation of the nodal unknowns is applied at the slipping surface. For the vector potential formulation, it is the circulation unknowns on the edges that undergo permutation.

This change is made in the connectivity table. The periodicity or anti-periodicity conditions are provided by the unknowns located at the ends of the slipping surface. To perform relative motion, we correct only the connectivity in the moving elements that touch the slipping surface.

The advantage of this method is that it has a mesh that is always compliant. It is easy to implement and the properties of the finite elements are preserved. As a result, taking account of motion does not introduce a new numerical error. As a result, when making comparisons, this method is often considered the benchmark for assessing the quality of the solution.

However, the main drawback is the constraint on the motion step, which must correspond to the mesh step.

The blocked step method is implemented in code_Carmel (time-based version) and is used to model rotating machines with the vector potential or scalar potential formulation.

11.3 Overlapping method

This method proposed by [Tsukerman 1992] was originally developed for 2D modelling with the vector potential formulation. It was then further developed to be applied in 2D to electrical machines [Biddlecombe et al 1988], [Lepaul et al 1999].

To simplify presentation of the method, we use a 2D model of the cross-section of a machine. The principle remains the same in 3D, and the explicit calculation of 3D shape functions can be found in the annex [D](#).

11.3.1 Mesh layout with the overlapping method

To apply the overlapping method, the meshes of the rotor $\mathcal{M}_{\mathcal{D}_R}$ and the stator $\mathcal{M}_{\mathcal{D}_S}$ must be separated by a thin unmeshed layer \mathcal{D}_θ as shown in Figure [11.2](#).

We denote Σ_θ^R and Σ_θ^S the respective interfaces of \mathcal{D}_R and \mathcal{D}_S along \mathcal{D}_θ . In addition, we assume that the meshes on Σ_θ^R and Σ_θ^S are made up of regular quadrangles with the same periodic structure as θ . Thus, it can be assumed that the mesh of Σ_θ^S in the initial state is obtained by a normal projection of the mesh of \mathcal{D}_R . While the overlapping method can be applied to non-regular meshes, this assumption simplifies the method and its cost in computation time, making it compatible with the reduction of models.

11.3.2 Extension of nodal shape functions to \mathcal{D}_θ

The principle of the method is first to extend the nodal functions of $\mathbf{W}^0(\mathcal{D}_h)$ to the unmeshed domain \mathcal{D}_θ and to do so continuously. To this end, the support of the nodal functions of the stator (associated with the unknowns belonging to Σ_θ^S) is extended by normal projection on Σ_θ^R , as shown in Figure [11.4](#) (b) for a 2D example.

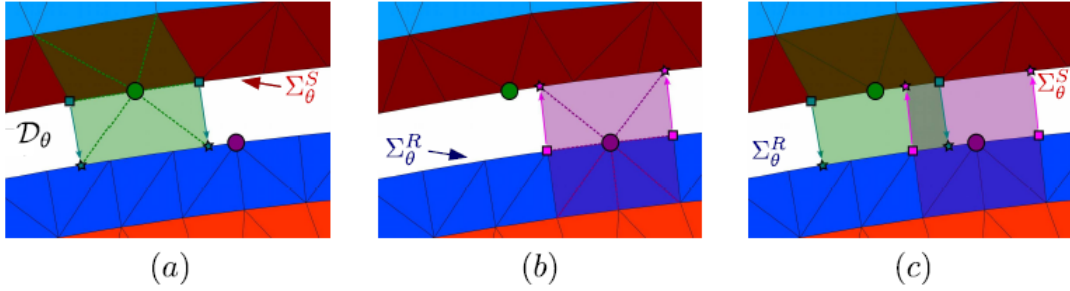


Figure 11.4: Overlapping interaction. (a): stator nodal function extended to Γ_θ^R . (b): rotor nodal function extended to Γ_θ^S . (c): interaction between the two nodal functions.

Similarly, the support of the rotor nodal functions is extended to \mathcal{D}_θ as shown in Figure [11.4](#) (b). Finally, Figure [11.4](#) (c) shows that there is an area where the two nodal functions overlap in \mathcal{D}_θ . This represents the interaction of one stator edge with two rotor edges. Figure [11.5](#) shows that two integration zones can be defined: one *gauche* and the other *droite*.

11.3.3 Overlapping reference element

To calculate quantities on both zones, we introduce two reference elements presented in Figures [11.6](#) and [11.7](#). These are two quadrangles with “legs” whose length depends on the values a , b , c and d defined in Figures [11.6](#) and [11.7](#).

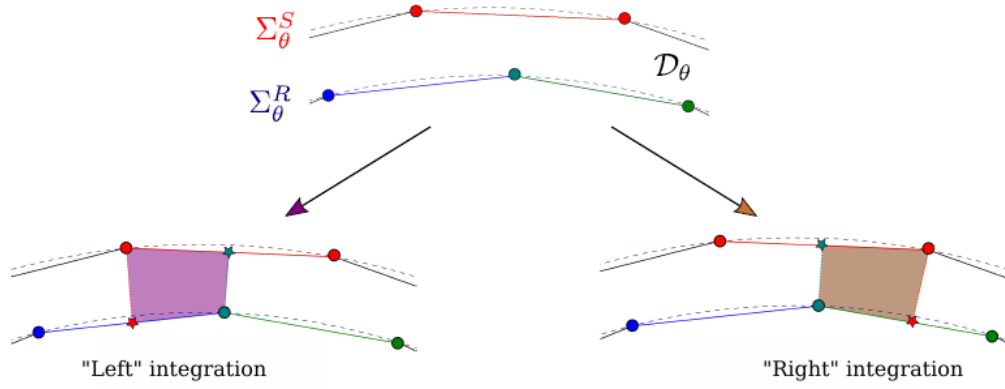


Figure 11.5: Left and right integration zones linked to the stator edge (red). The rings represent mesh nodes, while the stars are fictional nodes, obtained by normal projection of real nodes on the opposite edge. The latter are used only to define the integration zone and are not unknowns in the problem.

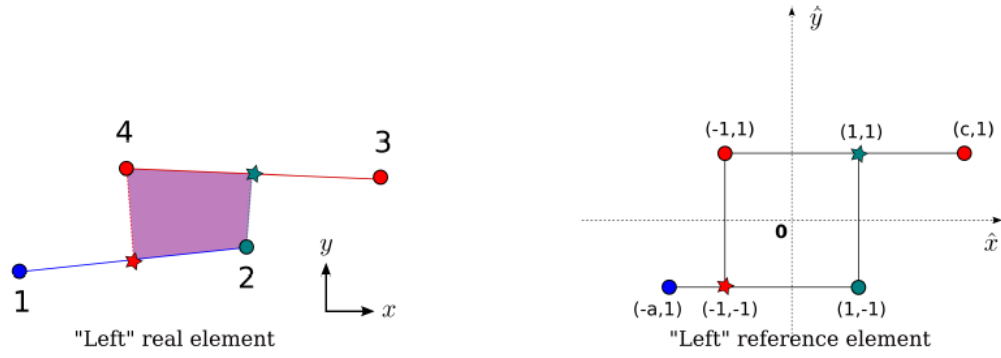


Figure 11.6: Real element and reference element for the left integration zone

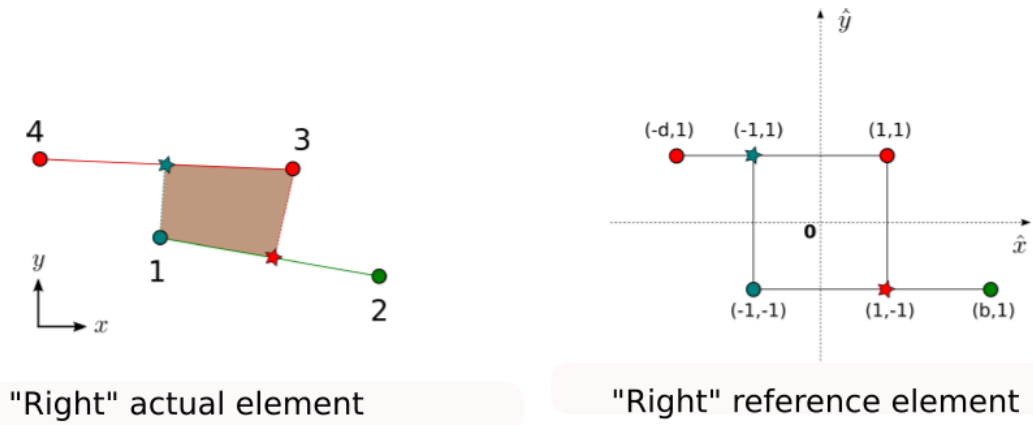


Figure 11.7: Real element and reference element for the right integration zone

We can thus define a generic reference element that is shown in Figure 11.8. If $a = c = 1$, then the right element can be found, while $b = d = 1$ describes the left element.

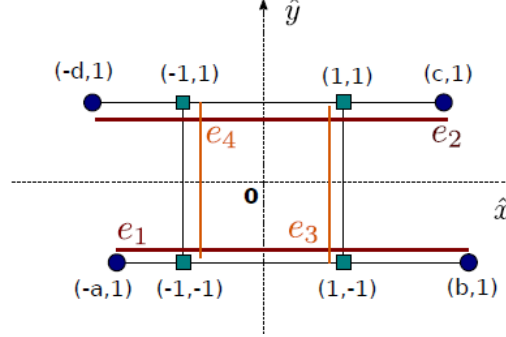


Figure 11.8: Generic reference element

The extension of this reference element to 3D, along with the shape functions, is presented in annex D. In practice, it is a hexahedron with “surface legs” analogous to the linear legs for the 2D reference element.

11.3.4 Dealing with edge unknowns

For the unknowns, the reference element in Figure 11.8 shows that integration terms can be calculated using the four real nodes, without introducing an additional unknown node. However, the same does not apply to the edge unknowns, required for 3D applications. Indeed, on Figure D we can see the two vertical edges e_3 and e_4 . These connect \mathcal{D}_R to \mathcal{D}_S on \mathcal{D}_θ . At first glance, therefore, unknowns would have to be associated with these edges. Hence, the principle of overlapping, which is to avoid creating additional unknowns on \mathcal{D}_θ , is no longer verified. Fortunately, use of the tree gauge avoids contradiction of this principle. There is an infinite number of vectors \mathbf{A} such that $\mathbf{B} = \text{rot } \mathbf{A}$.

From a numerical point of view, this means that the problem is under-determined and hence a number of unknowns can be eliminated. To this end, it is possible to eliminate the edges associated with a tree spanning the mesh [Le Menach 1999], i.e. traversing all the mesh nodes and not closing in on itself. There will thus be no unknowns associated with edges e_3 and e_4 linking \mathcal{D}_R to \mathcal{D}_S . This ultimately allows modelling motion without adding an additional unknown.

11.3.5 Notation of the total system with the overlapping method

It is recalled that the approach is presented in 2D, but can be directly applied in 3D if the two surface meshes of the rotor and the stator are composed of coincident regular quadrangles. Element overlapping in 3D as well as the nodal and edge shape functions are presented in annex D.

To conclude, we can summarise the overlapping approach in two steps:

- Determination of the overlapping reference elements: for each rotor position, we must determine the two rotor edges that will interact with each edge of the stator, and the four values a , b , c and d . In practice, if the mesh between the two interfaces Σ_θ^R and Σ_θ^S is periodic and coincident, this task can be performed on only one edge of the stator (which interacts with only two rotor edges). The periodic structure on Σ_θ^R and Σ_θ^S implies that the left and right reference elements will be identical on \mathcal{D}_θ .
- Assembly of finite element matrices on \mathcal{D}_θ . As \mathcal{D}_θ is in the air gap, consisting only of air, the only terms to be calculated are those of the Rot-Rot matrix. Thus, we define the positive semi-defined symmetric matrix $\mathbf{M}_{ovl}(\theta) \in \mathbb{R}^{N_a \times N_a}$ such that:

$$(\mathbf{M}_{ovl}(\theta))_{i,j} = \int_{\mathcal{D}_\theta} (\nu_0 \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_j^1) d\mathcal{D}_\theta \quad (11.11)$$

In practice, we calculate this expression by assembling the elementary matrices in each overlapping element. According to the previous point, the left and right elements are identical on θ . Thus, only one “left” and one “right” elementary matrix need to be calculated, which we will assemble globally on \mathcal{D}_θ in order to calculate $\mathbf{M}_{ovl}(\theta)$.

In the case of a linear magnetostatic problem, the final system is written:

$$\left[\begin{pmatrix} \mathbf{M}_{rr}^R & 0 \\ 0 & \mathbf{M}_{rr}^S \end{pmatrix} + \mathbf{M}_{ovl}(\theta) \right] \begin{pmatrix} \mathbf{X}^R \\ \mathbf{X}^S \end{pmatrix} = \begin{pmatrix} \mathbf{F}^R \\ \mathbf{F}^S \end{pmatrix} \quad (11.12)$$

where \mathbf{X}^R and \mathbf{X}^S are the unknowns on \mathcal{D}_R and \mathcal{D}_S respectively. Unlike the blocked step, the total system is not projected onto the interface. It suffices to add matrix $\mathbf{M}_{ovl}(\theta)$ to the original system, allowing the two sub-domains to be coupled, thus modelling the motion of the rotor.

The Gauss integration technique adapted to the shape functions in the Overlapping method is to be found in annex D.

11.4 Specific method for the spectral version

11.4.1 Principle of the blocked step

In general, methods that take account of motion can be seen as applications that link the unknowns of the fixed domain to those of the moving domain. These applications are usually reflected in the form of a transformation matrix $\mathbf{M}(t)$. To explain matrix $\mathbf{M}(t)$ of the blocked step, we take the 2D case where the spatial unknowns are on the vertices of the mesh (as shown in Figure 11.9). The unknowns on the moving part are denoted i_j^m and those on the fixed part i_j^f . At the initial

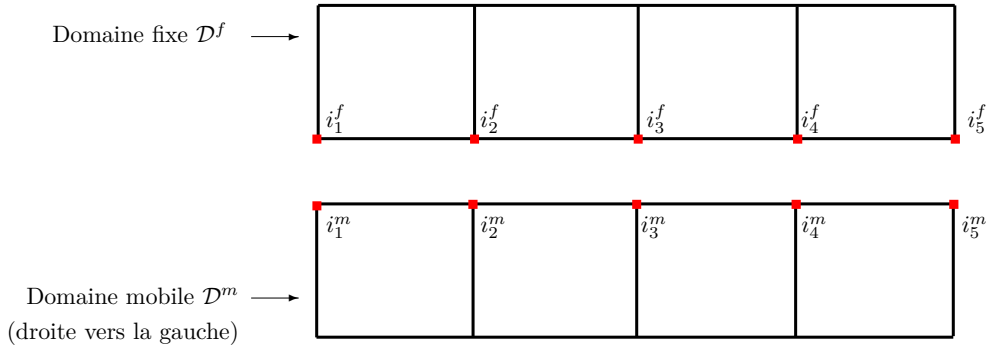


Figure 11.9: *Example of a compliant mesh for the blocked step.*

time t_0 , the unknowns on \mathcal{D}^f are related to those of \mathcal{D}^m by:

$$\begin{bmatrix} i_1^f \\ i_2^f \\ i_3^f \\ i_4^f \\ i_5^f \end{bmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}}_{\mathbf{M}(t_0)} \begin{bmatrix} i_1^m \\ i_2^m \\ i_3^m \\ i_4^m \\ i_5^m \end{bmatrix}$$

The stepwise motion of the moving part (one square of the grid) can be represented by matrix $\mathbf{M}(t_1)$ which links the moving unknowns to those of \mathcal{D}^f as follows:

$$\begin{bmatrix} i_1^f \\ i_2^f \\ i_3^f \\ i_4^f \\ i_5^f \end{bmatrix} = \underbrace{\begin{pmatrix} 0 \rightarrow & 1 & 0 & 0 & 0 \\ 0 & 0 \rightarrow & 1 & 0 & 0 \\ 0 & 0 & 0 \rightarrow & 1 & 0 \\ 0 & 0 & 0 & 0 \rightarrow & 1 \\ \rightarrow 1 & 0 & 0 & 0 & 0 \end{pmatrix}}_{\mathbf{M}(t_1)} = \begin{bmatrix} i_1^m \\ i_2^m \\ i_3^m \\ i_4^m \\ i_5^m \end{bmatrix}$$

This is an operation to shift by one column in the diagonal. In the same way, at $t = t_2$ we have:

$$\begin{bmatrix} i_1^f \\ i_2^f \\ i_3^f \\ i_4^f \\ i_5^f \end{bmatrix} = \underbrace{\begin{pmatrix} 0 & 0 \rightarrow & 1 & 0 & 0 \\ 0 & 0 & 0 \rightarrow & 1 & 0 \\ 0 & 0 & 0 & 0 \rightarrow & 1 \\ \rightarrow 1 & 0 & 0 & 0 & 0 \\ 0 & \rightarrow 1 & 0 & 0 & 0 \end{pmatrix}}_{\mathbf{M}(t_2)} = \begin{bmatrix} i_1^m \\ i_2^m \\ i_3^m \\ i_4^m \\ i_5^m \end{bmatrix}$$

This illustration shows that the move from position t_0 to position t_k is accomplished by permutation, on line i of matrix $\mathbf{M}(t)$, the i -th value (which is 1) and the value 0 located at position $j = k - [k/n] * n$ (k modulo n), where n is the size of matrix $\mathbf{M}(t)$.

11.4.2 Spectral representation of motion

Since the motion matrix is dependent on time t , it can be developed on the spectral basis \mathcal{C} in the following form:

$$\mathbf{M}(t) = \sum_i \mathbf{M}_i \psi_i(t)$$

By orthonormal projection, we show that the spectral matrices \mathbf{M}_i are given by:

$$\mathbf{M}_i = \int_{\mathcal{T}} \mathbf{M}(t) \psi_i(t) w(t) dt$$

These integrals are estimated using the modified quadrature described in annex Q. For each quadrature point t_q there is a corresponding position q of the motion, for which matrix $\mathbf{M}(t_q)$ is calculated with a permutation equal to $q\Delta h$, where Δh is the blocked step given by the mesh. To deduce the tensor form of the system of equations in the presence of motion, we take the discrete weak form 14.9. We introduce assembled matrices on the moving and fixed sub-domains by writing:

$$\mathbf{R}_i = \mathbf{R}_i^f + \mathbf{M}(t) \mathbf{R}_i^m$$

System 14.9 is thus rewritten:

$$\left\{ \begin{array}{l} \left(\int_{\mathcal{T}_w} \psi_s \psi_p dt \right) \left\{ \mathbf{R}_1^f \mathbf{A}_s + \mathbf{R}_2^f \mathbf{A}_s^\partial + \mathbf{R}_3^f \boldsymbol{\varphi}_s \right\} + \left(\int_{\mathcal{T}_w} \psi_s \psi_p \mathbf{M} dt \right) \left\{ \mathbf{R}_1^m \mathbf{A}_s + \mathbf{R}_2^m \mathbf{A}_s^\partial + \mathbf{R}_3^m \boldsymbol{\varphi}_s \right\} = \\ \left(\int_{\mathcal{T}_w} \psi_s \psi_p dt \right) \left\{ \mathbf{L}_1 \mathbf{J} \mathbf{f}_s^0 + \mathbf{L}_2 \mathbf{H} \mathbf{f}_s^\Gamma \right\} - \left(\int_{\mathcal{T}_w} \psi_p \mathbf{K}(\mathbf{A}) dt \right) \\ \left(\int_{\mathcal{T}_w} \psi_s \psi_p dt \right) \left\{ \mathbf{R}_3^f \mathbf{A}_s^\partial + \mathbf{R}_4^f \boldsymbol{\varphi}_s \right\} + \left(\int_{\mathcal{T}_w} \psi_s \psi_p \mathbf{M} dt \right) \left\{ \mathbf{R}_3^m \mathbf{A}_s^\partial + \mathbf{R}_4^m \boldsymbol{\varphi}_s \right\} = \\ \left(\int_{\mathcal{T}_w} \psi_s \psi_p dt \right) \left\{ \mathbf{L}_3 \mathbf{J} \mathbf{f}_s^\Gamma \right\} \end{array} \right.$$

Note that it has been assumed with this notation that the source terms, boundary conditions and non-linearities do not exist in the motion strip (no matrix $\mathbf{M}(t)$ in the second term). This assumption is by no means restrictive, but we have not found any applications where this is the case.

By introducing matrices \mathbf{S} and the differentiation matrix and its pseudo-inverse $\widehat{\mathbf{D}}$ and defining the matrix $\check{\mathbf{M}}$ as a matrix of $N^t \times N^t$ blocks of size $(N^{mvt} \times N^{mvt})$, where N^{mvt} is the number of degrees of freedom on the motion interface. Block (s, p) of matrix $\check{\mathbf{M}}$ is written:

$$\check{\mathbf{M}}_{sp} = \sum_q \mathbf{M}_q \int_{\mathcal{T}_w} \psi_s(t) \psi_p(t) \psi_q(t) dt \quad (11.13)$$

We obtain the system of equations describing the problem with motion, for all $1 \leq s, p \leq N^t$:

$$\begin{cases} S_{sp} \left\{ \mathbf{R}_1^f \mathbf{A}_s + \mathbf{R}_2^f \mathbf{A}_s^\partial + \mathbf{R}_3^f \boldsymbol{\varphi}_s \right\} + \check{\mathbf{M}}_{sp} \left\{ \mathbf{R}_1^m \mathbf{A}_s + \mathbf{R}_2^m \mathbf{A}_s^\partial + \mathbf{R}_3^m \boldsymbol{\varphi}_s \right\} = S_{sp} \left\{ \mathbf{L}_1 \mathbf{J} \mathbf{f}_s^0 + \mathbf{L}_2 \mathbf{H} \mathbf{f}_s^\Gamma \right\} \\ \quad - \left(\int_{\mathcal{T}_w} \psi_p \mathbf{K}(\mathbf{A}) dt \right) \\ S_{sp} \left\{ \mathbf{R}_3^f \mathbf{A}_s^\partial + \mathbf{R}_4^f \boldsymbol{\varphi}_s \right\} + \check{\mathbf{M}}_{sp} \left\{ \mathbf{R}_3^m \mathbf{A}_s^\partial + \mathbf{R}_4^m \boldsymbol{\varphi}_s \right\} = S_{sp} \left\{ \mathbf{L}_3 \mathbf{J} \mathbf{f}_s^\Gamma \right\} \end{cases}$$

and further:

$$\begin{cases} \mathbf{S} \otimes \left\{ \mathbf{R}_1^f \mathbf{A} + \mathbf{R}_2^f \mathbf{A}^\partial + \mathbf{R}_3^f \boldsymbol{\varphi} \right\} + \check{\mathbf{M}} \circ \left\{ \mathbf{R}_1^m \mathbf{A} + \mathbf{R}_2^m \mathbf{A}^\partial + \mathbf{R}_3^m \boldsymbol{\varphi} \right\} = \mathbf{S} \otimes \left\{ \mathbf{L}_1 \mathbf{J} \mathbf{f}^0 + \mathbf{L}_2 \mathbf{H} \mathbf{f}^\Gamma \right\} \\ \quad - \left(\int_{\mathcal{T}_w} \psi_p \mathbf{K}(\mathbf{A}) dt \right) \\ \mathbf{S} \otimes \left\{ \mathbf{R}_3^f \mathbf{A}^\partial + \mathbf{R}_4^f \boldsymbol{\varphi} \right\} + \check{\mathbf{M}} \circ \left\{ \mathbf{R}_3^m \mathbf{A}^\partial + \mathbf{R}_4^m \boldsymbol{\varphi} \right\} = \mathbf{S} \otimes \left\{ \mathbf{L}_3 \mathbf{J} \mathbf{f}^\Gamma \right\} \end{cases}$$

where \circ is the Hadamard product for each block (see annex **P**). We substitute the differentiated vector \mathbf{A}^∂ by $(\mathbf{D} \otimes \mathbf{I}_{n_1}) \mathbf{A}$ to write:

$$\begin{cases} \left[(\mathbf{S} \otimes \mathbf{R}_1^f) \mathbf{A} + (\mathbf{S} \otimes \mathbf{R}_2^f) (\mathbf{D} \otimes \mathbf{I}_{n_1}) \mathbf{A} + (\mathbf{S} \otimes \mathbf{R}_3^f) \boldsymbol{\varphi} \right] + \\ \left[(\check{\mathbf{M}} \circ \mathbf{R}_1^m) \mathbf{A} + (\check{\mathbf{M}} \circ \mathbf{R}_2^m) (\mathbf{D} \otimes \mathbf{I}_{n_1}) \mathbf{A} + (\check{\mathbf{M}} \circ \mathbf{R}_3^m) \boldsymbol{\varphi} \right] = \mathbf{S} \otimes \left\{ \mathbf{L}_1 \mathbf{J} \mathbf{f}^0 + \mathbf{L}_2 \mathbf{H} \mathbf{f}^\Gamma \right\} - \\ \left(\int_{\mathcal{T}_w} \psi_p \mathbf{K}(\mathbf{A}) dt \right) \\ (\mathbf{S} \otimes \mathbf{R}_3^f) \mathbf{A} + (\mathbf{S} \widehat{\mathbf{D}} \otimes \mathbf{R}_4^f) \boldsymbol{\varphi} + (\check{\mathbf{M}} \circ \mathbf{R}_3^m) \mathbf{A} + (\check{\mathbf{M}} \circ \mathbf{R}_4^m) (\widehat{\mathbf{D}} \otimes \mathbf{I}_{n_1}) \boldsymbol{\varphi} = \\ \mathbf{S} \widehat{\mathbf{D}} \otimes \left\{ \mathbf{L}_3 \mathbf{J} \mathbf{f}^\Gamma \right\} \end{cases}$$

By simple calculation, we show:

$$(\check{\mathbf{M}} \circ \mathbf{R})(\widehat{\mathbf{D}} \otimes \mathbf{I}) = \left[\check{\mathbf{M}} (\widehat{\mathbf{D}} \otimes \mathbf{I}) \right] \circ \mathbf{R} \quad (11.14)$$

$$= \left[\sum_{q=1}^{N^t} (\mathbf{E}^q \widehat{\mathbf{D}}) \otimes \mathbf{M}_q \right] \circ \mathbf{R} \quad (11.15)$$

where \mathbf{E}^q is the expected value matrix of size $N^t \times N^t$:

$$(\mathbf{E}^q)_{ij} = \int_{\mathcal{T}} \psi_i(t) \psi_j(t) \psi_q(t) w(t) dt \quad (11.16)$$

From the point of view of mathematical formalism, we adopt the notation of (11.14) as it clearly distinguishes the different dimensions of the problem (spatial, spectral and motion). However, at this stage of general formalisation, we will keep in mind the notation of (11.15) as it can be used to design special spectral bases (doubly orthogonal bases, analytical expressions).

Finally, the system of equations of the magnetodynamic problem with motion is given by (recalling that $\mathbf{S} = \mathbf{I}$):

$$\left\{ \begin{array}{l} (\mathbf{I} \otimes \mathbf{R}_1^f) \mathbf{A} + (\mathbf{D} \otimes \mathbf{R}_2^f) \mathbf{A} + (\mathbf{I} \otimes \mathbf{R}_3^f) \boldsymbol{\varphi} + \\ (\check{\mathbf{M}} \circ \mathbf{R}_1^m) \mathbf{A} + [\check{\mathbf{M}}(\mathbf{D} \otimes \mathbf{I})] \circ \mathbf{R}_2^m \mathbf{A} + (\check{\mathbf{M}} \circ \mathbf{R}_3^m) \boldsymbol{\varphi} = \\ \mathbf{I} \otimes \left\{ \mathbf{L}_1 \mathbf{J} \mathbf{f}^0 + \mathbf{L}_2 \mathbf{H} \mathbf{f}^\Gamma \right\} - \left(\int_{\mathcal{T}_w} \psi_p \mathbf{K}(\mathbf{A}) dt \right) \\ (\mathbf{I} \otimes \mathbf{R}_3^f) \mathbf{A} + (\hat{\mathbf{D}} \otimes \mathbf{R}_4^f) \boldsymbol{\varphi} + (\check{\mathbf{M}} \circ \mathbf{R}_3^m) \mathbf{A} + [\check{\mathbf{M}}(\hat{\mathbf{D}} \otimes \mathbf{I})] \circ \mathbf{R}_4^m \boldsymbol{\varphi} = \\ \mathbf{I} \hat{\mathbf{D}} \otimes \left\{ \mathbf{L}_3 \mathbf{J} \mathbf{f}^\Gamma \right\} \end{array} \right. \quad (11.17)$$

Finally, we show that the linearised matrix of the system is written:

$$\left(\begin{array}{cc} \mathbf{I} \otimes \mathbf{R}_1^f + \mathbf{D} \otimes \mathbf{R}_2^f + \check{\mathbf{M}} \circ \mathbf{R}_1^m + [\check{\mathbf{M}}(\mathbf{D} \otimes \mathbf{I})] \circ \mathbf{R}_2^m & \mathbf{I} \otimes \mathbf{R}_3^f + \check{\mathbf{M}} \circ \mathbf{R}_3^m \\ \mathbf{I} \otimes \mathbf{R}_3^f + \check{\mathbf{M}} \circ \mathbf{R}_3^m & \hat{\mathbf{D}} \otimes \mathbf{R}_4^f + [\check{\mathbf{M}}(\hat{\mathbf{D}} \otimes \mathbf{I})] \circ \mathbf{R}_4^m \end{array} \right) \quad (11.18)$$

By using the definitions of (14.26), the complete system is thus written:

$$\left(\left[\mathbf{I} \otimes \mathbf{G}_1^f + \mathbf{D} \otimes \mathbf{G}_2^f + \hat{\mathbf{D}} \otimes \mathbf{G}_3^f \right] + \left[\check{\mathbf{M}} \circ \mathbf{G}_1^m + [\check{\mathbf{M}}(\mathbf{D} \otimes \mathbf{I})] \circ \mathbf{G}_2^m + [\check{\mathbf{M}}(\hat{\mathbf{D}} \otimes \mathbf{I})] \circ \mathbf{G}_3^m \right] \right) \mathbf{X} = \mathbf{B}_1 + \mathbf{B}_2 - \boldsymbol{\Psi} \tilde{\mathbf{K}} \quad (11.19)$$

Remark 11.4.1 For the time being, we will only deal with motion using the blocked step method. It should be noted that there are more advanced methods for dealing with non-compliant meshes. For example, coupling between the finite element method and the spectral element methods. These latter approaches require complex theoretical treatment that we will leave for another time.

11.5 Kinematic coupling

11.5.1 Formation of the equation of the physical problem

It is possible to associate the part in motion with a mechanical equation such as:

$$J \frac{d\Omega}{dt} = C_{em} + C_r - f \Omega \quad (11.20)$$

where:

- C_{em} is the electromagnetic torque calculated by code_Carmel (S.I. units: N.m);
- C_r is the resistant torque imposed by the user (S.I. units: N.m);
- J is the inertia of the part in motion (S.I. units: N.m.s²);
- f is the friction coefficient (S.I. units: N.m.s);
- $\Omega = \frac{d\theta(t)}{dt}$ is the rotational speed (S.I. units: rad.s⁻¹).

11.5.2 Treatment

After calculation of the electromagnetic torque, time discretisation of the mechanical equation using the backward Euler method allows calculation of the rotational speed at the given computational time step Ω_t .

$$J \frac{\Omega_t - \Omega_{t-1}}{\Delta t} = C_{em_t} + C_{r_t} - f \Omega_t \quad (11.21)$$

The angular position of the rotating part is increased by the quantity $\Delta\theta = \Delta\Omega\Delta t$.

11.5.3 Weak coupling of the magnetic equation and mechanical equation

It thus remains to couple the magnetoquasistatic problem with the mechanical equation. Since the mechanical time constant for typical electrotechnical applications is much greater than in the magnetic problem, a strong coupling between the two problems is not necessary.

To go further, chaining of the two equations is even possible provided that the time discretisation constant is small enough to capture the dynamics of both models. Hence, the magnetic and mechanical equations will be solved successively during the simulation.

Chapter 12

Processing non-linearity

12.1 Fixed point

12.1.1 Description of the method

The fixed point method, also called Picard's method [Miellou, Spiteri 1985], consists in transforming the equation of the original system $f(x) = 0$ into an equivalent $g(x) = x$ having the same solution as described in Figure 12.1.

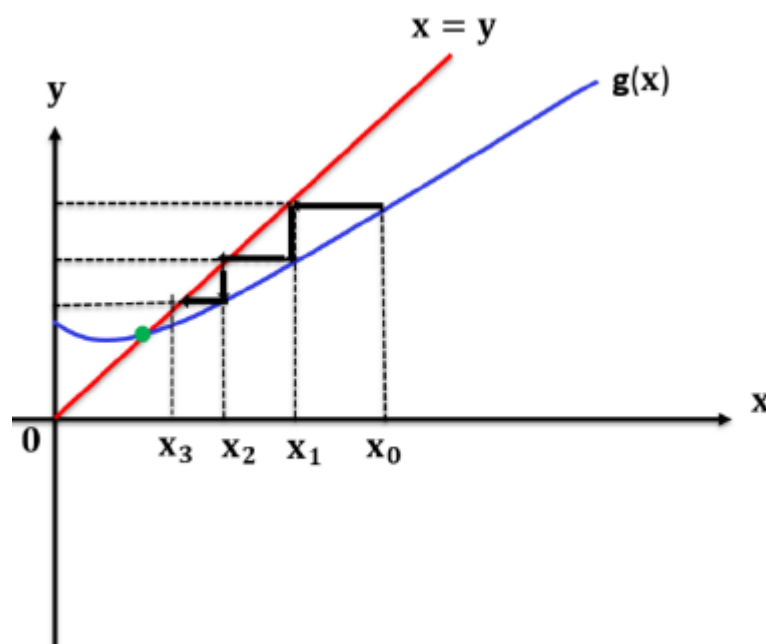


Figure 12.1: Fixed point method.

Thus, approaching zero for the initial function f is equivalent to approaching the fixed points of the equivalent function g , which is motivated by the requirements of the fixed point theorem. To better understand the mechanism of this method, we need to introduce some definitions.

Definition 12.1.1 Let $f : I \rightarrow \mathbb{R}$, a zero or root of f is any $\bar{x} \in I$ that satisfies $f(\bar{x}) = 0$.

Definition 12.1.2 A fixed point of f is any \bar{x} that satisfies $f(\bar{x}) = \bar{x}$.

Theorem 12.1.1 *Intermediate value theorem: Let f be a function, continuous on $I = [a, b]$. Thus f reaches all values between $f(a)$ and $f(b)$, $\forall d \in [f(a), f(b)]$ there is $c \in I$ such that $f(c) = d$.*

Corollary 12.1.1 *Let $f : I = [a, b] \rightarrow \mathbb{R}$ be a continuous application such that $f(a)f(b) < 0$, i.e. $f(a)$ and $f(b)$ are non-zero and of opposite sign. Hence, there is $\bar{x} \in]a, b[$ such that $f(\bar{x}) = 0$. If, in addition, f is strictly monotonic, then \bar{x} is unique.*

Corollary 12.1.2 *fixed point theorem: Let $g : [a, b] \rightarrow [a, b]$ be continuous on $[a, b]$. Then g allows a fixed point \bar{x} in the interval $[a, b]$.*

Definition 12.1.3 *A function $g : [a, b] \rightarrow \mathbb{R}$ is said to be a contraction mapping if there is $0 < \eta < 1$ such that for all $x, y \in [a, b]$ we have $|g(x) - g(y)| \leq \eta |x - y|$.*

Theorem 12.1.2 *Let $g : [a, b] \rightarrow [a, b]$ be a contraction mapping. Then, the sequence x_n defined by $x_0 \in [a, b]$, $x_{n+1} = g(x_n)$ converges on the unique fixed point of g in $[a, b]$.*

12.1.2 Approximate method and solutions of the fixed point

There are three sources of error that lead to the use of the approximate method.

1. the mathematical model studied, represented in our case by function f , may depend on parameters that are the result of experimental data, measurements made with finite precision or approximate calculations.
2. rounding errors due to some types of arithmetic used by computers.
3. approximation and truncation errors: after a finite number of steps, limit processes are stopped and transcendent functions are replaced by approximations.

Instead of looking at the conventional method $x_{n+1} = g(x_n)$ for the calculation of the fixed point x of f , calculation of x_{n+1} is performed with an error $\varepsilon > 0$ such that:

$$d(x_{n+1}, g(x_n)) \leq \varepsilon \quad (12.1)$$

There is always a possibility that all three errors can occur simultaneously, which leads to the use of the iterative method. More in-depth studies have been developed for non-linear problems by [Chaitin-Chatelin et Frayssé 1996] and [Higham 2002] for linear algebra.

The method studied is based on algorithm 12.1.

Algorithm 12.1 Fixed point algorithm.

```

1: Input :  $x_0 \in \mathbb{R}^N, \varepsilon > 0$ 
2: while  $|x_{n+1} - x_n| \geq \varepsilon$  do
3:     Calculation of  $x_{n+1} = g(x_n)$ 
4:     Increment  $n = n + 1$ 
5: end while
6: Return  $x_{n+1}$ 

```

12.1.3 Study of convergence

To measure how quickly the sequence will converge towards the fixed point, we need to introduce some tools. We define $e_n = x_n - x_*$ as the approximation error, where x_* is the minimum and x_n the estimate at iteration n . The rate of convergence is the rate at which error e_n falls towards 0. The order of convergence of sequence e_n towards 0 is defined as the largest $p > 0$ such that there is a finite limit α with:

$$\lim_{n \rightarrow \infty} \frac{e_{n+1}}{e_n^p} \leq \alpha \quad (12.2)$$

A distinction is made between different cases:

1. Linear or geometric convergence of rate α if $p = 1$ and $\alpha < 1$.
2. Superlinear convergence if $p = 1$ and $\alpha = 0$.
3. Quadratic convergence if $p = 2$.

The behaviour of the method depends on x_0 , so how should this point be chosen to guarantee convergence?

Definition 12.1.4 *The basin of attraction of a fixed point \bar{x} of g is the set of points x_0 for which the method converges towards \bar{x} .*

Ideally, the starting point is chosen in the basin of attraction. The fixed points are characterised using the relation between the basin and the derivative of g .

1. If $0 < |g'(x)| < 1$, the fixed point is said to be attracting.
2. If $|g'(x)| > 1$, the fixed point is said to be repelling.
3. If $g'(x) = 1$ the fixed point is undetermined, nothing can be said.

The methods we will study next (Newton's methods) all work on a common principle: reinterpreting equation $f(x) = 0$ as a fixed point problem $g(x) = x$, for a certain function g . The choice of function g leads to the existence of these different methods.

12.1.4 Advantages and disadvantages

The fixed point method is characterised by its robustness and ease of use, but it has a slow convergence rate, since it is only linear and the convergence factor is generally low.

12.2 Newton-Raphson

This method was described by Isaac Newton (1643 - 1727) and appears in a very general context in “De Analysi per æquationes numero terminorum infinitas” of 1669, in which Newton considers polynomial equations and uses a linearisation technique. In 1687 he published a book entitled “Philosophiæ Naturalis Principia Mathematica”, in which he describes the case of the Kepler equation in the form $x - e \sin(x) = M$, which is not polynomial. Since it is no longer possible to linearise this method using algebraic techniques, Joseph Raphson (1648 - 1715) presented a new method of solving polynomial equations in his book “Analysis æquationum universalis” in 1690. Then came Simpson (1710 - 1761) who in his “Essays in mathematicks” introduced the method of fluxions, i.e. derivatives, in 1740. The first proofs of convergence was developed by J. R. Mouraille (1721 - 1808) in 1768, then J. Fourier and A. Cauchy for the case of functions of one variable.

L. Kantorovich (1912 - 1986) and A. Ostrowski (1893 - 1986), two of the greatest names in numerical analysis, provided precise results on Newton's method of convergence. Not to forget S. Smale, the last of the great names associated with Newton's method, who introduced the “alpha theory”, which appeared very recently, in the 1980s and 1990s. For more information, the history of Newton's method is detailed in [Ypma 1995].

12.2.1 Description of the method

This method consists in linearising the non-linear problem from the approximate values of the solution and constructing a sequence that converges towards the solution [Dembo,Steihaug1983]. If the starting estimate is in the basin of convergence, the Newton-Raphson method generally converges quickly to the solution sought, otherwise it diverges because of the unreliable direction and length of the step [Kuczmamm 2010].

Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a differentiable function on an interval I . For an equation $f(x) = 0$, Newton's method is based on study of the sequence:

$$d_n = -f'(x_n)^{-1} f(x_n), \quad x_{n+1} = x_n + d_n, \quad \forall x_0 \in I \quad (12.3)$$

Definition 12.2.1 Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$, a non-zero vector d of \mathbb{R}^n is said to be a descent direction if there is $\lambda > 0$ such that for any $\alpha \in]0, \lambda[$ we have $f(x + \alpha d) < f(x)$.

Newton's method corresponds to $d_n = -G(x_n) \cdot f'(x_n)$. d_n is a descent direction if the Hesse matrix $G(x_n)$ is defined as positive.

The multidimensional Newton algorithm is given in algorithm 12.2.

Algorithm 12.2 Newton's method.

```

1: Input :  $x_0 \in \mathbb{R}^N, \varepsilon > 0$ 
2: for  $n \rightarrow n + 1$  do
3:     while  $\frac{f(x_n)}{f'(x_n)} > \varepsilon$  do
4:         Resolve  $d_n = -[f'(x_n)]^{-1} f(x_n)$ 
5:         Update  $x_{n+1} = x_n + d_n$ 
6:         Calculation of the residual  $f(x_{n+1})$ 
7:     end while
8: end for
9: Return  $x_{n+1}$ 

```

Each iteration of this algorithm requires evaluation of the Jacobian matrix $\mathbf{J} = \left[\frac{\partial f}{\partial x_i} \right]$ and the resolution of a linear system involving the Jacobian matrix that may be incorrectly conditioned [Kelley 2003].

12.2.2 Study of convergence

Theorem 12.2.1 Let $f : \Omega \rightarrow E$ be an application of class C^2 , where E is a full normalised vector space and Ω is an open set of E . If $f(x) = 0$ has a solution $x_* \in \Omega$ then there is a neighbourhood B of x_* such that for any $x_0 \in B$, the sequence x_n generated by:

$$x_{n+1} = x_n - f'(x_n)^{-1} f(x_n), \quad n = 0, 1, 2, \dots$$

exists and converges towards x_* . In addition, there is a real number $C > 1$ such that for any $n \leq 0$:

$$|x_n - x_*| \leq C^{-2n}$$

12.2.3 Magnetostatic example

All numerical examples presented in this chapter use the non-linear magnetostatic problem for vector potential \mathbf{A} . This involves solving:

$$\text{rot} \left(\frac{1}{\mu} \text{rot} \mathbf{A} \right) = \mathbf{J} \quad (12.4)$$

Figure 12.2 shows the domain under study for this 2D example inspired by a T.E.A.M. 13 workshop [Nakata et al 1995].

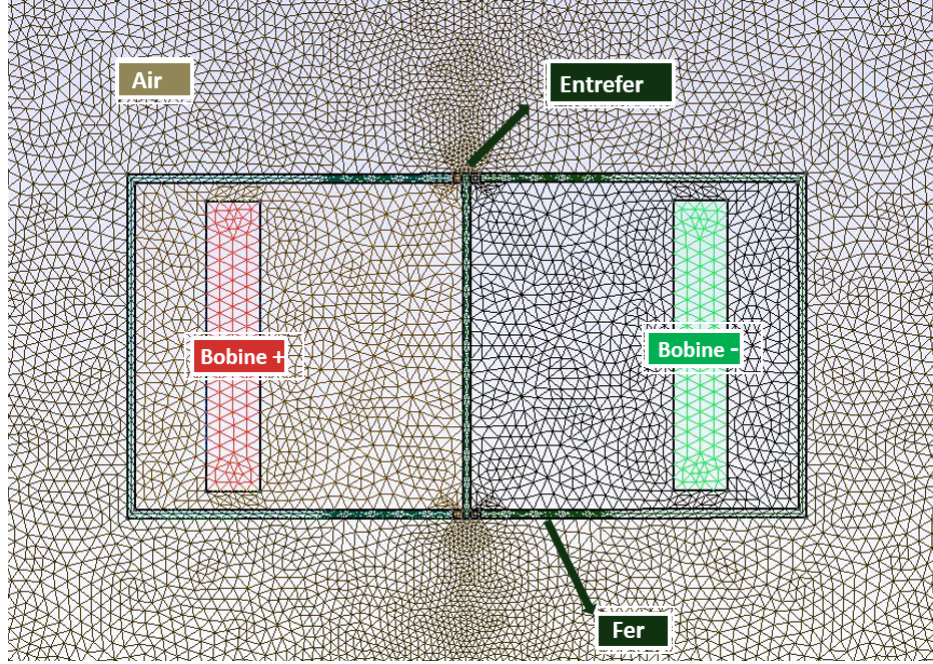


Figure 12.2: Mesh of the non-linear magnetostatic problem (TEAM13).

It consists of two U-shaped ferromagnetic cores arranged symmetrically on either side of a third central plate which is surrounded by a DC coil, which makes four air gaps. Since the plates have non-linear ferromagnetic properties, the point measured on the B - H curve in Figure 12.3 is taken according to the Marrocco model.

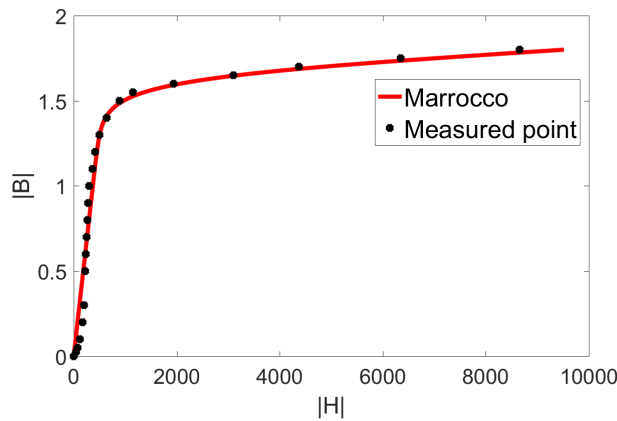


Figure 12.3: Constitutive relation of the ferromagnetic material (TEAM13).

The characteristics of the different physical domains are:

- Air: magnetic permeability $\mu_0 = 4\pi 10^{-7} H.m^{-1}$;

- Iron: non-linear permeability $\mu(B)$;
- Coils: the magnetomotive forces of the excitation coil are 1,000 At and 3,000 At, sufficient to saturate the plates.

Different levels of mesh fineness are adopted for this problem, as described in Table 12.1.

Mesh	Mesh 1	Mesh 2	Mesh 3
Number of elements	7,122	28,488	113,952
Number of nodes	1,207	4,828	19,313

Table 12.1: Information on the meshes used.

Conventional Newton's method is tested at different mesh fineness and different currents (500, 1,000, 2,000 and 3,000 At) with the conventional starting point $\mathbf{A} = 0$.

The convergence results are summarised in Table 12.2.

Current	Mesh 1	Mesh 2	Mesh 3
500 At	5	8	10
1,000 At	9	Diverges	Diverges
2,000 At	Diverges	Diverges	Diverges
3,000 At	Diverges	Diverges	Diverges

Table 12.2: Convergence results for different meshes (TEAM13)

It can be seen that Newton's method diverges as the size of the system increases and as strong saturation appears.

12.2.4 Advantages and disadvantages

The major advantage of Newton's method over a fixed point method is its 2nd-order convergence rate. This convergence always remains local.

It should also be noted that if the method does not converge, for example if the initial estimate x_0 was not chosen in the basin of convergence, then the method may diverge very quickly.

The major disadvantage of Newton's method is its cost: evaluation of the Jacobian matrix is required at each iteration, and the resolution of linear systems $f'(x_n)(x_{n+1} - x_n) = -f(x_n)$ involves the Jacobian matrix, which may be poorly conditioned.

Remark 12.2.1 *It is recalled that to solve a linear system we do not calculate the inverse of the matrix, but rather we factorise it, using LU factorisation for example, then we calculate the solutions of the systems with triangular matrices.*

12.3 Solving non linear time based problems

It is further recalled that the discretised generic problem is written:

Find $\mathbf{X}^k(t) \in \mathbb{R}^N$ such that:

$$\left(\frac{\mathbf{K}}{\tau} + \mathbf{M}_\theta(\theta) + \mathbf{M}(\mathbf{X}^k) \right) \mathbf{X}^k = \mathbf{C} \mathbf{U}^k + \frac{\mathbf{K}}{\tau} \mathbf{X}^{k-1}, \quad k = 1, \dots, N^t \quad (9.106)$$

and find $(\theta^{k+1}, \Omega^{k+1}) \in \mathbb{R}^2$ such that:

$$\begin{cases} \Omega^{k+1} &= \left(1 - \frac{\tau f_M}{J_M} \right) \Omega^k + \frac{\tau}{J_M} (\Gamma_B(\mathbf{X}^k) + \Gamma_M) \\ \theta^{k+1} &= \theta^k + \tau \Omega^{k+1} \end{cases}, \quad k = 0, \dots, N_t - 1 \quad (9.107)$$

At time step t^k , equation 9.106 defines a system of equations that is non-linear due to the operator $\mathbf{M}(\mathbf{X}^k)$. A non-linear problem is difficult to solve directly with a numerical computer. An approximation method such as the Banach fixed point or Newton-Raphson method may then be used. These two iterative approaches consist in transforming the non-linear problem 9.106 with solution \mathbf{X}^k into a series of linear problems (\mathcal{P}_j) with solution \mathbf{X}_j^k . Under certain conditions, these approaches converge to give:

$$\|\mathbf{X}^k - \mathbf{X}_j^k\| \xrightarrow{j \rightarrow +\infty} 0 \quad (12.5)$$

In practice, it is hoped that the number of non-linear iterations N_{nl} does not exceed fifty, so that the computation time remains reasonable.

To evaluate the quality of approximation \mathbf{X}_j^k , the residual vector $R(\mathbf{X}_j^k)$ is used. This is actually an image of the error, which generally behaves in the same way but with different orders of magnitude. It is simply obtained by re-injecting the approximation into the initial problem 9.106. Thus, the residual vector is written:

$$\mathbf{R}(\mathbf{X}_j^k) = \left(\frac{\mathbf{K}}{\tau} + \mathbf{M}_\theta(\theta^k) + \mathbf{M}(\mathbf{X}_j^k) \right) \mathbf{X}_j^k - \mathbf{C} \mathbf{U}^k - \frac{\mathbf{K}}{\tau} \mathbf{X}^{k-1} \quad (12.6)$$

In practice, the user chooses an error criterion $\epsilon_{nl} > 0$ and considers that the algorithm has converged when:

$$\|\mathbf{R}(\mathbf{X}_j^k)\| < \epsilon_{nl} \quad (12.7)$$

In this case, we define:

$$\mathbf{X}^k = \mathbf{X}_j^k \quad (12.8)$$

and then we move on to the next time step. We will thus detail the two linear problems 12.9 and 12.12 that must be solved with the fixed point and Newton methods respectively.

12.3.1 Numerical resolution by the fixed point method

The fixed point method consists in transforming the initial non-linear problem into a series of linear problems 12.9 defined by:

Find $\mathbf{X}_j^k(t) \in \mathbb{R}^N$ such that:

$$\left(\frac{\mathbf{K}}{\tau} + \mathbf{M}_\theta(\theta^k) + \mathbf{M}(\mathbf{X}_{j-1}^k) \right) \mathbf{X}_j^k = \mathbf{C} \mathbf{U}^k + \frac{\mathbf{K}}{\tau} \mathbf{X}^{k-1} \quad (12.9)$$

The iterative algorithm for the fixed point method is presented below:

Algorithm 12.3 Fixed point algorithm.

-
- 1: **Data:** An initial vector \mathbf{X}_0^k . Typically we take \mathbf{X}^{k-1} or the null vector.
 - 2: **Result:** The solution vector \mathbf{X}^k .
 - 3: Initialisation of $j = 1$ and $\eta = \epsilon + 1$;
 - 4: **while** $j < N_{nl}^{max}$ and $\eta > \epsilon$ **do**
 - 5: Calculation of \mathbf{X}_j^k , solution of 12.9;
 - 6: Calculation of the error associated with \mathbf{X}_j^k : $\eta_j^k = \|\mathbf{R}(\mathbf{X}_j^k)\|$;
 - 7: Increment: $j := j + 1$
 - 8: **end while**
 - 9: Saving the solution $\mathbf{X}^k = \mathbf{X}_j^k$
-

12.3.2 Numerical resolution by the Newton-Raphson method

Like the fixed-point method, the Newton-Raphson approach turns the initial non-linear problem into a sequence of linear equations 12.12. These are obtained after a development limited to the 1st order of the functional associated with the residual. This is written as the non-linear iteration j :

$$\mathbf{R}(\mathbf{X}_j^k) = \mathbf{R}(\mathbf{X}_{j-1}^k) + \frac{\partial \mathbf{R}}{\partial \mathbf{X}}(\mathbf{X}_{j-1}^k) \cdot (\mathbf{X}_j^k - \mathbf{X}_{j-1}^k) + o(\mathbf{X}_j^k - \mathbf{X}_{j-1}^k) \quad (12.10)$$

where $o(a)$ is a term that is negligible compared with a when $\|a\|$ tends towards 0:

$$\frac{o(a)}{\|a\|} \xrightarrow{a \rightarrow 0} 0 \quad (12.11)$$

Assuming that $\|\Delta \mathbf{X}\| = \|\mathbf{X}_j^k - \mathbf{X}_{j-1}^k\|$ is small enough, the term $o(\mathbf{X}_j^k - \mathbf{X}_{j-1}^k)$ becomes negligible compared with the other sides of the equation. This is referred to as a linear approximation of the functional associated with the residual. Newton's method then consists in assuming that under these assumptions of linearity, $\mathbf{R}(\mathbf{X}_j^k)$ is zero. We can thus define the non-linear Newton problem at iteration j :

Find $\mathbf{X}_j^k \in \mathbb{R}^N$ such that:

$$\mathbf{X}_j^k = \mathbf{X}_{j-1}^k - [\mathbf{J}(\mathbf{X}_{j-1}^k)]^{-1} \mathbf{R}(\mathbf{X}_{j-1}^k) \quad (12.12)$$

where the Jacobian associated with $\mathbf{R}(\cdot)$, $\mathbf{J}(\cdot) = \frac{\partial \mathbf{R}}{\partial \mathbf{X}}(\cdot) \in \mathbb{R}^{N \times N}$ is a positive semi-defined symmetric matrix. Its expression is detailed in annex E.2.

Of course, the calculated solution \mathbf{X}_j^k will not usually yield a residual $\mathbf{R}(\mathbf{X}_j^k)$ that is strictly zero. Indeed, problem 12.12 was obtained by assuming that functional \mathbf{R} was linear. Since this assumption is only an approximation in the general case, the residual is not strictly zero and this is why the approach is iterative. The algorithm for the Newton-Raphson method is presented below.

Algorithm 12.4 Newton-Raphson algorithm

-
- 1: **Data:** An initial vector \mathbf{X}_0^k . Typically we take \mathbf{X}^{k-1} or the null vector.
 - 2: **Result:** The solution vector \mathbf{X}^k .
 - 3: Initialisation of $j = 1$ and $\eta = \epsilon + 1$;
 - 4: **while** $j < N_{nl}^{max}$ and $\eta > \epsilon$ **do**
 - 5: Calculation of \mathbf{X}_j^k , solution of 12.12;
 - 6: Calculation of the error associated with \mathbf{X}_j^k : $\eta_j^k = \|\mathbf{R}(\mathbf{X}_j^k)\|$;
 - 7: Increment: $j := j + 1$
 - 8: **end while**
 - 9: Saving the solution $\mathbf{X}^k = \mathbf{X}_j^k$
-

12.3.3 Overall solution method

In the remainder of this presentation, we thus solve Newton or fixed-point methods associated with the set of equations 9.106 - 9.107. Figure 12.4 shows the overall solution method of the non-linear problem 9.106 chained with the mechanical equation 9.107.

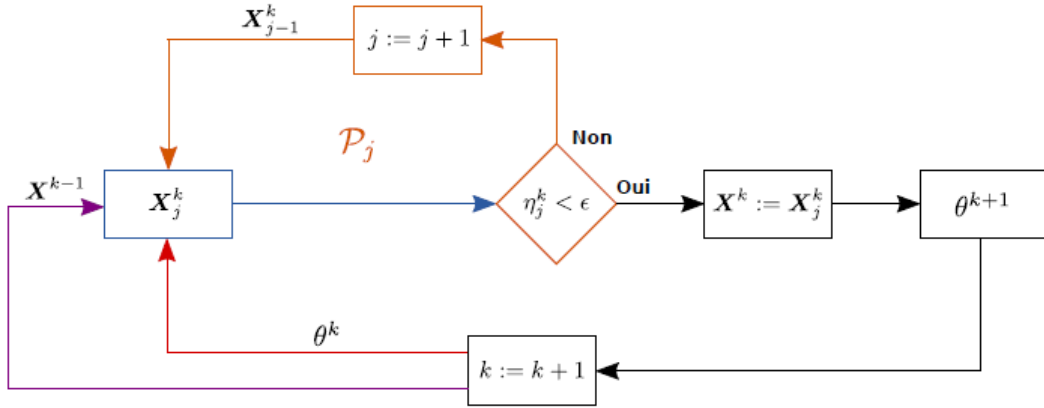


Figure 12.4: Overall solution method

12.3.4 Magnetostatic matrix system**12.3.4.1 Fixed point method for the vector magnetic potential formulation**

We recall the weak form of this formulation:

$$\forall \mathbf{w}_i^1 \in \mathcal{W}_{\Gamma_b}^1 \quad \sum_{a \in \mathcal{A}} a_a \int_{\mathcal{D}} \nu \operatorname{rot} \mathbf{w}_i^1 \operatorname{rot} \mathbf{w}_a^1 d\mathcal{D} = \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}_i^1 d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \operatorname{rot} \mathbf{w}_i^1 \cdot \mathbf{B}_r d\mathcal{D} \quad (9.32)$$

We introduce the vector of the unknowns:

$$\mathbf{X} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{N_a} \end{pmatrix} \quad (12.13)$$

We calculate the vector of the unknowns at iteration j by solving the equation 9.32 and calculate the residual consisting of:

$$\forall \mathbf{w}_i^1 \in \mathcal{W}_{\Gamma_b}^1 \quad \sum_{a \in \mathcal{A}} a_a^j \int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} - \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}_i^1 d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{B}_r d\mathcal{D} = 0 \quad (12.14)$$

The norm of this residual provides a test for the convergence criterion.

12.3.4.2 Newton's method for the vector magnetic potential formulation

We recall the weak form of this formulation

$$\forall \mathbf{w}_i^1 \in \mathcal{W}_{\Gamma_b}^1 \quad \sum_{a \in \mathcal{A}} a_a \int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} = \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}_i^1 d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{B}_r d\mathcal{D} \quad (9.32)$$

We introduce the vector of the unknowns:

$$\mathbf{X} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{N_a} \end{pmatrix} \quad (12.15)$$

We can then apply the previous equation to the residual:

$$\mathbf{R}(\mathbf{X}_j^k) = \mathbf{R}(\mathbf{X}_{j-1}^k) + \frac{\partial \mathbf{R}}{\partial \mathbf{X}}(\mathbf{X}_{j-1}^k) \cdot (\mathbf{X}_j^k - \mathbf{X}_{j-1}^k) + o(\mathbf{X}_j^k - \mathbf{X}_{j-1}^k) \quad (12.10)$$

If equation 12.10 converges at iteration j then:

$$\mathbf{R}(\mathbf{X}_j^k) = 0$$

The equation then becomes:

$$-\mathbf{R}(\mathbf{X}_{j-1}^k) = \frac{\partial \mathbf{R}}{\partial \mathbf{X}}(\mathbf{X}_{j-1}^k) \cdot (\mathbf{X}_j^k - \mathbf{X}_{j-1}^k) \quad (12.16)$$

In the current functionality of code_Carmel, only the left side in equation 9.32 depends on \mathbf{A} . At iteration j we thus have:

$$\left. \frac{\partial \mathbf{R}}{\partial \mathbf{X}} \right|_{j-1} = \frac{\partial}{\partial \mathbf{A}} \left\{ \left[\int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \right] [\mathbf{A}] \right\}_{j-1} \quad (12.17)$$

Hence:

$$\left. \frac{\partial \mathbf{R}}{\partial \mathbf{X}} \right|_{j-1} = \int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} + \left\{ \left[\int_{\mathcal{D}} \frac{\partial}{\partial \mathbf{A}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \right] [\mathbf{A}] \right\}_{j-1} \quad (12.18)$$

The second term of this equation can be written:

$$\sum_{l=1}^{n_1} \int_{\mathcal{D}} \frac{\partial}{\partial A_a} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_l^1 d\mathcal{D} A_l \quad (12.19)$$

In code_Carmel, ν depends only on \mathbf{B}^2 . However:

$$\mathbf{B} = \sum_{m=1}^{n_1} \mathbf{rot} \mathbf{w}_m^1 A_m$$

Consequently:

$$\|\mathbf{B}\|^2 = \sum_{p=1}^{n_1} \sum_{m=1}^{n_1} \mathbf{rot} \mathbf{w}_p^1 \cdot \mathbf{rot} \mathbf{w}_m^1 A_p A_m$$

This gives:

$$\frac{\partial \|\mathbf{B}\|^2}{\partial A_a} = 2 \sum_{m=1}^{n_1} \mathbf{rot} \mathbf{w}_a^1 \cdot \mathbf{rot} \mathbf{w}_m^1 A_m$$

And further:

$$\frac{\partial \|\mathbf{B}\|^2}{\partial A_j} = 2 \mathbf{rot} \mathbf{w}_a^1 \cdot \sum_{m=1}^{n_1} \mathbf{rot} \mathbf{w}_m^1 A_m|_{j-1}$$

This gives:

$$\sum_{l=1}^{n_1} \int_{\mathcal{D}} \frac{\partial}{\partial A_a} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_l^1 d\mathcal{D} A_l = \sum_{l=1}^{n_1} \int_{\mathcal{D}} \frac{\partial \nu}{\partial B^2} \frac{\partial B^2}{\partial A_a} \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_l^1 A_l d\mathcal{D} \quad (12.20)$$

This expression changes to:

$$\sum_{l=1}^{n_1} \int_{\mathcal{D}} \frac{\partial}{\partial A_l} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_l^1 d\mathcal{D} A_l = \int_{\mathcal{D}} \frac{\partial \nu}{\partial B^2} \sum_{l=1}^{n_1} \frac{\partial B^2}{\partial A_a} \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_l^1 A_l d\mathcal{D} \quad (12.21)$$

Hence:

$$\begin{aligned} \sum_{l=1}^{n_1} \int_{\mathcal{D}} \frac{\partial}{\partial A_l} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_l^1 d\mathcal{D} A_l = \\ 2 \int_{\mathcal{D}} \frac{\partial \nu}{\partial B^2} \sum_{l=1}^{n_1} \left[\left\{ \mathbf{rot} \mathbf{w}_a^1 \cdot \sum_{m=1}^{n_1} \mathbf{rot} \mathbf{w}_m^1 A_m|_{j-1} \right\} \left\{ \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_l^1 A_l|_{j-1} \right\} \right] d\mathcal{D} \end{aligned} \quad (12.22)$$

This gives the following expression:

$$\begin{aligned} \sum_{l=1}^{n_1} \int_{\mathcal{D}} \frac{\partial}{\partial A_l} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_l^1 d\mathcal{D} A_l = \\ 2 \int_{\mathcal{D}} \frac{\partial \nu}{\partial B^2} \left\{ \mathbf{rot} \mathbf{w}_a^1 \cdot \sum_{m=1}^{n_1} \mathbf{rot} \mathbf{w}_m^1 A_m|_{j-1} \right\} \left\{ \mathbf{rot} \mathbf{w}_i^1 \cdot \sum_{l=1}^{n_1} \mathbf{rot} \mathbf{w}_l^1 A_l|_{j-1} \right\} d\mathcal{D} \end{aligned} \quad (12.23)$$

Finally, we write:

$$\begin{aligned} \frac{\partial \mathbf{R}}{\partial \mathbf{X}} \Big|_{j-1} = \int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} + \\ 2 \int_{\mathcal{D}} \frac{\partial \nu}{\partial B^2} \left\{ \mathbf{rot} \mathbf{w}_a^1 \cdot \sum_{m=1}^{n_1} \mathbf{rot} \mathbf{w}_m^1 A_m|_{j-1} \right\} \left\{ \mathbf{rot} \mathbf{w}_i^1 \cdot \sum_{l=1}^{n_1} \mathbf{rot} \mathbf{w}_l^1 A_l|_{j-1} \right\} d\mathcal{D} \end{aligned} \quad (12.24)$$

The norm of this residual provides a test for the convergence criterion.

12.3.4.3 Fixed point method for the scalar electric potential formulation

We recall the formulation obtained:

$$\forall w_i^0 \in \mathcal{W}_{\Gamma_h}^0 \quad \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 d\mathcal{D} = \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{H}_s d\mathcal{D} - \int_{\mathcal{D}} w_i^0 \operatorname{div} \mathbf{B}_r d\mathcal{D} \quad (9.35)$$

We introduce the vector of the unknowns:

$$\mathbf{X} = \begin{pmatrix} \Omega_1 \\ \Omega_2 \\ \vdots \\ \Omega_{N_n} \end{pmatrix} \quad (12.25)$$

We calculate the vector of the unknowns at iteration j by solving the equation 9.35 and calculate the residual consisting of:

$$\forall w_i^0 \in \mathcal{W}_{\Gamma_h}^0 \quad \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 d\mathcal{D} - \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{H}_s d\mathcal{D} + \int_{\mathcal{D}} w_i^0 \operatorname{div} \mathbf{B}_r d\mathcal{D} = 0 \quad (12.26)$$

The norm of this residual provides a value for the convergence test.

12.3.4.4 Newton's method for the scalar magnetic potential formulation

This method is not currently implemented in the time-based version of code_Carmel.

12.3.5 Magnetodynamic matrix system

12.3.5.1 Fixed point method for the vector magnetic potential formulation

The weak form of the time-discretised equations is recalled below:

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{w}'_a \cdot \mathbf{rot} \mathbf{A}(t_{i+1}) + \sigma \mathbf{w}'_a \cdot \left(\frac{\mathbf{A}(t_{i+1})}{\Delta t} + \mathbf{grad} \varphi(t_{i+1}) \right) \right] d\mathcal{D} &= \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}'_a d\mathcal{D} \\ &+ \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{rot} \mathbf{w}'_a d\mathcal{D} + \int_{\mathcal{D}} \sigma \mathbf{w}'_a \frac{\mathbf{A}(t_i)}{\Delta t} d\mathcal{D} \\ \int_{\mathcal{D}} \sigma \mathbf{grad} w_n'^0 \left(\frac{\mathbf{A}(t_{i+1})}{\Delta t} + \mathbf{grad} \varphi(t_{i+1}) \right) d\mathcal{D} &= \int_{\mathcal{D}} \sigma \mathbf{grad} w_n'^0 \frac{\mathbf{A}(t_i)}{\Delta t} d\mathcal{D} \end{aligned} \quad (9.85)$$

We introduce the vector of the unknowns \mathbf{X} a time t_{i+1} :

$$\mathbf{X}(t_{i+1}) = \begin{pmatrix} a_1(t_{i+1}) \\ a_2(t_{i+1}) \\ \vdots \\ a_{N_a}(t_{i+1}) \\ \varphi_1(t_{i+1}) \\ \varphi_2(t_{i+1}) \\ \vdots \\ \varphi_{N_n}(t_{i+1}) \end{pmatrix} \quad (12.27)$$

We calculate the vector of the unknowns at iteration j by solving the equation 9.85 and calculate the residual consisting of:

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{w}'_a \cdot \mathbf{rot} \mathbf{A}(t_{i+1}) + \sigma \mathbf{w}'_a \cdot \left(\frac{\mathbf{A}(t_{i+1})}{\Delta t} + \mathbf{grad} \varphi(t_{i+1}) \right) \right] d\mathcal{D} - \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}'_a d\mathcal{D} - \\ \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{rot} \mathbf{w}'_a d\mathcal{D} - \int_{\mathcal{D}} \sigma \mathbf{w}'_a \frac{\mathbf{A}(t_i)}{\Delta t} d\mathcal{D} = 0 \\ \int_{\mathcal{D}} \sigma \mathbf{grad} w'_n \left(\frac{\mathbf{A}(t_{i+1})}{\Delta t} + \mathbf{grad} \varphi(t_{i+1}) \right) d\mathcal{D} - \int_{\mathcal{D}} \sigma \mathbf{grad} w'_n \frac{\mathbf{A}(t_i)}{\Delta t} d\mathcal{D} = 0 \end{aligned} \quad (12.28)$$

The norm of this residual provides the value used to test the criterion to stop the iterative process.

12.3.5.2 Newton's method for the vector magnetic potential formulation

The weak form of the time-discretised equations is recalled below:

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{w}'_a \cdot \mathbf{rot} \mathbf{A}(t_{i+1}) + \sigma \mathbf{w}'_a \cdot \left(\frac{\mathbf{A}(t_{i+1})}{\Delta t} + \mathbf{grad} \varphi(t_{i+1}) \right) \right] d\mathcal{D} = \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}'_a d\mathcal{D} \\ + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{rot} \mathbf{w}'_a d\mathcal{D} + \int_{\mathcal{D}} \sigma \mathbf{w}'_a \frac{\mathbf{A}(t_i)}{\Delta t} d\mathcal{D} \quad (9.85) \\ \int_{\mathcal{D}} \sigma \mathbf{grad} w'_n \left(\frac{\mathbf{A}(t_{i+1})}{\Delta t} + \mathbf{grad} \varphi(t_{i+1}) \right) d\mathcal{D} = \int_{\mathcal{D}} \sigma \mathbf{grad} w'_n \frac{\mathbf{A}(t_i)}{\Delta t} d\mathcal{D} \end{aligned}$$

We introduce the vector of the unknowns \mathbf{X} at time t_{i+1} :

$$\mathbf{X}(t_{i+1}) = \begin{pmatrix} a_1(t_{i+1}) \\ a_2(t_{i+1}) \\ \vdots \\ a_{N_a}(t_{i+1}) \\ \varphi_1(t_{i+1}) \\ \varphi_2(t_{i+1}) \\ \vdots \\ \varphi_{N_n}(t_{i+1}) \end{pmatrix} \quad (12.29)$$

As in magnetostatics, we can use the equation seen above on the residual:

$$\mathbf{R}(\mathbf{X}_j^k(t_{i+1})) = \mathbf{R}(\mathbf{X}_{j-1}^k(t_{i+1})) + \frac{\partial \mathbf{R}}{\partial \mathbf{X}}(\mathbf{X}_{j-1}^k(t_{i+1})) \cdot (\mathbf{X}_j^k(t_{i+1}) - \mathbf{X}_{j-1}^k(t_{i+1})) + o(\mathbf{X}_j^k(t_{i+1}) - \mathbf{X}_{j-1}^k(t_{i+1})) \quad (12.30)$$

If the previous equation converges at non-linear iteration j then:

$$\mathbf{R}(\mathbf{X}_j^k(t_{i+1})) = 0$$

The equation then becomes:

$$-\mathbf{R}(\mathbf{X}_{j-1}^k(t_{i+1})) = \frac{\partial \mathbf{R}}{\partial \mathbf{X}}(\mathbf{X}_{j-1}^k(t_{i+1})) \cdot (\mathbf{X}_j^k(t_{i+1}) - \mathbf{X}_{j-1}^k(t_{i+1})) \quad (12.31)$$

In the current functionality of code_Carmel, as before in magnetostatics, only one term depends on $\mathbf{A}(t_{i+1})$. At iteration j we thus have:

$$\frac{\partial \mathbf{R}}{\partial \mathbf{X}} \Big|_{j-1} = \frac{\partial}{\partial \mathbf{A}} \left\{ \left[\int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \right] [\mathbf{A}(t_{i+1})] \right\}_{j-1} \quad (12.32)$$

Hence:

$$\frac{\partial \mathbf{R}}{\partial \mathbf{X}} \Big|_{j-1} = \int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} + \left\{ \left[\int_{\mathcal{D}} \frac{\partial}{\partial \mathbf{A}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \right] [\mathbf{A}(t_{i+1})] \right\}_{j-1} \quad (12.33)$$

Finally, we write:

$$\begin{aligned} \frac{\partial \mathbf{R}}{\partial \mathbf{X}} \Big|_{j-1} &= \int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} + \\ 2 \int_{\mathcal{D}} \frac{\partial \nu}{\partial B^2} &\left\{ \mathbf{rot} \mathbf{w}_a^1 \cdot \sum_{m=1}^{n_1} \mathbf{rot} \mathbf{w}_m^1 A_m|_{j-1}(t_{i+1}) \right\} \left\{ \mathbf{rot} \mathbf{w}_i^1 \cdot \sum_{l=1}^{n_1} \mathbf{rot} \mathbf{w}_l^1 A_l|_{j-1}(t_{i+1}) \right\} d\mathcal{D} \end{aligned} \quad (12.34)$$

12.4 Solving non linear problem in spectral version

Non-linear resolution only applies to the vector magnetic potential formulation in the multi-harmonic version of code_Carmel. We recall the equations obtained:

$$\left\{ \begin{aligned} &\int_{\mathcal{T}_w} \psi_p \int_{\mathcal{D}} \mathcal{K}^{nl}(\mathbf{rot} \mathbf{A}) \cdot \mathbf{rot} \mathbf{w}_f^1 + \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_i A_{si} \int_{\mathcal{D}} \nu_{pf} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_f^1 + \sum_i A_{si}^{\partial} \int_{\mathcal{D}_c} \sigma \mathbf{w}_i^1 \cdot \mathbf{w}_f^1 \right. \\ &+ \left. \sum_j \varphi_{sj} \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_j^0 \cdot \mathbf{w}_f^1 \right] = \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_l J_{sl}^0 \int_{\mathcal{D}} \mathbf{w}_l^2 \cdot \mathbf{w}_f^1 + \sum_l \frac{1}{\mu} B_{sl} \int_{\mathcal{D}} \mathbf{w}_l^2 \cdot \mathbf{rot} \mathbf{w}_f^1 \right. \\ &+ \left. \sum_l H_{sl}^{\Gamma} \int_{\Gamma_H} (\mathbf{w}_l^1 \times \mathbf{n}) \cdot \mathbf{w}_f^1 \right] \\ &\sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_i A_{si}^{\partial} \int_{\mathcal{D}_c} \sigma \mathbf{w}_i^1 \cdot \mathbf{grad} w_g^0 + \sum_j \varphi_{sj} \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_j^0 \cdot \mathbf{grad} w_g^0 \right] = \\ &\sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p \right] \left[\sum_l J_{sl}^{\Gamma} \int_{\Gamma_H} (\mathbf{w}_l^2 \times \mathbf{n}) w_g^0 \right] \end{aligned} \right. \quad (9.67)$$

This system is solved by a fixed point method.

Chapter 13

Numbering the unknowns

13.1 General numbering principle

We recall the form of the generic matrix obtained above. In the preceding sections, we have seen that the modelling of electrotechnical devices can generate a number of different problems, depending on the formulation used and whether or not electrical or mechanical coupling is taken into account. Using the approaches described above, all these models can be represented by the following generic problem:

Find $\mathbf{X}(t) \in \mathbb{R}^N$ such that:

$$\mathbf{K} \frac{d\mathbf{X}(t)}{dt} + (\mathbf{M}_\theta(\theta) + \mathbf{M}(\mathbf{X})) \mathbf{X}(t) = \mathbf{C} \mathbf{U}(t), \quad \forall t \in [0, T], \quad (9.97)$$

and find $\theta(t) \in \mathbb{R}$ such that:

$$J_M \frac{d^2\theta(t)}{dt^2} + f_M \frac{d\theta(t)}{dt} = \Gamma_B(\mathbf{X}) + \Gamma_M(t) \quad (9.98)$$

with $\mathbf{U}(t)$ which represents the voltage and/or current control of the system. From these two equations some terms will be simplified depending on the application studied. so, if the problem has *ni* conductive domain, *ni* circuit coupling, we have $\mathbf{K} = 0$.

13.2 Numbering for the time-based version of code__Carmel

13.2.1 Electrokinetics

13.2.1.1 Formulation φ

In the case of an imposed voltage, it is recalled that the weak formulation of the φ problem is written as follows:

$$\forall w_i^0 \in \mathcal{W}_{\Gamma_b}^0 \quad \sum_{n \in \mathcal{N}_h} \varphi_n \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 d\mathcal{D}_c = - \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} \alpha V d\mathcal{D}_c \quad (9.25)$$

Hence the vector of the unknowns is written:

$$\mathbf{X}_\phi = \begin{pmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_{N_n} \end{pmatrix} \quad (13.1)$$

In the case of an imposed current, the system of equations in the integral formulation is:

$$\begin{aligned} \forall w_i^0 \in \mathcal{W}_{\Gamma_b}^0 \quad \sum_{n \in \mathcal{N}_h} \varphi_n \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 d\mathcal{D}_c + \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} \alpha V d\mathcal{D}_c &= 0 \\ \sum_{n \in \mathcal{N}_h} \varphi_n \int_{\mathcal{D}_c} \mathbf{grad} \alpha \cdot \sigma \mathbf{grad} (w_n^0 + \alpha V) d\mathcal{D}_c &= I \end{aligned} \quad (9.26)$$

The vector of the unknowns is thus written:

$$\mathbf{X} = \begin{pmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_{N_n} \\ V \end{pmatrix} \quad (13.2)$$

13.2.1.2 Formulation T

The integral form obtained above is recalled below:

$$\forall \mathbf{w}_i^1 \in \mathcal{W}_{\Gamma_b}^1 \quad \sum_{a \in \mathcal{A}_h} T_a \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} = - \sum_{a \in \mathcal{A}} h_{a,s} \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \quad (9.28)$$

The vector of the unknowns is thus as follows:

$$\mathbf{X}_T = \begin{pmatrix} T_1 \\ T_2 \\ \vdots \\ T_{N_a} \end{pmatrix} \quad (13.3)$$

13.2.2 Magnetostatics

13.2.2.1 Formulation A

The integral form obtained above is recalled below:

$$\forall \mathbf{w}_i^1 \in \mathcal{W}_{\Gamma_b}^1 \quad \sum_{a \in \mathcal{A}} a_a \int_{\mathcal{D}} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} = \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}_i^1 d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{B}_r d\mathcal{D} \quad (9.32)$$

The vector of the unknowns is thus as follows:

$$\mathbf{X}_A = \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_{N_a} \end{pmatrix} \quad (13.4)$$

13.2.2.2 Formulation Ω

The integral form obtained above is recalled below:

$$\begin{aligned} \forall w_i^0 \in \mathcal{W}_{\Gamma_h}^0 \quad \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 d\mathcal{D} = \\ \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{H}_s d\mathcal{D} - \int_{\mathcal{D}} w_i^0 \operatorname{div} \mathbf{B}_r d\mathcal{D} \end{aligned} \quad (9.35)$$

The vector of the unknowns is thus as follows:

$$\mathbf{X}_\Omega = \begin{pmatrix} \Omega_1 \\ \Omega_2 \\ \vdots \\ \Omega_{N_n} \end{pmatrix} \quad (13.5)$$

13.2.3 Magnetodynamics

13.2.3.1 Formulation $\mathbf{A} - \phi$

We consider a magnetodynamic problem in formulation $\mathbf{A} - \phi$ with voltage coupling, imposed magnetic potential differences and circuit coupling.

If the magnetic potential differences are imposed, magnetic fluxes are introduced as unknowns. In the case of circuit coupling, the mesh currents are also unknowns.

Thus \mathbf{X} is written:

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_A \\ \mathbf{X}_\phi \\ i_{\nu_1} \\ \vdots \\ i_{\nu_{|\nu|}} \\ \varphi_1 \\ \vdots \\ \varphi_{N_{mag}} \\ i_{m_1} \\ \vdots \\ i_{m_{N_{bocles}}} \end{pmatrix} \in \mathbb{R}^{N_a + N_n + |\nu| + N_{mag} + N_{bocles}} \quad (13.6)$$

with:

$$N = N_a + N_n + |\nu| + N_{mag} + N_{bocles}$$

Remark 13.2.1 *We start by numbering the unknowns of the edges, then the nodal unknowns and, finally, the electrical or magnetic unknowns.*

13.2.3.2 Formulation $\mathbf{T} - \Omega$

It is recalled that the integral weak form is written as follows:

$$\begin{aligned} \sum_{a \in \mathcal{A}_h} t_a \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_a^1 \cdot \mathbf{rot} \mathbf{w}_i^1 d\mathcal{D} \\ + \sum_{a \in \mathcal{A}_h} t_a \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot \frac{\partial}{\partial t} \mu \mathbf{w}_a^1 d\mathcal{D} - \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot \frac{\partial}{\partial t} \mu \mathbf{grad} w_n^0 d\mathcal{D} = \\ \sum_l H s_l \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_l^1 \cdot \mathbf{rot} \mathbf{w}_i^1 d\mathcal{D} + \sum_l H s_l \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot \frac{\partial}{\partial t} \mu \mathbf{w}_l^1 d\mathcal{D} \\ + \sum_l B r_l \int_{\mathcal{D}} \mathbf{w}_i^1 \cdot (\mathbf{w}_l^2 \times \mathbf{n}) d\mathcal{D} \quad (9.48) \end{aligned}$$

$$\begin{aligned} \sum_{a \in \mathcal{A}_h} t_a \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot \mu \mathbf{w}_a^1 d\mathcal{D} - \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot \mu \mathbf{grad} w_n^0 d\mathcal{D} = \\ \sum_l H s_l \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot \mu \mathbf{w}_l^1 d\mathcal{D} + \sum_l B r_l \int_{\mathcal{D}} \mathbf{grad} w_i^0 \cdot (\mathbf{w}_l^2 \times \mathbf{n}) d\mathcal{D} \end{aligned} \quad (9.49)$$

In addition, we must provide for current unknowns if voltage coupling is operational. If magnetic fluxes are imposed, magnetic potential differences are introduced as unknowns.

The vector of the unknowns is thus as follows:

$$\mathbf{X} = \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_{N_A} \\ \Omega_1 \\ \Omega_2 \\ \vdots \\ \Omega_{N_n} \\ i_{\nu_1} \\ \vdots \\ i_{\nu_{|\nu|}} \\ \varepsilon_1 \\ \vdots \\ \varepsilon_{N_{mag}} \end{pmatrix} \quad (13.7)$$

13.2.4 Numbering for the spectral version of code_Carmel

13.3 Dealing with floating potentials

Nodes belonging to a group with a floating potential property all have the same unknown number.

13.4 Dealing with boundary conditions

An integer is assigned to an unknown (node or edge) to indicate that the unknown is conditioned, i.e. it should not be treated as a real unknown but, for example, as part of a periodicity condition.

This integer is chosen as the largest possible integer: $2^{31} = 2147483647$. The mesh must contain fewer than 2,147,483,647 (2 billion) nodes or elements, which is within the current limits.

A test is nevertheless performed by the software to ensure this constraint.

13.5 Dealing with periodicity conditions

Depending on the periodicity condition (periodic or anti-periodic), a sign is defined and the associated unknown number is assigned this sign. Two matching nodes have the same unknown number in absolute value.

Chapter 14

Assembly

14.1 General assembly principle

The calculation of the overall matrices encountered in the formulations defined above, as well as the second terms, is performed using a general assembly procedure, which calculates the elementary matrices one element at a time and then inserts the coefficients of these matrices in the correct place in the overall matrices.

14.2 Magnetodynamic overall matrix - Harmonic case

14.2.1 Vector magnetic potential formulation

We show that system 9.67 can be written as:

$$\boxed{\mathcal{A}\mathbf{X}(t) = \mathbf{B}(t) ; \quad \forall t \in \mathcal{T}} \quad (14.1)$$

This is a non-linear system of size $N = n_0 + 2n_1$. \mathbf{X} contains the unknowns of the problem, i.e. $\mathbf{X}(t) = (\mathbf{A}(t), \mathbf{A}^\partial(t), \boldsymbol{\varphi}(t))^T = (A_1(t) \dots A_{n_1}(t), A_1^\partial(t) \dots A_{n_1}^\partial(t), \varphi_1(t) \dots \varphi_{n_0}(t))^T$.

To solve this system we use, among others, linearisation methods such as Newton- Raphson or Picard's fixed point method. By breaking down the non-linear magnetic constitutive relation as follows:

$$\mathbf{H}(\mathbf{x}, t) = \mathcal{K}(\mathbf{x}, t) = \nu_{pf} \mathbf{B}(\mathbf{x}, t) + \mathcal{K}^{nl}(\mathbf{B}(\mathbf{x}, t)) \quad (14.2)$$

System 14.1 changes to:

$$\mathcal{A}^{lin} \mathbf{X}(t) + \mathcal{A}^{nlin}(\mathbf{X}(t)) = \mathbf{B}(t) ; \quad \forall t \in \mathcal{T} \quad (14.3)$$

with:

$$\mathcal{A}^{lin} = \begin{pmatrix} \mathbf{R}_1 & \mathbf{R}_2 & \mathbf{R}_3 \\ 0 & \mathbf{R}_3^t & \mathbf{R}_4 \end{pmatrix} \quad \text{et} \quad \mathcal{A}^{nlin} = \begin{pmatrix} \mathbf{K}(\mathbf{A}(t)) & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad (14.4)$$

Matrix blocks \mathbf{R}_i , time invariant, and $\mathbf{K}(\mathbf{A}(t))$ are given by:

$$\begin{aligned}
(\mathbf{R}_1)_{ij} &= \int_{\mathcal{D}} \nu_{pf} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_j^1 d\mathcal{D}, \quad 1 \leq i, j \leq n_1 \\
(\mathbf{R}_2)_{ij} &= \int_{\mathcal{D}} \sigma \mathbf{w}_i^1 \cdot \mathbf{w}_j^1 d\mathcal{D}, \quad 1 \leq i, j \leq n_1 \\
(\mathbf{R}_3)_{ij} &= \int_{\mathcal{D}} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{w}_j^1 d\mathcal{D}, \quad 1 \leq i \leq n_0 \text{ et } 1 \leq j \leq n_1 \\
(\mathbf{R}_4)_{ij} &= \int_{\mathcal{D}} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_j^0 d\mathcal{D}, \quad 1 \leq i, j \leq n_0 \\
\mathbf{K}(\mathbf{A}) &= \int_{\mathcal{D}} \mathcal{K}(\mathbf{A}) \cdot \mathbf{rot} \mathbf{w}_f^1 d\mathcal{D}, \quad 1 \leq f \leq n_1
\end{aligned}$$

The second side $\mathbf{B}(t)$ of system 14.3 provides the volume sources and boundary conditions of the system. In the absence of current density in the conducting media, it is written as a matrix in the form:

$$\mathbf{B}(t) = \begin{pmatrix} \mathbf{L}_1 \mathbf{J}^0(t) \\ 0 \end{pmatrix} + \begin{pmatrix} \mathbf{L}_2 \mathbf{H}^\Gamma(t) \\ \mathbf{L}_3 \mathbf{J}^\Gamma(t) \end{pmatrix} \quad (14.5)$$

with:

$$\begin{aligned}
(\mathbf{L}_1)_{ij} &= \int_{\mathcal{D}} \mathbf{w}_i^2 \cdot \mathbf{w}_j^1 d\mathcal{D}, \quad 1 \leq i \leq n_2 \text{ et } 1 \leq j \leq n_1 \\
(\mathbf{L}_2)_{ij} &= \int_{\Gamma} (\mathbf{w}_i^1 \times \mathbf{n}) \cdot \mathbf{w}_j^1 d\gamma, \quad 1 \leq i, j \leq n_1 \\
(\mathbf{L}_3)_{ij} &= \int_{\Gamma} (\mathbf{w}_i^2 \times \mathbf{n}) \cdot \mathbf{w}_j^0 d\gamma, \quad 1 \leq i \leq n_2 \text{ et } 1 \leq j \leq n_1
\end{aligned} \quad (14.6)$$

and $\mathbf{J}^0(t) = (J_1^0(t), \dots, J_{n_2}^0(t))^T$ the vector of the development coefficients 7.43 of the imposed current density. Vectors $\mathbf{H}^\Gamma(t)$ and $\mathbf{J}^\Gamma(t)$ respectively the development coefficients 7.46 and 7.45.

14.2.1.1 Spectral approach to the scalable non-linear problem

Then, we denote \mathbf{X}^A the vector of size $n_1 \times N^t$ containing the ordered vectors \mathbf{A}_i of mode $i = 1$ to $i = N^t$ (see expression 9.53). The same procedure is used to define vector \mathbf{X}^φ of size $n_0 \times N^t$.

$$\mathbf{X}^A = \begin{pmatrix} \begin{bmatrix} A_{11} \\ \vdots \\ A_{1n_1} \\ A_{21} \\ \vdots \\ A_{2n_1} \\ \vdots \\ \vdots \\ A_{N^t 1} \\ \vdots \\ A_{N^t n_1} \end{bmatrix} \end{pmatrix} \quad \mathbf{X}^\varphi = \begin{pmatrix} \begin{bmatrix} \varphi_{11} \\ \vdots \\ \varphi_{1n_0} \\ \varphi_{21} \\ \vdots \\ \varphi_{2n_0} \\ \vdots \\ \vdots \\ \varphi_{N^t 1} \\ \vdots \\ \varphi_{N^t n_0} \end{bmatrix} \end{pmatrix} \quad (14.7)$$

14.2.1.1.1 Resolution using the Galerkin projection

For a fixed p ($1 \leq p \leq N^t$), the expression is contracted to:

$$\left\{ \begin{array}{l} \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p dt \right] \left[\mathbf{R}_1 \mathbf{A}_s + \mathbf{R}_2 \mathbf{A}_s^\partial + \mathbf{R}_3 \boldsymbol{\varphi}_s \right] = \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p dt \right] \left[\mathbf{L}_1 \mathbf{J}_s^0 + \mathbf{L}_3 \mathbf{B}_s + \mathbf{L}_2 \mathbf{H}_s^\Gamma \right] - \int_{\mathcal{T}_w} \psi_p \mathbf{K}(\mathbf{A}) dt \\ \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p dt \right] \left[\mathbf{R}_3^t \mathbf{A}_s^\partial + \mathbf{R}_4 \boldsymbol{\varphi}_s \right] = \left[\int_{\mathcal{T}_w} \psi_s \psi_p dt \right] \left[\mathbf{L}_3 \mathbf{J}_s^\Gamma \right] \end{array} \right. \quad (14.8)$$

14.2.1.1.2 Tensor notation

We can show that the system to solve resulting from 14.8 is a square system of size $NN^t \times NN^t$, where N is the number of spatial unknowns and N^t is the number of spectral unknowns.

In practice, the matrix of this system is never explicitly constructed because it would require significant memory resources. Given that in iterative solvers we perform matrix/vector products, it suffices to have a structure allowing us to access the matrix/vector product without storage of the entire matrix. The first solution is to incorporate the assembly operators into the matrix product module. This solution is very effective in terms of memory cost but is clearly very slow because in this case it will be necessary to construct elementary matrices for each iteration of the iterative resolution algorithm. The second solution, preferred here, is the construction of a tensor structure for which the memory cost is almost equal to that required by the harmonic finite elements matrix. Before formulating the complete system of equations in its tensor form, taking the example of the following term:

$$\sum_{s=1}^{N^t} \int_{\mathcal{T}_m} \psi_s \psi_p \sum_i A_{si} \int_{\mathcal{D}} \nu^{pf} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_f^2 = \sum_{s=1}^{N^t} \int_{\mathcal{T}} \psi_s \psi_p \mathbf{R}_1 \mathbf{A}_s ; \quad \forall p : 1 \leq p \leq N^t \quad (14.9)$$

By developing 14.9, we obtain:

$$\left(\begin{array}{ccc} \int_{\mathcal{T}} \psi_1 \psi_1 \mathbf{R}_1 & \cdots & \int_{\mathcal{T}} \psi_1 \psi_{N^t} \mathbf{R}_{N^t} \\ \vdots & \ddots & \vdots \\ \int_{\mathcal{T}} \psi_{N^t} \psi_1 \mathbf{R}_1 & \cdots & \int_{\mathcal{T}} \psi_{N^t} \psi_{N^t} \mathbf{R}_{N^t} \end{array} \right) \left(\begin{array}{c} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_{N^t} \end{array} \right) \quad (14.10)$$

By definition of the tensor product \otimes (see annex O), we show:

$$\sum_{s=1}^{N^t} \int_{\mathcal{T}_m} \psi_s \psi_p \sum_i A_{si} \int_{\mathcal{D}} \nu^{pf} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_f^2 = (\mathbf{S} \otimes \mathbf{R}_1) \mathbf{X}^A \quad (14.11)$$

where \mathbf{R}_1 is the matrix introduced in section 14.2.1, \mathbf{X}^A is the vector of the unknowns of the vector potential and \mathbf{S} refers to the square, symmetrical matrix of size $N^t \times N^t$ defined by:

$$(\mathbf{S})_{ij} = \int_{\mathcal{T}} \psi_i \psi_j w(t) dt \quad (14.12)$$

Remark 14.2.1 Note that if \mathcal{C} is an orthonormal base then \mathbf{S} is the identity matrix of size $N^t \times N^t$.

Remark 14.2.2 Due to property 7 of the Kronecker product, given in the annex (see annex O), we observe that the matrix/vector product in 14.12 is obtained without explicitly constructing $\mathbf{S} \otimes \mathbf{R}_1$ but in two steps: multiplication by \mathbf{S} then by \mathbf{R}_1 . The storage required for this operation is that required by matrix \mathbf{S} , which is generally quite small (because N^t is in the order of a few tens), and that required for storage of \mathbf{R}_1 .

It is recalled here that the coefficients of $\partial_t \mathbf{A}(\mathbf{x}, t)$ are directly related to those of $\mathbf{A}(\mathbf{x}, t)$ by equation 9.54. We will now give the tensor structure of each term of the system resulting from the space and time discretisation.

Linearised matrix

The tensor form of the matrix of the system to be solved (its linear part) can easily be calculated and is written:

$$\begin{pmatrix} \mathbf{S} \otimes \mathbf{R}_1 + (\mathbf{S} \otimes \mathbf{R}_2)(\mathbf{D} \otimes \mathbf{I}) & \mathbf{S} \otimes \mathbf{R}_3 \\ (\mathbf{S} \otimes \mathbf{R}_3^t)(\mathbf{D} \otimes \mathbf{I}) & \mathbf{S} \otimes \mathbf{R}_4 \end{pmatrix} \quad (14.13)$$

This is a non-symmetric square matrix. To make it symmetric, we multiply the second line by $\hat{\mathbf{D}} \otimes \mathbf{I}$ where $\hat{\mathbf{D}}$ is such that $\hat{\mathbf{D}}\mathbf{D} = \mathbf{I}$ ($\hat{\mathbf{D}}$ depends, like \mathbf{D} , on the spectral basis \mathcal{C} chosen). This gives:

$$\begin{pmatrix} \mathbf{S} \otimes \mathbf{R}_1 + (\mathbf{S} \otimes \mathbf{R}_2)(\mathbf{D} \otimes \mathbf{I}) & \mathbf{S} \otimes \mathbf{R}_3 \\ (\mathbf{S} \otimes \mathbf{R}_3^t)(\mathbf{D} \otimes \mathbf{I})(\hat{\mathbf{D}} \otimes \mathbf{I}) & (\mathbf{S} \otimes \mathbf{R}_4)(\hat{\mathbf{D}} \otimes \mathbf{I}) \end{pmatrix} \quad (14.14)$$

By exploiting property 3 of the Kronecker product, the tensor structure becomes:

$$\begin{pmatrix} \mathbf{S} \otimes \mathbf{R}_1 + (\mathbf{S}\mathbf{D} \otimes \mathbf{R}_2) & \mathbf{S} \otimes \mathbf{R}_3 \\ (\mathbf{S}\mathbf{D}\hat{\mathbf{D}} \otimes \mathbf{R}_3^t) & (\mathbf{S}\hat{\mathbf{D}} \otimes \mathbf{R}_4) \end{pmatrix} \quad (14.15)$$

As $\mathbf{D}\hat{\mathbf{D}} = \mathbf{I}$, the structure of the linearised matrix is finally written:

$$\begin{pmatrix} \mathbf{S} \otimes \mathbf{R}_1 + (\mathbf{S}\mathbf{D} \otimes \mathbf{R}_2) & \mathbf{S} \otimes \mathbf{R}_3 \\ (\mathbf{S} \otimes \mathbf{R}_3^t) & (\mathbf{S}\hat{\mathbf{D}} \otimes \mathbf{R}_4) \end{pmatrix} \quad (14.16)$$

Remark 14.2.3 *If matrix \mathbf{D} is diagonal, then we show that the problem to be solved comes down to the resolution of N^t problems (of harmonic finite element size) entirely decoupled (with the principle of overlapping for different frequencies. This is the case in CND, for example, where the frequencies are not applied simultaneously). What base \mathcal{C} will make \mathbf{D} diagonal? base of $e^{jk\omega t}$, for $k \neq 0$. (solutions of the differential equation $y(t) = \partial_t y(t)$).*

Second term resulting from the volume sources

The part of the second term here called volume sources is the part relating to the term of the system 14.8 containing the contribution of $(\mathbf{J}_0 \ 0)^t$. We show that this term is written:

$$\mathbf{S} \otimes \begin{pmatrix} \mathbf{L}_1 \mathbf{J}_0 \\ 0 \end{pmatrix} \quad (14.17)$$

Subsequently, this term will be referred to as \mathbf{B}_1 .

Second term resulting from the boundary conditions

$$\mathbf{S} \otimes \begin{pmatrix} \mathbf{L}_2 \mathbf{H}^\Gamma \\ \mathbf{L}_3 \mathbf{J}^\Gamma \end{pmatrix} \quad (14.18)$$

By taking account of the symmetry operator, we write:

$$\begin{pmatrix} \mathbf{S} \otimes \mathbf{L}_2 \mathbf{H}^\Gamma \\ \mathbf{S} \hat{\mathbf{D}} \otimes \mathbf{L}_3 \mathbf{J}^\Gamma \end{pmatrix} \quad (14.19)$$

We denote it B2.

Second term resulting from the non-linear part

We are interested in the term:

$$\int_{\mathcal{T}} \left(\psi_p \int_{\mathcal{D}} \mathcal{K}^{nl}(\mathbf{rot} \mathbf{A}(t)) \cdot \mathbf{rot} \mathbf{w}_f^1 \right) dt \quad (14.20)$$

Due to the non-linearity, it is not possible to decouple the spatial and temporal dimensions without introducing approximations¹. Here we propose to evaluate the time integral in a numerical way. We thus write the integral 14.20 as:

$$\forall p \in \{1 \dots N^t\} : \int_{\mathcal{T}} \left(\psi_p \int_{\mathcal{D}} \mathcal{K}^{nl}(\mathbf{rot} \mathbf{A}(t)) \cdot \mathbf{rot} \mathbf{w}_f^1 \right) w(x) dt = \sum_{q=1}^{n_q} \psi_p(t_q) \mathbf{K}(t_q) w_q \quad (14.21)$$

where \mathbf{K} is the vector of size N and given in section 9.4.2 for time t_q , n_q is the chosen order of quadrature, t_q and w_q are the points and weights of the quadrature respectively.

We now assume that we have the n_q vectors $\mathbf{K}(t_q)$ that we put into vector $\hat{\mathbf{K}} = (\mathbf{K}(t_1), \dots, \mathbf{K}(t_q))^T$. We show that term 14.21 is written:

$$\int_{\mathcal{T}} \left(\psi_p \int_{\mathcal{D}} \mathcal{K}^{nl}(\mathbf{rot} \mathbf{A}(t)) \cdot \mathbf{rot} \mathbf{w}_f^1 \right) dt = \Psi \hat{\mathbf{K}} \quad (14.22)$$

with Ψ the quadrature matrix defined by:

$$\Psi = \begin{pmatrix} \psi_1(t_1) w_1 & \psi_1(t_2) w_2 & \cdots & \psi_1(t_n) w_n \\ \psi_2(t_1) w_1 & \psi_2(t_2) w_2 & \cdots & \psi_2(t_n) w_n \\ \vdots & \vdots & \ddots & \vdots \\ \psi_P(t_1) w_1 & \psi_P(t_2) w_2 & \cdots & \psi_P(t_n) w_n \end{pmatrix} \quad (14.23)$$

General tensor form

The tensor structure of the spectral magnetodynamic system in the formulation $\mathbf{A}-\varphi$ presence of non-linearities and volume sources is written (considering the spectral basis as orthonormal i.e. $\mathbf{S} = \mathbf{I}$):

$$\boxed{\begin{pmatrix} \mathbf{I} \otimes \mathbf{R}_1 + \mathbf{D} \otimes \mathbf{R}_2 & \mathbf{I} \otimes \mathbf{R}_3 \\ \mathbf{I} \otimes \mathbf{R}_3^T & \hat{\mathbf{D}} \otimes \mathbf{R}_4 \end{pmatrix} \begin{pmatrix} \mathbf{X}^A \\ \mathbf{X}^\varphi \end{pmatrix} = \begin{pmatrix} \mathbf{I} \otimes (\mathbf{L}_1 \mathbf{J}^0 + \mathbf{L}_2 \mathbf{H}^\Gamma) - \Psi \hat{\mathbf{K}} \\ \mathbf{D} \otimes (\mathbf{L}_3 \mathbf{J}^\Gamma) \end{pmatrix}} \quad (14.24)$$

where \mathbf{X}^A is the unknown vector corresponding to the vector potential and \mathbf{X}^φ that corresponding to the unknowns of the scalar potential. The first N^t terms of \mathbf{X}^A correspond to the spectral coefficients associated with the first spatial degree of freedom. The N^t following are the spectral coefficients on the second spatial degree of freedom, etc. We proceed in the same way for \mathbf{X}^φ . We define the following matrices:

¹However, representing this term by separable functions, by techniques similar to those used for model reduction, can be used to construct an approximation of the solution (choice of initial point for example).

$$\mathbf{G}_1 = \begin{pmatrix} \mathbf{R}_1 & \mathbf{R}_3 \\ \mathbf{R}_3^T & 0 \end{pmatrix}, \quad \mathbf{G}_2 = \begin{pmatrix} \mathbf{R}_2 & 0 \\ 0 & 0 \end{pmatrix}, \quad \mathbf{G}_3 = \begin{pmatrix} 0 & 0 \\ 0 & \mathbf{R}_4 \end{pmatrix} \quad (14.25)$$

With this notation, system 14.24 is rewritten:

$$\boxed{(\mathbf{I} \otimes \mathbf{G}_1 + \mathbf{D} \otimes \mathbf{G}_2 + \hat{\mathbf{D}} \otimes \mathbf{G}_3) \mathbf{X} = \mathbf{B}_1 + \mathbf{B}_2 + \Psi \hat{\mathbf{K}}(\mathbf{X})} \quad (14.26)$$

where $\mathbf{G}_{i=1,2,3}$ are matrices of size $N \times N$ associated with the spatial dimension, and $\mathbf{I}, \mathbf{D}, \hat{\mathbf{D}}$ are matrices of size $N^t \times N^t$ associated with the time dimension.

14.2.2 Scalar electric potential formulation

We recall the system of equations obtained:

$$\begin{aligned} & \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_i T_{s,i} \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_i^1(\mathbf{x}) \cdot \mathbf{rot} \mathbf{w}_f^1 d\mathcal{D} \\ & \quad + \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_i T_{s,i}^\partial \int_{\mathcal{D}} \mu \mathbf{w}_f^1 \cdot \mathbf{w}_i^1(\mathbf{x}) d\mathcal{D} \\ & \quad - \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_j \Omega_{s,j}^\partial \int_{\mathcal{D}} \mu \mathbf{w}_f^1 \cdot \mathbf{grad} w_j^0(\mathbf{x}) d\mathcal{D} \\ & \quad - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{T}' d\gamma = \\ & \quad \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_l H s_{s,l}^\partial \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_l^1 \cdot \mathbf{rot} \mathbf{w}_f^1 d\mathcal{D} \\ & \quad + \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_l H s_{s,l}^\partial \int_{\mathcal{D}} \mu \mathbf{w}_f^1 \cdot \mathbf{w}_l^1 d\mathcal{D} \\ & \quad + \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_l B r_{s,l}^\partial \int_{\mathcal{D}} \mathbf{w}_f^1 \cdot \mathbf{w}_l^2 d\mathcal{D} \quad (9.80) \end{aligned}$$

$$\begin{aligned} & \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_i T_{s,i}^\partial \int_{\mathcal{D}} \mu \mathbf{grad} w_g^0 \cdot \mathbf{w}_i^1(\mathbf{x}) d\mathcal{D} \\ & - \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_j \Omega_{s,j}^\partial \int_{\mathcal{D}} \mu \mathbf{grad} w_g^0 \cdot \mathbf{grad} w_j^0(\mathbf{x}) d\mathcal{D} - \int_{\mathcal{T}} \int_{\partial \mathcal{D}} (\mathbf{E} \times \mathbf{n}) \cdot \mathbf{grad} \Omega' d\gamma = \\ & \quad \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_l H s_{s,l}^\partial \int_{\mathcal{D}} \mathbf{grad} w_g^0 \cdot \mu \mathbf{w}_l^1(\mathbf{x}) d\mathcal{D} \\ & \quad + \sum_s \int_{\mathcal{T}} \psi_s(t) \psi_p(t) \sum_l B r_{s,l}^\partial \int_{\mathcal{D}} \mathbf{grad} w_g^0 \cdot (\mathbf{w}_l^2(\mathbf{x}) \times \mathbf{n}) d\mathcal{D} \quad (9.81) \end{aligned}$$

Matrix blocks \mathbf{U}_i , time invariant, are given by:

$$\begin{aligned}
(\mathbf{U}_1)_{ij} &= \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_j^1 d\mathcal{D}, \quad 1 \leq i, j \leq n_1 \\
(\mathbf{U}_2)_{ij} &= \int_{\mathcal{D}} \mu \mathbf{w}_i^1 \cdot \mathbf{w}_j^1 d\mathcal{D}, \quad 1 \leq i, j \leq n_1 \\
(\mathbf{U}_3)_{ij} &= \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{w}_j^1 d\mathcal{D}, \quad 1 \leq i \leq n_0 \text{ et } 1 \leq j \leq n_1 \\
(\mathbf{U}_4)_{ij} &= \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_j^0 d\mathcal{D}, \quad 1 \leq i, j \leq n_0
\end{aligned}$$

The second term $\mathbf{C}(t)$ of system 9.80 and 9.81 provides the volume sources and boundary conditions of the system. In the absence of current density in the conducting media, it is written with matrix blocks in the form:

$$\begin{aligned}
(\mathbf{M}_1)_{ij} &= \int_{\mathcal{D}} \mathbf{w}_i^2 \cdot \mathbf{w}_j^1 d\mathcal{D}, \quad 1 \leq i \leq n_2 \text{ et } 1 \leq j \leq n_1 \\
(\mathbf{M}_2)_{ij} &= \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot (\mathbf{w}_j^2 \times \mathbf{n}) d\mathcal{D}, \quad 1 \leq i \leq n_0, 1 \leq j \leq n_2 \\
(\mathbf{M}_3)_{ij} &= \int_{\Gamma} \mathbf{w}_i^1 \cdot (\mathbf{E} \times \mathbf{n}) d\gamma, \quad 1 \leq i \leq n_1
\end{aligned} \tag{14.27}$$

For a fixed p ($1 \leq p \leq N^t$), the expression is contracted to:

$$\begin{aligned}
\sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p dt \right] \left[\mathbf{U}_1 \mathbf{T}_s + \mathbf{U}_2 \mathbf{T}_s^\partial + \mathbf{U}_3 \mathbf{\Omega}_s^\partial \right] = \\
\sum_s \left[\int_{\mathcal{T}} \psi_s(t) \psi_p(t) dt \right] \left[\mathbf{U}_1 + \mathbf{U}_2 \right] \mathbf{H}_s^\partial \\
+ \sum_s \left[\int_{\mathcal{T}} \psi_s(t) \psi_p(t) dt \right] \left[\mathbf{M}_1 \mathbf{B} \mathbf{r}_s^\partial \right] \tag{14.28}
\end{aligned}$$

$$\begin{aligned}
\sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p dt \right] \left[\mathbf{U}_3^t \mathbf{T}_s^\partial \right] - \sum_s \left[\int_{\mathcal{T}_w} \psi_s \psi_p dt \right] \left[\mathbf{U}_4 \mathbf{\Omega}_s^\partial \right] = \sum_s \left[\int_{\mathcal{T}} \psi_s(t) \psi_p(t) dt \right] \left[\mathbf{U}_3 \mathbf{H}_s^\partial \right] \\
+ \sum_s \left[\int_{\mathcal{T}} \psi_s(t) \psi_p(t) dt \right] \left[\mathbf{M}_2 \mathbf{B} \mathbf{r}_s^\partial \right] \tag{14.29}
\end{aligned}$$

As before, we introduce matrices defined by:

$$(\mathbf{S})_{ij} = \int_{\mathcal{T}} \psi_i \psi_j w(t) dt \tag{14.12}$$

The tensor form of the matrix of the system to be solved can easily be calculated and is written:

$$\begin{pmatrix} \mathbf{S} \otimes \mathbf{U}_1 + (\mathbf{S} \otimes \mathbf{U}_2) (\mathbf{D} \otimes \mathbf{I}) & (\mathbf{S} \otimes \mathbf{U}_3) (\mathbf{D} \otimes \mathbf{I}) \\ (\mathbf{S} \otimes \mathbf{U}_3^t) (\mathbf{D} \otimes \mathbf{I}) & (\mathbf{S} \otimes \mathbf{U}_4) (\mathbf{D} \otimes \mathbf{I}) \end{pmatrix} \tag{14.30}$$

14.3 Electrokinetic overall matrix

The electrokinetic formulation is only processed in the time-based version of code_Carmel.

14.3.1 Formulation φ with imposed voltage

The weak form of this formulation was obtained above:

$$\forall w_i^0 \in \mathcal{W}_{\Gamma_b}^0 \quad \sum_{n \in \mathcal{N}_h} \varphi_n \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 d\mathcal{D}_c = - \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} \alpha V d\mathcal{D}_c \quad (9.25)$$

This can be written in matrix form:

$$[\text{SPhi}] = \left[\int_{\mathcal{D}} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 d\mathcal{D} \right]_{\substack{1 \leq i \leq n_0 \\ 1 \leq n \leq n_0}} \quad (14.31)$$

Remark 14.3.1 In the time-based version, σ can be a tensor.

And, having previously calculated function α :

$$[\text{source4}] = - \left[\int_{\mathcal{D}} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} \alpha V d\mathcal{D} \right]_{1 \leq i \leq n_0} \quad (14.32)$$

We obtain the synthetic expression:

$$\begin{bmatrix} [\text{SPhi}] \end{bmatrix} \begin{bmatrix} \varphi_1 \\ \vdots \\ \varphi_{n_0} \end{bmatrix} = \begin{bmatrix} \text{source4} \end{bmatrix} \quad (14.33)$$

14.3.2 Formulation φ with imposed current

The electrokinetic formulation was obtained above in its integral form:

$$\begin{aligned} \forall w_i^0 \in \mathcal{W}_{\Gamma_b}^0 \quad \sum_{n \in \mathcal{N}_h} \varphi_n \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 d\mathcal{D}_c + \int_{\mathcal{D}_c} \sigma \mathbf{grad} w_i^0 \cdot \mathbf{grad} \alpha V d\mathcal{D}_c &= 0 \\ \sum_{n \in \mathcal{N}_h} \varphi_n \int_{\mathcal{D}_c} \mathbf{grad} \alpha \cdot \sigma \mathbf{grad} (w_n^0 + \alpha V) d\mathcal{D}_c &= I \end{aligned} \quad (9.26)$$

We obtain the synthetic expression:

$$\left[\begin{array}{c|c} [\text{SPhi}] & \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \\ \hline [\text{SPhi}] & [\text{SPhi}] \end{array} \right] \begin{bmatrix} \varphi_1 \\ \vdots \\ \varphi_{n_0} \\ \frac{\varphi_{n_0}}{V} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ I \end{bmatrix} \quad (14.34)$$

14.3.3 Formulation \mathbf{T}

We recall the weak form of the equation:

$$\forall \mathbf{w}_i^1 \in \mathcal{W}_{\Gamma_h}^1 \quad \sum_{a \in \mathcal{A}_h} T_a \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} = - \sum_{a \in \mathcal{A}} h_{a,s} \int_{\mathcal{D}} \frac{1}{\sigma} \mathbf{rot} \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \quad (14.35)$$

14.4 Magnetostatic overall matrix - Time-based case

14.4.1 Vector magnetic potential formulation

We recall the weak form of this formulation:

$$\forall \mathbf{w}_i^1 \in \mathcal{W}_{\Gamma_b}^1 \quad \sum_{a \in \mathcal{A}} a_a \int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} = \int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}_i^1 d\mathcal{D} + \int_{\mathcal{D}} \nu \mathbf{w}_i^1 \cdot \mathbf{rot} \mathbf{B}_r d\mathcal{D} \quad (9.32)$$

14.4.1.1 Linear magnetostatic vector magnetic potential

We assume a linear relationship between magnetic field and flux density.

We introduce the matrix:

$$\text{SALineaire} = \left[\int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \right]_{\substack{1 \leq i \leq n_1 \\ 1 \leq a \leq n_1}} \quad (14.36)$$

For the source terms, the following matrices are defined:

$$\text{tCAi} I_s = \left[\int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}_j^1 d\mathcal{D} \right]_{1 \leq j \leq n_1} \quad (14.37)$$

$$\text{source2} = \left[\int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_j^1 \cdot \mathbf{B}_r d\mathcal{D} \right]_{1 \leq j \leq n_1} \quad (14.38)$$

We introduce the vector of the unknowns:

$$\mathbf{X} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{N_a} \end{pmatrix} \quad (14.39)$$

This is written in synthetic form:

$$\begin{pmatrix} \text{SALineaire} \end{pmatrix} \begin{pmatrix} \mathbf{X} \end{pmatrix} = \begin{pmatrix} \text{tCAi} \end{pmatrix} I_s + \begin{pmatrix} \text{source2} \end{pmatrix} \quad (14.40)$$

14.4.1.2 Non-linear magnetostatic vector magnetic potential

We recall that the equation to be solved is:

$$-\mathbf{R}(\mathbf{X}_{j-1}^k) = \frac{\partial \mathbf{R}}{\partial \mathbf{X}}(\mathbf{X}_{j-1}^k) \cdot (\mathbf{X}_j^k - \mathbf{X}_{j-1}^k) \quad (12.16)$$

The index on the vector of the unknowns is the number of the iteration. The derivative matrix of the residual \mathbf{R} depends on the vector of the unknowns and is written:

$$\frac{\partial \mathbf{R}}{\partial \mathbf{X}} \Big|_{j-1} = \int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} + \left\{ \left[\int_{\mathcal{D}} \frac{\partial}{\partial \mathbf{A}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \right] [\mathbf{A}] \right\}_{j-1} \quad (14.41)$$

The second term in the previous equation is written:

$$\begin{aligned} \sum_{l=1}^{n_1} \int_{\mathcal{D}} \frac{\partial}{\partial A_l} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_l^1 d\mathcal{D} A_l = \\ 2 \int_{\mathcal{D}} \frac{\partial \nu}{\partial B^2} \left\{ \mathbf{rot} \mathbf{w}_a^1 \cdot \sum_{m=1}^{n_1} \mathbf{rot} \mathbf{w}_m^1 A_m|_{j-1} \right\} \left\{ \mathbf{rot} \mathbf{w}_i^1 \cdot \sum_{l=1}^{n_1} \mathbf{rot} \mathbf{w}_l^1 A_l|_{j-1} \right\} d\mathcal{D} \end{aligned} \quad (12.23)$$

This is written in matrix form by taking:

$$\text{SALineaire}(\mathbf{X}_{j-1}) = \left[\int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \right]_{\substack{1 \leq i \leq n_1 \\ 1 \leq a \leq n_1}} \quad (14.42)$$

Remark 14.4.1 The matrix term depends on \mathbf{X} because ν depends on $\|\mathbf{B}\|$

We take:

$$\text{rotRotX2D}|_a = \mathbf{rot} \mathbf{w}_a^1 \cdot \sum_{m=1}^{n_1} \mathbf{rot} \mathbf{w}_m^1 A_m|_{j-1} \quad (14.43)$$

The non-linear term linked to the reluctivity (or magnetic permeability) is written:

$$\text{SANonLineaire}(\mathbf{X}_{j-1}) = \left[\int_{\mathcal{D}} 2.0 \frac{d\nu}{dB^2} \text{rotRotX2D}|_a \cdot \text{rotRotX2D}|_i d\mathcal{D} \right]_{\substack{1 \leq i \leq n_1 \\ 1 \leq a \leq n_1}} \quad (14.44)$$

The matrix system is thus written:

$$\begin{aligned} - \left(\text{SALineaire}(\mathbf{X}_{j-1}) \right) \begin{pmatrix} \mathbf{X}_{j-1} \end{pmatrix} + \begin{pmatrix} \text{tCAi} I_s \end{pmatrix} + \begin{pmatrix} \text{source2} \end{pmatrix} = \\ \left(\text{SALineaire}(\mathbf{X}_{j-1}) + \text{SANonLineaire}(\mathbf{X}_{j-1}) \right) \begin{pmatrix} \mathbf{X}_j - \mathbf{X}_{j-1} \end{pmatrix} \end{aligned} \quad (14.45)$$

For the source terms, we recall:

$$\text{tCAi} I_s = \left[\int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}_j^1 d\mathcal{D} \right]_{1 \leq j \leq n_1} \quad (14.46)$$

$$\text{source2} = \left[\int_{\mathcal{D}} \nu \mathbf{w}_j^1 \cdot \mathbf{rot} \mathbf{B}_r d\mathcal{D} \right]_{1 \leq j \leq n_1} \quad (14.47)$$

14.4.2 Scalar magnetic potential formulation

We recall the weak form of this formulation:

$$\begin{aligned} \forall w_i^0 \in \mathcal{W}_{\Gamma_h}^0 \quad \sum_{n \in \mathcal{N}_h} \Omega_n \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{grad} w_n^0 d\mathcal{D} = \\ \int_{\mathcal{D}} \mu \mathbf{grad} w_i^0 \cdot \mathbf{H}_s d\mathcal{D} - \int_{\mathcal{D}} w_i^0 \text{div} \mathbf{B}_r d\mathcal{D} \end{aligned} \quad (14.48)$$

14.5 Magnetostatic overall matrix - Harmonic case

14.5.1 Vector magnetic potential formulation

This case is drawn directly from 14.2.1.1.2.

14.5.2 Scalar magnetic potential formulation

This case is drawn directly from 14.2.2.

14.6 Magnetodynamic overall matrix - Time-based case

14.6.1 Vector magnetic potential formulation

The discretised form of the system of equations to be resolved is recalled below.

$$\begin{aligned} & \sum_{a=1}^{n_1} a_a(i+1) \left[\int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{w}'_a \cdot \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} + \frac{1}{\Delta t} \int_{\mathcal{D}} \sigma \mathbf{w}'_a \cdot \mathbf{w}_a^1 d\mathcal{D} \right] + \\ & \sum_{n=1}^{n_0} \phi_n(i+1) \int_{\mathcal{D}} \sigma \mathbf{w}'_a \mathbf{grad} w_n^0 d\mathcal{D} = \int_{\mathcal{D}} \mathbf{J}_{s(i+1)} \cdot \mathbf{w}'_a d\mathcal{D} + \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{w}'_a d\mathcal{D} + \\ & \sum_{a=1}^{n_1} a_a(i) \frac{1}{\Delta t} \int_{\mathcal{D}} \sigma \mathbf{w}'_a \cdot \mathbf{w}_a^1 d\mathcal{D} \end{aligned} \quad (9.87)$$

$$\begin{aligned} & \sum_{a=1}^{n_1} a_a(i+1) \frac{1}{\Delta t} \int_{\mathcal{D}} \sigma \mathbf{grad} w_n'^0 \mathbf{w}_a^1 d\mathcal{D} + \sum_{n=1}^{n_0} \phi_n(i+1) \int_{\mathcal{D}} \sigma \mathbf{grad} w_n'^0 \mathbf{grad} w_n^0 d\mathcal{D} = \\ & + \sum_{a=1}^{n_1} a_a(i) \frac{1}{\Delta t} \int_{\mathcal{D}} \sigma \mathbf{grad} w_n'^0 \mathbf{w}_a^1 d\mathcal{D} \end{aligned} \quad (9.88)$$

This is written in matrix form by taking:

$$\text{SALineaire} = \left[\int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}'_i \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \right]_{\substack{1 \leq i \leq n_1 \\ 1 \leq j \leq n_1}} \quad (14.49)$$

$$\text{TA} = \int_{\mathcal{D}} \sigma \mathbf{w}'_a \cdot \mathbf{w}_a^1 d\mathcal{D} \quad (14.50)$$

$$\text{SPhi} = \int_{\mathcal{D}} \sigma \mathbf{grad} w_n'^0 \mathbf{grad} w_n^0 d\mathcal{D} \quad (14.51)$$

$$\text{tCAPhi} = \int_{\mathcal{D}} \sigma \mathbf{w}'_a \mathbf{grad} w_n^0 d\mathcal{D} \quad (14.52)$$

For the source terms:

$$\text{tCAiI}_s = \left[\int_{\mathcal{D}} \mathbf{J}_s \cdot \mathbf{w}_j^1 d\mathcal{D} \right]_{1 \leq j \leq n_1} \quad (14.53)$$

$$\text{source2} = \left[\int_{\mathcal{D}} \nu \mathbf{w}_j^1 \cdot \mathbf{rot} \mathbf{B}_r d\mathcal{D} \right]_{1 \leq j \leq n_1} \quad (14.54)$$

In the case of a wound inductor supplied with voltage or current, we calculate:

$$\text{tCAi} = \int_{\mathcal{D}} \mathbf{w}_j^1 N d\mathcal{D} \quad (14.55)$$

For an imposed current, we add the source term:

$$\text{source1} = \text{tCAi} I_{\text{inducteur}} \quad (14.56)$$

For an imposed voltage, we add the source term:

$$\text{source7} = -\text{tCAi} \mathbf{X}_{j-1} \quad (14.57)$$

where: \mathbf{X}_{j-1} is the vector of the unknowns at the previous non-linear iteration.

For circuit coupling with Qucs, we calculate the matrix:

$$\text{SN} = \int_{\mathcal{D}} \mathbf{w}_j^1 N d\mathcal{D} \text{matJi} \quad (14.58)$$

We also define:

$$\text{source3} = [\text{SALineaire}] \mathbf{X}_{A,j-1} \quad (14.59)$$

where $\mathbf{X}_{A,j-1}$ is the vector of the unknowns of the edges at the previous non-linear iteration.

This is written in matrix form by taking:

$$\text{SALineaire} = \left[\int_{\mathcal{D}} \nu \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \right] \begin{matrix} 1 \leq i \leq n_1 \\ 1 \leq a \leq n_1 \end{matrix} \quad (14.60)$$

The non-linear term linked to the reluctivity (or magnetic permeability) is written:

$$\text{SANonLineaire} = \left[\int_{\mathcal{D}} 2.0 \frac{d\nu}{dB} \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 d\mathcal{D} \right] \begin{matrix} 1 \leq i \leq n_1 \\ 1 \leq a \leq n_1 \end{matrix} \quad (14.61)$$

The first dynamic matrix is written:

$$\text{SPhi} = \left[\int_{\mathcal{D}} \sigma \mathbf{grad} w_n^0 \mathbf{grad} w_n^0 d\mathcal{D} \right] \quad (14.62)$$

The second dynamic matrix is written:

$$\text{CPhi} = \left[\int_{\mathcal{D}} \sigma \mathbf{w}_a^1 \mathbf{grad} w_n^0 d\mathcal{D} \right] \quad (14.63)$$

The third dynamic matrix is written:

$$\text{TA} = \left[\int_{\mathcal{D}} \sigma \mathbf{w}_a^1 \mathbf{w}_a^1 d\mathcal{D} \right] \quad (14.64)$$

For the second term, we have:

$$\text{CAi} = \left[\int_{\mathcal{D}} \mathbf{J}_{s(i+1)} \cdot \mathbf{w}_a^1 d\mathcal{D} \right] \quad (14.65)$$

If there are inductors supplied with voltage, we add an unknown.

If there are magnets:

$$\text{source2} = \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{w}_a^1 d\mathcal{D} \quad (14.66)$$

We introduce imposed voltage source terms on a solid conductor.

$$\text{source4} = \left[dt \int_{\mathcal{D}} \sigma \mathbf{grad} w_n^0 \mathbf{grad} w_n^0 V d\mathcal{D} \right] \quad (14.67)$$

$$\text{source5} = \left[\int_{\mathcal{D}} \sigma \mathbf{w}_a^1 \mathbf{grad} w_n^0 V d\mathcal{D} \right] \quad (14.68)$$

For magnetic sources with an imposed flux, we add:

$$\text{source8} = \left[\int_{\mathcal{D}} 2.0 \frac{d\nu}{dB} \mathbf{rot} \mathbf{w}_i^1 \mathbf{rot} \mathbf{w}_a^1 \mathbf{K} d\mathcal{D} \right] \begin{matrix} 1 \leq i \leq n_1 \\ 1 \leq j \leq n_2 \end{matrix} \quad (14.69)$$

$$\text{source9} = \left[\int_{\mathcal{D}} \sigma \mathbf{w}_a^1 \mathbf{w}_a^1 d\mathcal{D} \right] (\mathbf{K} - \mathbf{K}_{prec}) / dt \quad (14.70)$$

$$\text{source10} = \left[\int_{\mathcal{D}} \sigma \mathbf{w}_a^1 \mathbf{grad} w_n^0 d\mathcal{D} \right] (\mathbf{K} - \mathbf{K}_{prec}) \quad (14.71)$$

14.6.2 Scalar magnetic potential formulation

14.7 Processing overall values

14.7.1 Magnetodynamics

14.7.1.1 Imposing a voltage on a coiled conductor

$$\begin{aligned} \int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{w}_a^1 \cdot \mathbf{rot} \mathbf{A}_{(i+1)} + \sigma \mathbf{w}_a^1 \cdot \left(\frac{\mathbf{A}_{(i+1)}}{\Delta t} + \mathbf{grad} \varphi_{(i+1)} \right) \right] d\mathcal{D} &= \int_{\mathcal{D}} \mathbf{J}_{s(i+1)} \cdot \mathbf{w}_a^1 d\mathcal{D} \\ &+ \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_r \cdot \mathbf{w}_a^1 d\mathcal{D} + \int_{\mathcal{D}} \sigma \mathbf{w}_a^1 \frac{\mathbf{A}_{(i)}}{\Delta t} d\mathcal{D} \\ \int_{\mathcal{D}} \sigma \mathbf{grad} w_n^0 \left(\frac{\mathbf{A}_{(i+1)}}{\Delta t} \mathbf{grad} \varphi_{(i+1)} \right) d\mathcal{D} &= \int_{\mathcal{D}} \sigma \mathbf{grad} w_n^0 \frac{\mathbf{A}_{(i)}}{\Delta t} d\mathcal{D} \end{aligned} \quad (9.85)$$

$$\int_{\mathcal{D}} \frac{\mathbf{A}_{(i+1)}}{\Delta t} \cdot \mathbf{N} d\mathcal{D} + R i = V + \int_{\mathcal{D}} \frac{\mathbf{A}_{(i)}}{\Delta t} \cdot \mathbf{N} d\mathcal{D} \quad (14.72)$$

14.8 Coupling with an external circuit

14.8.1 Breakdown of the source current

We recall that when a device is powered by n^I wound inductors, the total source current density $\mathbf{J}_s(\mathbf{X}, t)$ is broken down in the form:

$$\mathbf{J}_s(\mathbf{X}, t) = \sum_{k=1}^{n^I} \mathbf{N}_k(\mathbf{x}) i_k(t) \quad (14.73)$$

where $\mathbf{N}_k(\mathbf{x})$ (m^{-2}) is the coil density associated with inductor k , $k = 1, \dots, n^I$ and $i_k(t)$ (A) is the current flowing inside. $\mathbf{N}_k(\mathbf{x})$ can be defined by:

$$\mathbf{N}_k(\mathbf{x}) = \frac{n_k^s}{|\Sigma_k|} \mathbf{n}_k(\mathbf{x}) \quad (6.2)$$

with $|\Sigma_k|$ the surface generated by the inductor, n_k^s its number of coils and \mathbf{n}_k the normal unit vector at the cross-section of the coil. In discrete terms, we thus write the second term of magnetoquasistatic problems as:

$$\mathbf{F}(\mathbf{x}, t) = \sum_{k=1}^{n^I} \mathbf{F}_k(\mathbf{x}) i_k(t) \quad (14.74)$$

where we introduce the discretisation of the coil density vectors $(\mathbf{F}_k)_i$ defined by:

$$(\mathbf{F}_k)_i = \int_{\mathcal{D}} (\mathbf{N}_k \cdot \mathbf{w}_i^1) d\mathbf{D} \quad (14.75)$$

14.8.2 Circuit equation

As seen above, we can impose either the current flowing in the wound inductors or the voltage at their terminals. In the first case, the current is the premise of the problem. In the second, the current flowing inside becomes an unknown in the problem. It is assumed that a voltage $v_k(t)$ is imposed on the inductor terminals k in a circuit containing a voltage source $v_k(t)$ in series with resistance R_k and inductance L_k . R_k represents the resistance of the winding and possibly an external resistance, while L_k models for magnetic leaks associated with non-modelled winding overhang and/or an external inductance. Finally, the current $i_k(t)$ in this circuit is a solution of:

$$\frac{\partial \phi_k(t)}{\partial t} + L_k \frac{\partial i_k(t)}{\partial t} + R_k i_k(t) = v_k(t) \quad (14.76)$$

where ϕ_k is the magnetic flux captured by the coil k . This is the term that will be used to couple the circuit equations with the magnetoquasistatic problem.

14.8.3 Expression for the magnetic flux

The flux generated by the inductor is expressed by definition as:

$$\phi_k = n_k^s \int_{S_k} (\mathbf{B} \cdot d\mathbf{S}_k) \quad (14.77)$$

where \mathbf{S}_k is the surface generated by the contour of the coil k .

Applying Stokes' theorem and using $\mathbf{B} = \text{rot } \mathbf{A}$, we have:

$$\phi_k = n_k^s \oint_{l_k} (\mathbf{A} \cdot d\mathbf{l}_k) = n_k^s \oint_{l_k} (\mathbf{A} \cdot \mathbf{N}_k) dl_k \quad (14.78)$$

where l_k is the closed contour bounding the surface S_k , again shown in Figure 6.1. Using the definition of \mathbf{N}_k (see equation 6.2), we finally find:

$$\phi_k = \int_{V_k} (\mathbf{A} \cdot \mathbf{N}_k) dV_k \quad (14.79)$$

where: $V_k = \oint_{l_k} |\Sigma_k| dl_k$ is the inductor volume. In discrete terms, this relation is simply written:

$$\phi_k = \mathbf{F}_k^t \mathbf{X}_\mathbf{A} \quad (14.80)$$

with \mathbf{F}_k defined by expression 14.75.

14.8.4 Strong coupling of the magnetic equation with circuit equations

To clearly present the coupling, we take the case of a linear magnetostatic problem. This is written:

$$\mathbf{M}_{rr} \mathbf{X}_A(t) = \sum_{k \in \mathcal{I}} \mathbf{F}_k i_k(t) + \sum_{k \in \mathcal{V}} \mathbf{F}_k i_k(t), \quad \forall t \in [0, T] \quad (14.81)$$

where the sets \mathcal{I} and \mathcal{V} contain the indices $|\mathcal{I}|$ and $|\mathcal{V}|$ of the inductors at the imposed voltage and current respectively with $n^I = |\mathcal{I}| + |\mathcal{V}|$. As explained above, the $|\mathcal{V}|$ imposed currents are a premise of the problem while the $|\mathcal{I}|$ others become unknowns. Thus, we define the new unknown vector \mathbf{X} containing the magnetic unknowns and the unknown currents by:

$$\mathbf{X}(t) = \begin{pmatrix} \mathbf{X}_A(t) \\ i_{\mathcal{V}_1}(t) \\ \vdots \\ i_{\mathcal{V}_{|\mathcal{V}|}}(t) \end{pmatrix} \quad (14.82)$$

It then remains to couple the magnetic equation with the $|\mathcal{V}|$ circuit equations. Using the flux expression, the coupled magnetostatic problem is written:

Find $\mathbf{X}(t) \in \mathbb{R}^{N_A + |\mathcal{V}|}$ such that:

$$\mathbf{K} \frac{d\mathbf{X}(t)}{dt} + \mathbf{M} \mathbf{X} = \mathbf{F}^{\mathcal{I}} \mathbf{I}(t) + \mathbf{F}^{\mathcal{V}} \mathbf{V}(t), \quad \forall t \in [0, T] \quad (14.83)$$

with:

$$\mathbf{M} = \left(\begin{array}{c|ccc} & & & & \\ & \mathbf{M}_{rr} & & & \\ \hline & 0 & & & \\ & \vdots & & & \\ & 0 & & & \end{array} \middle| \begin{array}{ccc} -\mathbf{F}_{\mathcal{V}_1} & \cdots & -\mathbf{F}_{\mathcal{V}_{|\mathcal{V}|}} \\ R_{\mathcal{V}_1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & R_{\mathcal{V}_{|\mathcal{V}|}} \end{array} \right) \quad (14.84)$$

$$\mathbf{K} = \left(\begin{array}{c|ccc} & & & & \\ & 0 & & & \\ \hline & \mathbf{F}_{\mathcal{V}_1}^t & & & \\ & \vdots & & & \\ & \mathbf{F}_{\mathcal{V}_{|\mathcal{V}|}}^t & & & \end{array} \middle| \begin{array}{ccc} & & \\ L_{\mathcal{V}_1} & & \\ & \ddots & \\ & & L_{\mathcal{V}_{|\mathcal{V}|}} \end{array} \right) \quad (14.85)$$

$$\mathbf{F}^{\mathcal{I}} = \begin{pmatrix} \mathbf{F}_{\mathcal{I}_1} & \cdots & \mathbf{F}_{\mathcal{I}_{|\mathcal{I}|}} \\ \hline 0 \end{pmatrix} \in \mathbb{R}^{(N_A+|\mathcal{V}|) \times |\mathcal{I}|}, \quad \mathbf{I}(t) = \begin{pmatrix} i_{\mathcal{I}_1}(t) \\ \vdots \\ i_{\mathcal{I}_{|\mathcal{I}|}}(t) \end{pmatrix} \in \mathbb{R}^{|\mathcal{I}|} \quad (14.86)$$

$$\mathbf{F}^{\mathcal{V}} = \begin{pmatrix} 0 \\ \hline 1 & 0 & 0 \\ & \ddots & \\ 0 & 0 & 1 \end{pmatrix} \in \mathbb{R}^{(N_A+|\mathcal{V}|) \times |\mathcal{V}|} \quad \text{et} \quad \mathbf{V}(t) = \begin{pmatrix} v_{\mathcal{V}_1}(t) \\ \vdots \\ v_{\mathcal{V}_{|\mathcal{V}|}}(t) \end{pmatrix} \in \mathbb{R}^{|\mathcal{V}|} \quad (14.87)$$

Finally, we choose to introduce the source vector $\mathbf{U}(t) \in \mathbb{R}^{|\mathcal{V}|+|\mathcal{I}|}$ in which we vertically concatenated $\mathbf{I}(t)$ with $\mathbf{V}(t)$, as well as matrix $\mathbf{C} \in \mathbb{R}^{N \times |\mathcal{V}|+|\mathcal{I}|}$ which contains matrices $\mathbf{F}^{\mathcal{V}}$ and $\mathbf{F}^{\mathcal{I}}$. We thus have:

$$\mathbf{C} \mathbf{U}(t) = \mathbf{F}^{\mathcal{I}} \mathbf{I}(t) + \mathbf{F}^{\mathcal{V}} \mathbf{V}(t) \quad (14.88)$$

and finally, the problem is rewritten:

Find $\mathbf{X}(t) \in \mathbb{R}^{N_A+|\mathcal{V}|}$ such that:

$$\mathbf{K} \frac{d\mathbf{X}(t)}{dt} + \mathbf{M} \mathbf{X} = \mathbf{C} \mathbf{U}(t), \quad \forall t \in [0, T] \quad (14.89)$$

14.9 Dealing with domains that are not simply connected

If the conductive domain is not contractile and has a “hole” for example, we introduce a vector \mathbf{K} and, under these conditions, \mathbf{J}_{ind} becomes equal to:

$$\mathbf{J}_{ind} = \mathbf{rot}(\mathbf{T} + i \mathbf{K}) \quad (14.90)$$

with i a real coefficient associated with a current. Vector \mathbf{K} is defined throughout the domain and vector \mathbf{T} always equals zero outside of \mathcal{D}_c . The use of vector \mathbf{K} is detailed in section 3.1.

Part IV

Resolution of the matrix system

Chapter 15

Resolution of the linear system

15.1 Overview of linear systems

15.1.1 Calculation costs for field physics simulations

In numerical simulation of physical phenomena, a *high computation cost (RAM, disk, CPU) often results from constructing and resolving linear systems*. Simulation in electromagnetism is no exception! The cost of building the system depends on the number of integration points and the complexity of the constitutive relations, while the cost of resolution depends on the number of unknowns, the models chosen and the topology. When the number of unknowns increases dramatically, the second stage becomes predominant and will thus be our main focus here.

Remark 15.1.1 *In addition, when it is possible to perform better in this resolution phase (in time and in RAM consumption), through access to a parallel machine, this advantage can spread to the actual system building phase (elementary calculations and assemblies) via the “distributed parallel” mode. This is done by distributing the elementary calculations and associated matrix blocks on the processors. For example, we can adopt the distribution, natural in finite elements, that each processor is responsible for a group of elements. This is the principle behind the parallelism of certain codes developed by EDF, e.g. Code_Aster and Telemac. In the future, it can also be applied to code_Carmel.*

Remark 15.1.2 *In code_Carmel, even in linear, the cost of the construction phase of the system is not negligible (in time and memory) compared with the actual resolution phase. This construction step is being redesigned to return to a more conventional cost hierarchy: switching to dynamic allocation, optimising profile search, limiting the number of loops nested in the assembly routine, etc.*

These *inversions of linear system* are in fact ubiquitous in field calculation codes and often buried deep in other numerical algorithms: non-linear method, time integration, modal analysis, etc. Hence in code_Carmel, we most often seek to calculate the vector of unknowns \mathbf{u} verifying a real symmetrical linear system.¹ of type:

$$\mathbf{K} \mathbf{u} = \mathbf{f} \tag{15.1}$$

with \mathbf{K} a matrix and \mathbf{f} a second member vector.

In general, solving this type of problem requires more thought than might appear:

- Do we have access to the matrix or do we simply know its action on a vector?

¹In double precision if using the MUMPS direct solver or the PCG iterative solver. In single precision at the preconditioning step of the PCG if using the MUMPS preconditioner.

- Is this matrix sparse or dense?
- What are its numerical properties (symmetry, positivity, regularity, etc.) and structural properties (real/complex, banded, in blocks, etc.)?
- Do we want to solve one system, several at the same time² or consecutively³? Even several different and successive systems whose matrices are very close⁴?
- In the case of successive resolutions, can previous results be reused to facilitate future resolutions (see restart technique, partial factorisation)?
- What is the order of magnitude of the size of the problem size, the matrix, and its factorisation compared with the processing capabilities of processors and associated memory (RAM, disk)?
- Do we want a very precise solution or just an estimate (see nested solvers)?
- Do we have access to linear algebra libraries (and their prerequisites MPI, BLAS, LAPACK, etc.), do we have to use “in-house” products, or possibly a combination of both?

In *code_Carmel*, the matrix is explicitly constructed, we store it in CSR format⁵ and it is completely managed in RAM (no dumping to disk). With most models, the matrix is sparse (finite element discretisation), more or less well conditioned (because sometimes numerically singular due to non-gauged modelling) and, for the moment, essentially symmetric real double precision. In addition, it does not currently offer any particular structure (blocks, bands, etc.) on which optimised processing could have been based.

Most of the resolutions are “one-shot”, i.e. we change the matrix and second member every time. Except in non-linear, where to save time, the same tangent matrix can be kept within several Newton iterations (following the values of the new parameter `reacprecond_methodeNL` see section J.3.1). This places us firmly within the framework of a strategy of *multiple second members*. As for the size of the problems, even if they increase year by year, they are modest compared with CFD: at most, in the order of a few million unknowns.

In addition, *from a functional view point*, the code now potentially relies on⁶ certain libraries⁷ that are optimised and durable in time (BLAS, MUMPS and its dependencies) and can be used on multi-core desktops and computer clusters. The aim is therefore to optimise the use of linear and non-linear solvers in this way, while allowing for both “push-button” use (training, standard studies, prototyping in electrical engineering) and “advanced” use (numerical expertise, difficult or excessively large calculations).

Remark 15.1.3 *The code_Carmel requirements in terms of linear solvers are complementary to those of code_Aster: QA requirement, ease of prototyping, physics handled. Far from doing us any harm, this enhances our product feedback and our external credibility in these areas.*

²Same matrix but several independent second members; See construction of a Schur complement.

³Problems with multiple second members: same matrix but several successive and interdependent second members; See Newton’s method without recalculating the tangent matrix.

⁴Problems with multiple second members: several matrices and several successive and interdependent second members, but with matrices that are “spectrally” similar; See Newton’s method without recalculating the tangent matrix.

⁵Compressed Sparse Row like in Code_Aster (also called MORSE format). TELEMAT and Code_Saturne have sometimes chosen other strategies: memory-optimised matrix storage format and adapted for the matrix-vector product, non-assembly of the overall matrix, non-use of a reference finite element, integration of the basic terms by analytical means, etc.

⁶Following the values of the pre-compilation options `USE_MUMPS` and `USE_BLAS` (see section 6 [Boiteau 2014]).

⁷For a more detailed analysis of linear algebra libraries see paragraph



Figure 15.1: Two classes of methods to resolve a linear system of type $\mathbf{K} \mathbf{u} = \mathbf{f}$: direct and iterative

15.1.2 Two families of methods to resolve a linear system

For 60 years, two types of technique have been competing for supremacy in the field, direct linear solvers and iterative linear solvers. The first are robust, ergonomic and universal and result in a finite number of operations (theoretically) known beforehand. Their theory is relatively well developed and their application to many types of matrices and software architectures is very complete. In particular, their multi-level algorithmics is well suited to the memory hierarchies of current machines. However, they require *storage capacities that grow rapidly with the size of the problem* which limits the *extensibility of their parallelism*⁸ Even if this parallelism can be broken down into several independent strata, thus increasing performance. On the other hand, *iterative methods are more “scalable”* when increasing the number of processors⁹. They consume *little memory*¹⁰ but their implementation is often “problem-dependent”. Their theory is full of many “open problems”, especially in finite arithmetic. In practise, their convergence in a “reasonable” number of iterations is not always achieved, it depends on the structure of the matrix, the starting point, the stopping criterion, etc. In addition, they are not well suited to effectively solving problems of the “multiple second members” type. Hence, unlike their direct counterparts, *it is not possible to offer THE iterative solver that will resolve any linear system*. The algorithm type is matched to a problem class on a case-by-case basis. They do, however, have other advantages that have historically made them the preferred choice for certain applications. With equivalent memory management, they require less memory than direct solvers, because we just need to know the action of the matrix on any vector, without actually having to store it. On the other hand, we are not subject to the “dictates” of the fill-in phenomenon that deteriorates the profile of the matrices, we can effectively exploit the sparse character of the operators and control the accuracy of the results¹¹.

In short, the use of direct solvers is more a technical matter, whereas choosing the right combination of iterative method and preconditioner is more of an art! Despite its basic simplicity on paper, solving a linear system, even a symmetric one, is not a “long quiet river”. You have to choose between two evils, filling/pivoting or preconditioning!

15.1.3 Solutions offered by Code_Carmel

Nevertheless, iterative methods work rather well in code_Carmel. The code has a “house” conjugate gradient. It can be pre-conditioned (see section 15.2.2.3) by an ILU(0) single-level incomplete Crout (parameter `LinearSolverType=1`, see section 15.2.3.2) or by a simple Jacobi

⁸This is also referred to as “scalability” or scaling up.

⁹This is the choice made by Code_Saturne, Syrthes, TELEMAC and the PCG strategies of Code_Carmel and Code_Aster.

¹⁰apart from the needs of some preconditioners.

¹¹This can be very interesting in the context of nested solvers (e.g. Newton + PCG), see V. Frayssé. The power of backward error analysis. HDR of the Institut National Polytechnique de Toulouse (2000).

(`LinearSolverType=2`, see section 15.2.3.1). However, *its robustness is sometimes criticised* and there may be a need for *another type of solver to corroborate its results*.

This is why we introduced the MUMPS direct solver. This product can serve as:

- *Benchmark linear solver*: expensive but very rich numerically (`LinearSolverType = 4`, see section 15.3.6).
- *Expertise tool*: analysis of singularity and matrix conditioning, numerical tools easily controllable (see section 15.3.5.3).
- *Flexible elementary building block* (see section 15.4) to construct a preconditioner (`LinearSolverType=3`, see section 15.2.3.3; In single precision and possibly by relaxing terms via `mumps_relax`) or to optimise non-linear solver-linear solver coupling (pooling of the factorisation step via `reacprecond_methodeNL`, see section J.3).
- Future parallelism vector in the code: centralised/distributed parallelism via MPI possibly with threaded BLAS (see section 15.3.5.2).

Other codes such as Code_Aster are less lucky with their iterative solvers. The latter, because of its mixed modelling, very dissimilar material characteristics and saddle point problems, often requires more sophisticated and more costly preconditioners to ensure relatively robust operation (non-relaxed single-precision MUMPS, adapted multi-grid method).

In TELEMAT, Syrthes and Code_Saturne, the organisation of the data flow and the external algorithmics are “tuned” to take maximum advantage of PCG-type iterative solvers. Sometimes, however, at the cost of constraints on data input, code developability/maintenance and restrictions on the choice of analysis methods.¹²

Remark 15.1.4 *A third class of methods tries to take advantage of the respective advantages of direct and iterative. Depending on the context, they are referred to as “hybrid methods” (HIPS, MaPhyS, etc.) or “Domain Decomposition (DD) methods” (FETI, Neumann-Neumann, etc.).*

Remark 15.1.5 *The two main families of methods should be seen as complementary rather than in competition. We often try to mix them together: DD methods, preconditioner by incomplete factorisation or multi-grid type, iterative refinement procedure at the end of the direct solver, etc.*

15.2 Conjugate gradient (CG) type iterative methods

15.2.1 Principle

15.2.1.1 Positioning of the problem

There is a host of iterative methods to resolve a linear system. But in practice, the most commonly used are:

- *Stationary methods*: Richardson, Jacobi, Gauss-Seidel, SSOR, etc.
- *Krylov methods*: GC, GMRES, BiCGStab, Orthomin, etc.

Here we will detail *the second family* which is the one actually used in code_Carmel (`LinearSolverType = 1, 2 or 3`). More specifically, the conjugate gradient (CG) algorithm. This algorithm developed by Hestenes and Steifel (1951) was ranked third in the “top 10” of the best numerical algorithms of the twentieth century¹³.

¹²In order, for example, to limit MPI communications and conditioning damage to the matrix system

¹³B.A.Cipra. The Best of the 20th century: editors name top 10 algorithms. SIAM News, 33-4 (2000).

In code `_Carmel`, the methods of the first family are used in addition to build a preconditioner (Jacobi preconditioner with `LinearSolverType=2`). Just like “more or less complete” factorisations constructed by direct solvers (see ILU(0) preconditioner and MUMPS preconditioner with, respectively, `LinearSolverType=1` and `3`).

When matrix \mathbf{K} of the system to be resolved (see equation 15.1) has the “good taste” to be symmetric positive definite (SPD in the English literature), it is shown, by differentiation, that the initial problem:

$$(P_1) \quad \mathbf{K} \mathbf{u} = \mathbf{f} \quad (15.2)$$

can also be formalised as the minimisation of a quadratic functional of the form:

$$(P_2) \quad \mathbf{u} = \underset{\mathbf{v} \in \mathbb{R}^N}{\text{Argmin}} J(\mathbf{v}) \quad (15.3)$$

$$\text{with: } J(\mathbf{v}) := \frac{1}{2} \langle \mathbf{v}, \mathbf{K} \mathbf{v} \rangle - \langle \mathbf{f}, \mathbf{v} \rangle = \frac{1}{2} \mathbf{v}^T \mathbf{K} \mathbf{v} - \mathbf{f}^T \mathbf{v}$$

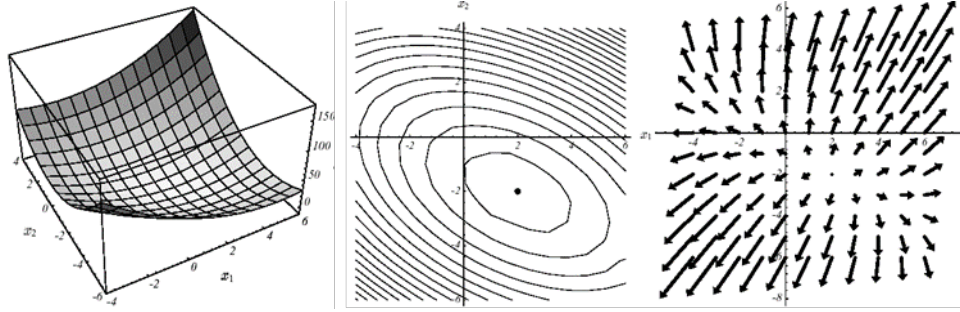


Figure 15.2: Example of J quadratic in $N=2$ dimensions

The figure above provides an example of J quadratic in $N=2$ dimensions with:

$$\mathbf{K} := \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}$$

and:

$$\mathbf{K} := \begin{bmatrix} 2 \\ -8 \end{bmatrix}$$

On the left is the graph of the functional, in the centre its level lines and, on the right, the gradient vectors.

The operator’s spectrum is $(\lambda_1; \mathbf{v}_1) = (7; [1, 2]^T)$ and $(\lambda_2; \mathbf{v}_2) = (2; [-2, 1]^T)$ ¹⁴

Due to the “positive definite” character of the matrix that makes J strictly convex, the cancelling vector ∇J corresponds to the only overall minimum \mathbf{u} . This is illustrated by the following relationship, valid regardless of \mathbf{K} symmetry:

$$J(\mathbf{v}) = J(\mathbf{u}) + \frac{1}{2} (\mathbf{v} - \mathbf{u})^T \mathbf{K} (\mathbf{v} - \mathbf{u}) \quad (15.4)$$

Thus, for any vector \mathbf{v} different from the solution \mathbf{u} , the positive definite character of the operator makes the second term strictly positive and hence \mathbf{u} is also an overall minimum.

This result, which is very important in practice, is based entirely on the famous positive-definite property of the working matrix, which is a little “ethereal”. For a two-dimensional problem it is possible to make a clear representation (see Figure 15.2): the paraboloid shape that focuses the unique minimum at the point $[2, -2]^T$ of zero slope.

¹⁴Figures taken from J. R. Shewchuck’s paper, with his kind permission. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain Carnegie Mellon University (1994).

15.2.1.2 Steepest Descent

Hence the idea behind the classic method best known by its English name “Steepest Descent”: we construct the sequence of iterates \mathbf{u}^i by following the direction in which J decreases the most, at least locally, i.e.:

$$\mathbf{d}^i = -\nabla J^i = \mathbf{r}^i$$

with:

$$J^i := J(\mathbf{u}^i)$$

and:

$$\mathbf{r}^i := \mathbf{f} - \mathbf{K} \mathbf{u}^i$$

At the i th iteration, we will thus seek to construct \mathbf{u}^{i+1} such that:

$$\mathbf{u}^{i+1} := \mathbf{u}^i + \alpha^i \mathbf{d}^i \quad (15.5)$$

and:

$$J^{i+1} < J^i \quad (15.6)$$

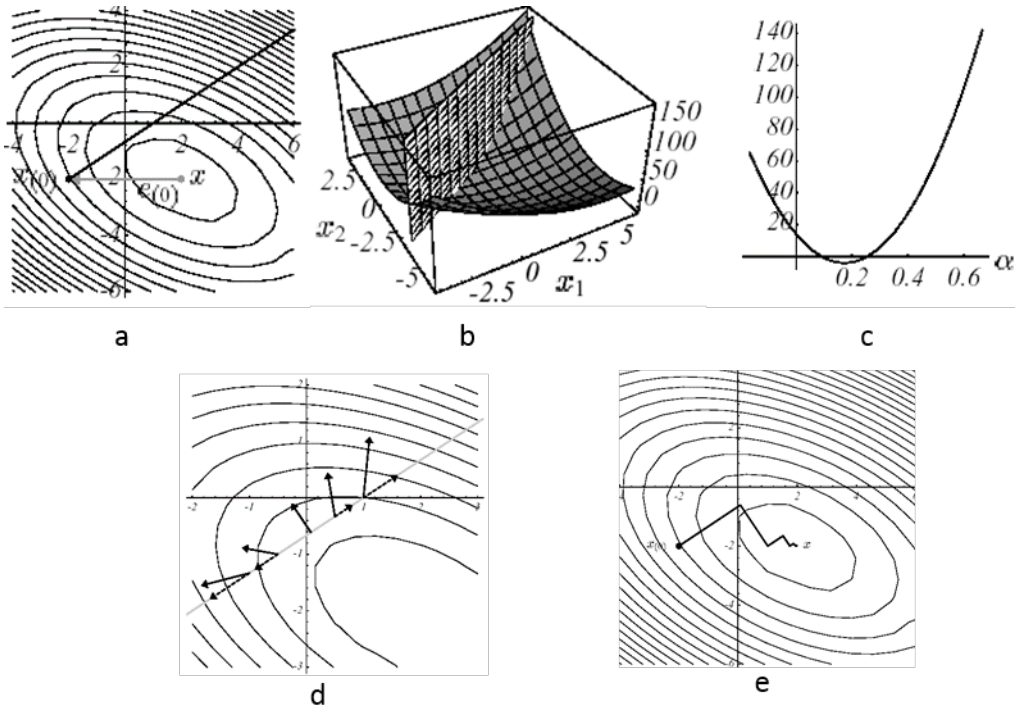


Figure 15.3: Illustration of Steepest Descent on example n° 1: initial descent direction (a), intersection of surfaces (b), corresponding parabola (c), gradient vectors and their projection along the initial descent direction (d) and overall process until convergence (e).

As a result of this formulation, we have thus transformed a quadratic minimisation problem of size N (in J and \mathbf{u}) into a one-dimensional minimisation (in G and α):

$$\begin{aligned} \text{Find } \alpha^i \text{ as } \alpha^i &= \underset{\alpha \in [\alpha_m, \alpha_M]}{\text{Argmin}} G^i(\alpha) \\ \text{avec } G^i &:= J(\mathbf{u}^i + \alpha \mathbf{r}^i) \end{aligned} \quad (15.7)$$

The figures above illustrate how this procedure works on example n° 1: starting from point $\mathbf{u}_0 = [-2, -2]^T$ (see (a)) we seek the optimal descent parameter, α_0 , along the line of steepest slope \mathbf{r}_0 ; this is equivalent to looking for a point belonging to the intersection of a vertical plane and a paraboloid (b), signified by the parabola (c). Trivially, this point cancels out the derivative of the parabola (d) :

$$\frac{\partial G^0(\alpha^0)}{\partial \alpha} = 0 \Leftrightarrow \langle \nabla J(\mathbf{u}^1), \mathbf{d}^0 \rangle = 0 \Leftrightarrow \langle \mathbf{d}^1, \mathbf{d}^0 \rangle = 0 \Leftrightarrow \alpha^0 := \frac{\|\mathbf{d}^0\|^2}{\langle \mathbf{d}^0, \mathbf{K} \mathbf{d}^0 \rangle} \quad (15.8)$$

This orthogonality between two successive residuals (i.e. successive gradients) produces a *characteristic path, called a “zigzag”*, towards the solution (e). Thus, in the case of a *poorly conditioned system* producing narrow and elongated ellipses¹⁵, the *number of iterations required can be considerable* (see Figure 15.3).

15.2.1.3 Principle of the conjugate gradient

To avoid this less-than-optimal zigzag path, a whole subset of descent methods known as “conjugate direction methods” has been developed. The CG algorithm belongs to this subset of methods. These recommend the progressive construction of descent directions $\mathbf{d}^0, \mathbf{d}^1, \mathbf{d}^2$, etc., linearly independent so as to avoid the zigzags of the conventional descent method.

So what linear combination should be used to construct the new direction of descent at step i ? Knowing, of course, that it must take account of two crucial pieces of information: the value of the gradient $\nabla J^i = -\mathbf{r}^i$ and of the directions $\mathbf{d}^0, \mathbf{d}^1, \dots, \mathbf{d}^{i-1}$.

$$? \quad \mathbf{d}^i = \alpha_i \mathbf{r}^i + \sum_{j < i} \beta^j \mathbf{d}^j \quad (15.9)$$

*The trick is to choose a vector independence of type **K**-orthogonality* (as the working operator is SPD, it does define a scalar product through which two vectors can be orthogonal, see Figure 3.1-4)

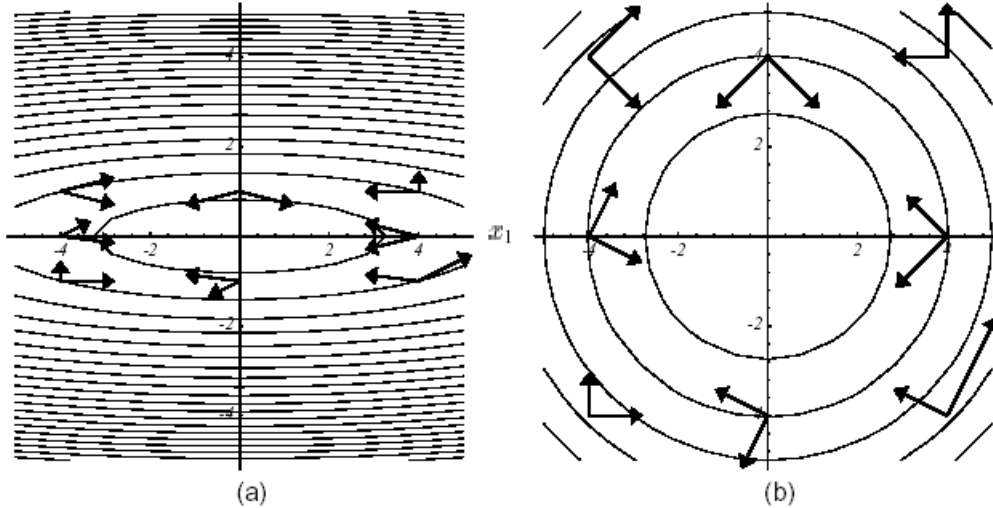


Figure 15.4: Example of vector pairs **K**-orthogonal in 2D: conditioning of any **K** (a), perfect conditioning (i.e. equal to 1) = usual orthogonality (b).

We can therefore *accept a linear combination* of the type:

¹⁵The conditioning of the operator **K** is written as the ratio of its extreme eigenvalues $\eta(\mathbf{K}) = \frac{\lambda_{max}}{\lambda_{min}}$ which are themselves proportional to the axes of the ellipses. Hence the direct, visual link between poor matrix conditioning and the narrow, tortuous valley where minimisation is mishandled.

$$\mathbf{d}^i := \mathbf{r}^i + \beta^i \mathbf{d}^{i-1} \quad (15.10)$$

We thus show that the one-dimensional search (see equation 15.7) takes place in an optimal space: the plane formed by the two orthogonal directions $(\mathbf{r}^i, \mathbf{d}^{i-1})$.

It thus remains to determine the *optimal value of the proportionality coefficient* β^i . In CG, this choice is made in such a way as to maximise the attenuation factor (ratio between the error at iteration $i - 1$ and that at iteration i , expressed with the matrix norm associated with \mathbf{K}):

$$\frac{\|\mathbf{u} - \mathbf{u}^i\|_{\mathbf{K}}^2}{\|\mathbf{u}^{i-1} - \mathbf{u}\|_{\mathbf{K}}^2} = \frac{\langle \mathbf{r}^i, \mathbf{d}^i \rangle^2}{\langle \mathbf{K}^{-1} \mathbf{r}^i, \mathbf{K}^{-1} \mathbf{d}^i \rangle} \quad (15.11)$$

It leads to the expression:

$$\beta^i := \frac{\|\mathbf{r}^i\|^2}{\|\mathbf{r}^{i-1}\|^2} \quad (15.12)$$

and induces the same orthogonal property of successive residuals as for the Steepest Descent (*but without the zigzags!*):

$$\langle \mathbf{r}^i, \mathbf{r}^{i-1} \rangle = 0 \quad (15.13)$$

Adding a “residual-dd” condition:

$$\langle \mathbf{r}^i, \mathbf{d}^i \rangle = \|\mathbf{r}^i\|^2 \quad (15.14)$$

which requires initialising the process via:

$$\mathbf{d}^0 = \mathbf{r}^0$$

15.2.1.4 Conjugate gradient algorithm

In short, by recapitulating the previous relations, we arrive at the classic algorithm (3.1-1) below.

Initialisation \mathbf{u}^0 given, $\mathbf{r}^0 = \mathbf{f} - \mathbf{K} \mathbf{u}^0$ $\mathbf{d}^0 = \mathbf{r}^0$

Loop in i

- | | | |
|-----|------------------------------------------------------------------------------------|-------------------------------|
| (1) | $\mathbf{z}^i = \mathbf{K} \mathbf{d}^i$ | |
| (2) | $\alpha^i = \frac{\ \mathbf{r}^i\ ^2}{\langle \mathbf{d}^i, \mathbf{z}^i \rangle}$ | (optimal descent parameter) |
| (3) | $\mathbf{u}^{i+1} = \mathbf{u}^i + \alpha^i \mathbf{d}^i$ | (new iterate) |
| (4) | $\mathbf{r}^{i+1} = \mathbf{r}^i - \alpha^i \mathbf{z}^i$ | (new residual) |
| (5) | Stop test via $\ \mathbf{r}^{i+1}\ $ | (for example) |
| (6) | $\beta^{i+1} = \frac{\ \mathbf{r}^{i+1}\ ^2}{\ \mathbf{r}^i\ ^2}$ | (optimal conjugate parameter) |
| (7) | $\mathbf{d}^{i+1} = \mathbf{r}^{i+1} + \beta^{i+1} \mathbf{d}^i$ | (new direction of descent) |

Table 15.1: Conjugate gradient (CG) algorithm.

In example n° 1, the “supremacy” of CG over Steepest Descent is clear (see Figure 15.5). In both cases, the same starting points and stopping criteria were chosen: $\mathbf{u}^0 = [-2, -2]^T$ and $\|\mathbf{r}^i\|^2 < \varepsilon = 10^{-6}$.

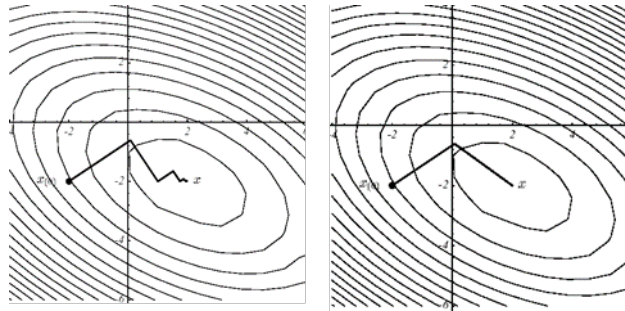
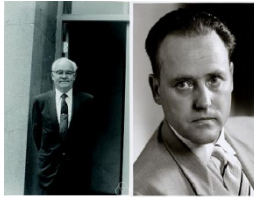


Figure 15.5: Comparison of convergence, in example n° 1, for Steepest Descent, on the left, and CG, on the right.

In practice, we often use this algorithm on systems that are not necessarily SPD and even singular systems. This may be the case with `code_Carmel`. Convergence is then slowed down and the robustness of the process is not guaranteed. It may diverge! But often, in `code_Carmel` (as in `Code_Carmel3D`, `TELEMAC`, `Syrthes`, etc.), the algorithm behaves rather well even when it is used “out of scope”. Especially when we make the effort to provide it with a well-conditioned matrix (non-dimensional equation, no flattened meshes, etc.) and a second member that respects the Fredholm alternative ($\mathbf{f} \in \text{Im}(\mathbf{K})$).

Remark 15.2.1 *This CG method was developed in 1951 by M. R. Hestenes (left) and E. Stiefel (right) of the National Bureau of Standards in Washington D.C. (a breeding ground for numerical analysts, also including C. Lanczos). See portraits opposite.*



Remark 15.2.2 *The first theoretical results on convergence are due to the work of S. Kaniel (1966) and H. A. Van der Vorst (1986) and it was really popularised for solving large sparse systems by J. K. Reid (1971). Interested readers will find an annotated history and an exhaustive bibliography on the subject in the papers by G. H. Golub, H. A. Van der Vorst and Y. Saad.¹⁶*

Remark 15.2.3 *Instead of a stopping test based on the norm of the residual, which is theoretically permissible but in practice can be difficult to calibrate, we often prefer a non-dimensional stopping criterion, such as the residual relative to the i th iteration:*

$$\delta^i := \frac{\|\mathbf{r}^i\|}{\|\mathbf{f}\|}$$

This is what is done in particular in `Code_Carmel(3D)`, `Code_Aster` and `TELEMAC`.

15.2.2 Preconditioned conjugate gradient (PCG)

15.2.2.1 Principles

As we have seen (and hammered home!) in previous sections, *the speed of convergence* of the conjugate gradient depends on the conditioning of the matrix $\eta(\mathbf{K})$. The closer it is to its floor value, 1, the better the convergence.

¹⁶Golub et al. Some history of the conjugate gradient and Lanczos algorithms: 1948–1976. SIAM review, 31-1 (1989). Closer to the solution: iterative linear solvers. The state of the art in numerical analysis. Ed. Clarendon Press (1997). Y. Saad & H. A. Van Der Vorst. Iterative solution of linear systems in the 20th-century. J.Comp.Appl.Math., 123 (2000).

The principle of preconditioning is thus “posed”, it consists in replacing the linear system of the problem (P_1) (equation 15.1) by an equivalent system of the type:

$$(\tilde{P}_1) \quad \underbrace{\mathbf{M}^{-1} \mathbf{K}}_{\tilde{\mathbf{K}}} \mathbf{u} = \underbrace{\mathbf{M}^{-1} \mathbf{f}}_{\tilde{\mathbf{f}}} \quad (15.15)$$

such that, ideally:

- *The conditioning* is clearly *improved*¹⁷: $\eta(\tilde{\mathbf{K}}) \ll \eta(\mathbf{K})$.
- Just like the spectral distribution: more packed eigenvalues.
- \mathbf{M}^{-1} is inexpensive to evaluate (as with the initial operator, we often just need to know the action of the preconditioner on a vector): $\mathbf{M} \mathbf{v} = \mathbf{u}$ easy to invert.
- \mathbf{M}^{-1} easy to implement and, possibly, efficient to parallelise.
- \mathbf{M}^{-1} is fairly sparse because the aim is to limit the additional memory requirement.
- \mathbf{M}^{-1} keeps the working matrix $\tilde{\mathbf{K}}$ with the same properties as the original (here, the SPD character).

In theory, the best choice would be $\mathbf{M}^{-1} = \mathbf{K}^{-1}$ because then $\eta(\tilde{\mathbf{K}} = \mathbf{I}_N) = 1$, but if you have to completely invert the operator by a direct method to construct this preconditioner, it is of little practical interest!

However, we will see later that this idea is not as far-fetched as that (see ILU(0) and relaxed single precision MUMPS preconditioners). Especially when we seek to optimise not just a “one-shot” resolution, but a whole succession of resolutions within a non-linear process.

In other words, *the purpose of a preconditioner is to compress, at a lower cost*¹⁸, *the spectrum of the working operator*. Thus, as already mentioned, its “effective conditioning” will be improved in tandem with the convergence of the PCG.

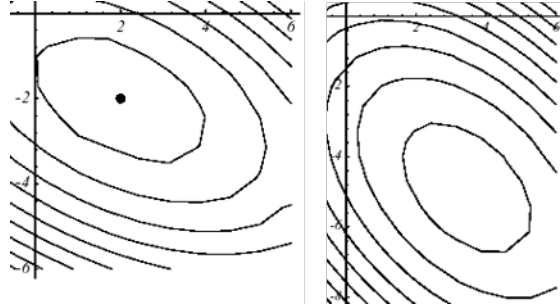


Figure 15.6: Effect of diagonal preconditioning (Jacobi) on the paraboloid of example n° 1: left, without $\eta(\mathbf{K}) = 3.5$; on the right with $\eta(\tilde{\mathbf{K}}) = 2.8$

Graphically, this means that the graph of the quadratic form is more spherical. Even on an $N=2$ dimensional system and with a “defective” preconditioner (see Figure 15.6), the effects are noticeable.

In absolute terms, we *can precondition a linear system from the left* (“left preconditioning”), from *the right* (“right preconditioning”) or by a mixture of the two (“split preconditioning”). It is the last version that will be adopted for our SPD operator, as we cannot directly apply the CG to resolve (\tilde{P}_1): even if \mathbf{M}^{-1} and \mathbf{K} are SPD, this is not necessarily the case for their product.

¹⁷This theoretical property, just like the next, is very rarely demonstrated. They are often only supported by numerical experiments.

¹⁸Memory, computation time, robustness, even maintenance/ergonomics.

The trick then consists in using an SPD preconditioning matrix, \mathbf{M} , for which we will be able to define another matrix (\mathbf{M} being real symmetric, it is diagonalisable in the form $\mathbf{M} = \mathbf{U} \mathbf{D} \mathbf{U}^T$ with $\mathbf{D} := \text{diag}(\lambda_i)$, $\lambda_i > 0$ and \mathbf{U} orthogonal matrix). The SPD matrix sought thus comes from the associated decomposition $\mathbf{M}^{1/2} = \mathbf{U} \text{diag}(\sqrt{\lambda_i}) \mathbf{U}^T$ with $\mathbf{M}^{1/2}$ defined such that $(\mathbf{M}^{1/2})^2 = \mathbf{M}$. Hence the new problem, this time SPD:

$$\left(\hat{P}_1 \right) \underbrace{\mathbf{M}^{-1/2} \mathbf{K} \mathbf{M}^{-1/2}}_{\mathbf{K}} \underbrace{\mathbf{M}^{1/2} \mathbf{u}}_{\hat{\mathbf{u}}} = \underbrace{\mathbf{M}^{-1/2} \mathbf{f}}_{\hat{\mathbf{f}}} \quad (15.16)$$

to which we can apply the standard CG algorithm to create what we call a Preconditioned Conjugate Gradient (PCG).

15.2.2.2 PCG algorithm

By substituting in the algorithm 15.1, the expressions of the previous problem (\hat{P}_1) and by working to simplify the whole to manipulate only expressions in \mathbf{K} , \mathbf{u} and \mathbf{f} , the result is as follows:

Initialisation	\mathbf{u}_0 given $\mathbf{r}^0 = \mathbf{f} - \mathbf{K} \mathbf{u}^0$, $\mathbf{d}^0 = \mathbf{M}^{-1} \mathbf{r}^0$	
Loop in i		
(1)	$\mathbf{z}^i = \mathbf{K} \mathbf{d}^i$	
(2)	$\alpha^i = \frac{\langle \mathbf{r}^i, \mathbf{g}^i \rangle}{\langle \mathbf{d}^i, \mathbf{z}^i \rangle}$	(optimal descent parameter)
(3)	$\mathbf{u}^{i+1} = \mathbf{u}^i + \alpha^i \mathbf{d}^i$	(new iterate)
(4)	$\mathbf{r}^{i+1} = \mathbf{r}^i - \alpha^i \mathbf{z}^i$	(new residual)
(5)	Stop test via $\ \mathbf{r}^{i+1}\ $	(for example)
(6)	$\mathbf{g}^{i+1} = \mathbf{M}^{-1} \mathbf{r}^{i+1}$	(preconditioned residual)
(7)	$\beta^{i+1} = \frac{\langle \mathbf{r}^{i+1}, \mathbf{g}^{i+1} \rangle}{\langle \mathbf{r}^i, \mathbf{g}^i \rangle}$	(optimal conjugate parameter)
(8)	$\mathbf{d}^{i+1} = \mathbf{g}^{i+1} + \beta^{i+1} \mathbf{d}^i$	(new direction of descent)

Table 15.2: Preconditioned conjugate gradient (PCG) algorithm.

But in fact, the *symmetric character of the initial preconditioned problem* (\hat{P}_1) is all relative. It is inseparable from the underlying scalar product. If, instead of taking the usual Euclidean scalar product, we use a matrix scalar product defined with respect to \mathbf{K} , \mathbf{M} , \mathbf{M}^{-1} , it is possible to make the preconditioned problem symmetric even though it was not initially. As with Krylov methods in modal, it is the (working operator, scalar product) pair that needs to be modulated to adapt to the problem!

Thus, $\mathbf{M}^{-1} \mathbf{K}$ being symmetric with respect to the \mathbf{M} -scalar product, this new working operator and this new scalar product can be substituted in the non-preconditioned CG algorithm (see algorithm 15.1):

$$\begin{aligned} \mathbf{K} &\Leftarrow \mathbf{M}^{-1} \mathbf{K} \\ \langle, \rangle &\Leftarrow \langle, \rangle_{\mathbf{M}} \end{aligned} \quad (15.17)$$

And (what a surprise!) by working with the expressions a little, we find exactly the previous PCG algorithm (see algorithm 15.2). We do the same with a right preconditioning, $\mathbf{K} \mathbf{M}^{-1}$, via a \mathbf{M}^{-1} -scalar product. Hence, right, left or “split SPD style” preconditioning all lead rigorously to

the same algorithm. This observation is used when the preconditioners do not conform to the ideal scenario (\hat{P}_1) : this is exactly the case with the preconditioners in code_Carmel and Code_Aster.

Remark 15.2.4 *This variant of PCG, which is by far the most widespread, is sometimes referred to in the literature as the “untransformed preconditioned conjugate gradient”. As opposed to the transformed version, which manipulates the entities specific to the new formulation.*

15.2.2.3 PCG in code_Carmel

The PCG algorithm implemented in the code is very similar to the one described in 15.2.. We just note that its stopping criterion is relative to the norm of the initial second member and that the initial estimate is taken to be equal to zero¹⁹. ($\mathbf{u}^0 = 0$).

Its maximum number of iterations is configured using `nbIterationMax` and the stopping criterion, now based on the norm of the residual (and no longer the squared norm), is controlled via `kEpsilonGCP`. This last point is important because it is easier (and permitted!), in double precision arithmetic, to control a parameter typically changing between 10^{-9} and 10^{-6} than its squared value which thus changes between 10^{-18} and 10^{-12} .

The recommended values for these parameters are 300 and 10^{-6} , respectively. With some thought required when it is considered that a large number of iterations are needed (>1000) or the stopping criterion is extreme: $>10^{-9}$ or $<10^{-3}$. For more information, see section 7.2 of [Boiteau 2014].

More recently, a software project has been launched to rationalise and pool sources and make a number of minor improvements:

- Pre-testing algorithm control parameters
- Following up on the occurrence of an error or warning;
- Premature stop and exit with $\mathbf{u}^{\text{sol}}=0$ (solution vector), if the Euclidean norm of the second member is below the machine accuracy.

Remark 15.2.5 *Code_Carmel now also provides an algorithm for non-preconditioned PCG ($\mathbf{M} = \mathbf{Id}$, `LinearSolverType` = 0). It is only used to make comparisons and for validations (numerical and computer-related).*

Remark 15.2.6 *In case of non-convergence, the PCG in Code_Carmel stops on the fatal error: `kErreurConvergenceGCP`. Otherwise, the error code is `kAucuneErreur`. If the non-linear/linear optimisation strategy is enabled (via `reacprecond_methodeNL`), the error code can also be set to `kErreurReacPrecond` to alert the Newton algorithm of the need to recalculate the tangent matrix.*

Remark 15.2.7 *The change in the residual until convergence can be tracked on the screen, in a convergence bar or in a file. These options are controlled by the parameters:*

- `kAfficheBarreConvergence`;
- `kSauveConvergence`;
- `descripteurFichierConvergence`.

¹⁹This choice may seem a little counter-intuitive, especially when successively solving a Newton algorithm, but in practice it is often the one that is favoured (see Code_Aster, TELEMAT). Starting from the origin does not bias the search and provides, on average, the best compromise between time and robustness. The only case where the initial iterate would be something other than the origin would be in the case of restart techniques (to control a loss of orthogonality or to manage problems with multiple second members).

15.2.3 Range of preconditioners available in code_Carmel

Among the myriad of possible preconditioners, only three possibilities have been retained in the code:

- Jacobi (`LinearSolverType` = 2).
- Crout ILU(0) (`LinearSolverType` = 1).
- Single precision MUMPS (`LinearSolverType` = 3) and possibly relaxed²⁰ (if the condition `mumps_relax` > 0 is respected). This preconditioner can only be enabled if the code_Carmel program has been linked with MUMPS (see `makefile` and variable `USE_MUMPS`).

These preconditioners are listed here *in ascending order of memory consumption, robustness, and complexity* (numerical and IT). Often the efficiency in terms of number of iterations follows the same hierarchy. For example, for Rubinacci's cube (see Table 3.2-1), we have, whatever the stop criterion, a number of iterations that ranges from a few hundred (for Jacobi) to a hundred (for Crout) and then to only a few iterations (for MUMPS). However, a MUMPS iteration costs much more in time and memory than a Crout iteration. So, there is a compromise to be found.

These preconditioners are based on the following strategy, which is in fact only verified on a few canonical problems, but which often proves to “pay off”, even for industrial problems:

- As we move further away from the main diagonal, the orders of magnitude of the terms decrease.
- Terms of small order of magnitude play little part in the calculation.

Based on these “axioms”, all “moves are thus allowed” to construct an approximation of \mathbf{K}^{-1} at the lowest possible cost:

- With Jacobi: $\mathbf{M}_{\text{Jacobi}} = \text{diag}(\mathbf{K})$ ²¹.
- With Crout: $\mathbf{M}_{\text{Crout}} = \mathbf{L} \mathbf{D} \mathbf{L}^T$ not taking account of factorisation fill ($\text{profil}(\mathbf{M}_{\text{Crout}}) = \text{profil}(\mathbf{K})$ ²²).
- With Mumps: $\mathbf{M}_{\text{MUMPS}} = \text{simple_précision}(\mathbf{L} \mathbf{D} \mathbf{L}^T)$ previously filtering out extra-diagonal terms that are too small. But here the profile of $\mathbf{M}_{\text{MUMPS}}$ can be much larger than that of \mathbf{K} hence a bigger memory requirement (even in single precision).

code_Carmel thus offers a whole continuum of preconditioners for which the calculation cost and memory cost can be adjusted according to the available resources and the difficulty of the problem. Knowing that in case of non-convergence, the benchmark linear solver is always available: MUMPS as a direct solver (see section 15.3).

In non-linear, we can also take great advantage of pooling, between several tens of iterations of the non-linear solver (often a Newton algorithm), of the construction of the preconditioner. The non-linear process may require more iterations, but in the end, as these are faster, the user often wins!

This strategy is especially beneficial for the most costly combination: PCG + MUMPS preconditioner. It is activated via the parameter `reacprecond_methodeNL`. With a strictly positive value of this keyword (e.g. 30), the preconditioner is recalculated only if:

- The PCG has been through more than `reacprecond_methodeNL` iterations for a given iteration of the non-linear solver.

²⁰I.e. before factorising the matrix in single precision, we “sparsify” it, we make it sparse by removing extra-diagonal terms that are too small.

²¹A matrix whose unique, possibly non-zero terms match the diagonal terms of \mathbf{K} .

²²The profile here is the set of non-zero terms in a sparse matrix.

Type of preconditioner	number of iterations
Without (<code>LinearSolverType=0</code>)	965
Jacobi (<code>LinearSolverType=2</code>)	360
Crout ILU(0) (<code>LinearSolverType=1</code>)	179
MUMPS SP very relaxed (<code>LinearSolverType=3 + mumps_relax = 10⁻³</code>)	6
... moderately relaxed (<code>... + mumps_relax = 10⁻⁴</code>)	4
... slightly relaxed (<code>... + mumps_relax = 10⁻⁵</code>)	2
... not relaxed (<code>... + mumps_relax < 0</code>)	2

Table 15.3: Number of PCG iterations on the test case of Rubinacci’s cube with $\varepsilon = 10^{-9}$ (Code_Carmel v1.7.6 on a 7-caliber station).

- That makes at least `reacprecond_method`NL iterations of the non-linear solver without this re-calculation.
- The residual of the non-linear solver increases rather than decreases²³.

Remark 15.2.8 *The specialist literature offers many preconditioners: explicit (polynomial, SPAI, AINV, etc.), implicit (Schwarz, IC, etc.), multi-Level (domain decomposition, multi-grid, etc.). Some are dedicated to one application, others are more general. “Fashion” has also played a role! Further information can be found, for example, in works by G. Meurant, Y. Saad, H. A. Van der Vorst, J. W. Demmel, G. W. Stewart, etc.*

Remark 15.2.9 *For the moment, the preconditioner is being pooled while the tangent matrix continues to be updated (for nothing). In the future, we will also be able to avoid the unnecessary extra cost of elementary calculations and assemblies. This strategy will often pay off, even for preconditioners with a very low cost (such as Crout or Jacobi). For the moment, the current redesign of the tangent matrix construction routines and the lack of modularity of those present in v1.7.3 have not enabled us to make any progress on this point.*

15.2.3.1 Jacobi preconditioner

The first option is often offered in codes for its ease of implementation, its good ratio of “numerical effectiveness to additional computation cost” for problems that are not too badly conditioned, and its very good scalability (in parallel mode).

It consists in preconditioning the initial operator by the diagonal:

$$\mathbf{M}_{\text{Jacobi}} := \text{diag}(\mathbf{K}) \quad (15.18)$$

This is called diagonal or Jacobi preconditioning (JCG for Jacobi Conjugate Gradient) by reference to the stationary method of the same name. Since its memory overhead compared with basic CG is negligible, in the order of $\mathcal{O}(N)$, it would be a mistake to forgo the acceleration it offers. Even though the latter may be modest.

On the other hand, since in Code_Carmel, the resolved system is modified “in place” to take this preconditioning into account, the additional computation cost of the preconditioning step (step (6) of 15.2) is zero. We only “pay” for the initial and final transformation of the problem.

²³This criterion could be made more stringent by replacing it with “the residual has not decreased by at least x%” with x=20 or 30%

Remark 15.2.10 *This is a solution often chosen in CFD (for EDF R&D: Code_Saturne, TELEMAC). In these fields, great attention is paid to the non-linear solution method and the construction of the mesh so that they produce a well-conditioned problem. This is what has made JCG so famous, transformed for the occasion into a veritable “speed demon” reserved for large parallel computers.*

Remark 15.2.11 *It is a solution that has not, however, been adopted in Code_Aster because of its lack of robustness in thermo-mechanical industrial studies.*

Remark 15.2.12 *Until v1.7.6, the use of this preconditioner in Code_Carmel had a bug. At the output of the conjugate gradient we did not correctly reconvert the work problem so that the matrix and the second member no longer contained any trace of the preconditioning (inverse conversion of equation 15.16). This could potentially produce false results if these elements were used for post-processing. In contrast, the running of a Newton or the ODE solver was not affected, as these reinitialise the linear system for each of their iterations. This bug was fixed as part of this software project.*

15.2.3.2 Crout preconditioner

The second option is preconditioning by an incomplete Cholesky factorisation $IC(0)$ (also known as a Cholesky-Crout factorisation). Since the initial operator is assumed to be SPD, it allows a Cholesky decomposition of type $\mathbf{K} = \mathbf{C} \mathbf{C}^T$ where \mathbf{C} is a lower triangular matrix. An incomplete Cholesky factorisation is the search for a lower triangular matrix \mathbf{F} as sparse as possible and such that $\mathbf{F} \mathbf{F}^T$ is close to \mathbf{K} in a direction to be defined. For example, by taking $\mathbf{B} = \mathbf{K} - \mathbf{F} \mathbf{F}^T$, we will ask that the relative error (expressed in a matrix norm of choice):

$$\Delta := \frac{\|\mathbf{B}\|}{\|\mathbf{K}\|} \quad (15.19)$$

is as small as possible. On reading this “rather evasive” definition, the profusion of possible scenarios can be seen. Everyone came up with their own incomplete factorisation! The work of G. Meurant²⁴, among others, shows the great diversity: $IC(n)$, $MIC(n)$, relaxed, reordered, by blocks, etc.

So, this option consists in taking as a preconditioner:

$$\mathbf{M}_{\text{Crout}} := (\mathbf{F}) (\mathbf{F})^T \quad (15.20)$$

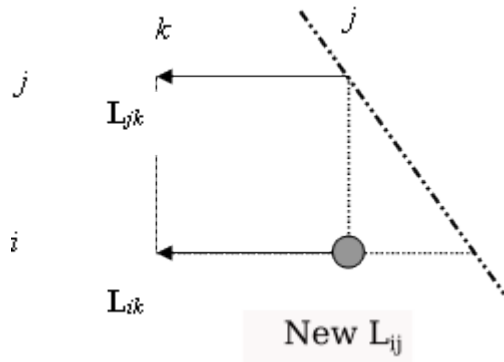


Figure 15.7: Fill-in phenomenon during factorisation.

However, to simplify the task, we often impose *a priori* the sparse structure of \mathbf{F} , i.e. its graph (the triangular part of what was previously called the profile):

²⁴G. Meurant. Computer solution of large linear systems. Ed. Elsevier (1999).

$$\Xi(\mathbf{F}) := \{(i, j), 1 \leq j \leq i-1, 1 \leq i \leq N, F_{ij} \neq 0\} \quad (15.21)$$

It is obviously a question of finding a compromise: the further this graph is extended, the smaller the error (see equation 15.19), but the more costly the calculation and storage of what is (in our case) only a preconditioner. Usually, the preconditioners are recursive and at their basic level, they impose on \mathbf{F} the same structure as that of \mathbf{K} : $\Xi(\mathbf{F}) = \Xi(\mathbf{K})$.

Because Code_Carmel systems are not usually SPD but only symmetric, this concept is generalised to LU factorisation. This is called the ILU preconditioner for “Incomplete LU”.

The factorisation is constructed line by line using the usual formula:

$$L_{ij} = \frac{1}{D_j} \left(K_{ij} - \sum_{k=1}^{j-1} L_{ik} D_k L_{jk} \right) \quad (15.22)$$

Hence the phenomenon of progressive fill-in of the profile: initially matrix \mathbf{L} has the same fill-in as matrix \mathbf{K} , but in the course of the process, a zero term in K_{ij} may correspond to a non-zero term in L_{ij} . It suffices that there is a column k ($< j$) with a non-zero term for rows i and j (see Figure 15.7).

These non-zero terms may themselves correspond to previous fill-in, giving rise to a concept of recursivity that can be interpreted as so many “levels” of fill-in. We thus talk of a level 0 incomplete factorisation (stored in $\mathbf{L}(0)$) if it identically reproduces the structure (but not of course the values, which are different) of the strict lower diagonal part of \mathbf{K} (i.e. the same graph). Level 1 factorisation (respectively $\mathbf{L}(1)$) may include the fill-in from non-zero terms of \mathbf{K} , level 2 (respectively $\mathbf{L}(2)$) may mix in new previous non-zero terms to form possible new terms, and so on recursively.

In Code_Carmel, we have limited ourselves to level 0. In Code_Aster, problems usually being much less well conditioned, the user is allowed several levels of fill-in²⁵.

Remark 15.2.13 *Since the matrix is no longer SPD but simply regular symmetric, it is not certain, a priori, that a factorisation $\mathbf{L}\mathbf{D}\mathbf{L}^T$ exists without the use of permutations of rows and columns ($\mathbf{P}\mathbf{K} = \mathbf{L}\mathbf{D}\mathbf{L}^T$ with \mathbf{P} permutation matrix). A management scenario for these pivots has been planned in Code_Carmel (parameters `kPivotsCrout` and `kEpsilonPivotsCrout`). But this scenario can sometimes go wrong. In this case, the code stops on a fatal error: `kErreurPreconditionneur`.*

Remark 15.2.14 *Strictly speaking, we should talk of ILDLT-type incomplete factorisation, but in the literature and in code documentation, ILU and IC, and even their variants, are already often mixed up, so there’s no need to add to the list of acronyms! This documentation will refer indifferently to Crout, IC(0) or ILU(0) factorisation.*

15.2.3.3 MUMPS preconditioner

When linking Code_Carmel to the external product MUMPS (see paragraph 15.3.5.2), it can be used as a direct solver (double precision) or as a preconditioner (single precision). With this scenario, we benefit from a more robust solution than the previous two preconditioners but potentially at a greater cost (in CPU and especially in peak memory requirement).

Even if the user can relax the extra-diagonal matrix terms provided to “MUMPS preconditioner” by adjusting the parameter `mumps_relax` (see formula 15.23 and Figure)

$$\begin{aligned} &\text{If } \text{mumps_relax} > 0 \text{ and } i \neq j, |\mathbf{K}_{ij}| < \text{mumps_relax} (|\mathbf{K}_{ii}| + |\mathbf{K}_{jj}|) \Rightarrow \tilde{\mathbf{K}}_{ij} = 0 \\ &\text{Else if } \tilde{\mathbf{K}}_{ij} = \mathbf{K}_{ij} \end{aligned} \quad (15.23)$$

²⁵Empirically, we find that the CPU and memory costs double, at least, between each level. In general, the fill-in factor ranges from 10 to 100. This leaves some room for manoeuvre on this parameter. To really take advantage of this preconditioner we limit ourselves to a maximum of 3 levels. Otherwise, you might as well use the MUMPS preconditioner!

We thus construct, from \mathbf{K} , the working matrix $\tilde{\mathbf{K}}$ that will serve as input data for the single precision MUMPS factorisation. So, this option consists in taking as a preconditioner:

$$\mathbf{M}_{\text{MUMPS}} := \text{simple_precision}(\tilde{\mathbf{L}}\tilde{\mathbf{D}}\tilde{\mathbf{L}}^T) \quad (15.24)$$

It extends the previous concept of incomplete factorisation of the preconditioner. The fact that most of the terms of the factorisation are preserved and that the tool handles a whole series of numerical difficulties (pivoting, singularities, heterogeneity of the orders of magnitude of the terms, etc. see section 15.3.5.3) gives it *great robustness even on difficult problems*.

It is for these reasons that this preconditioner has met with great success in Code_Aster thermo-mechanical simulations. It is “armed” to handle the diversity of situations and a wide range of numerical difficulties.

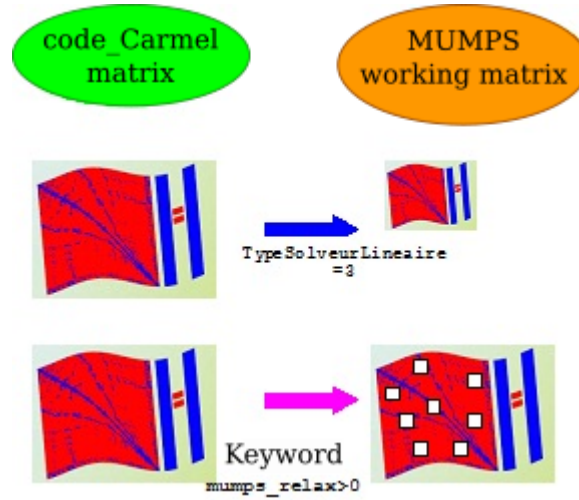


Figure 15.8: Advanced features using MUMPS as a preconditioner: mixing of single/double precision calculations and matrix filtering.

This strategy therefore provides an industrial solution that is viable, at least twice less intensive in CPU and peak RAM use than the direct solver strategy. And it takes advantage of all the numerical improvements and acceleration of its “direct big brother”:

- To reduce peak RAM use, in addition to lowering `mumps_relax`, much of the memory can be dumped to disk (`mumps_memory = 'OOC'`), change of re-numberer to reduce fill-in (`mumps_renum`) or, eventually, increase the number of processors (distributed parallelism via MPI). Even if the recommended values of these parameters, to optimise both time and peak memory, are instead, respectively, 10-6, “IC” and “AUTO”.
- To reduce computation time, we should try to keep all MUMPS objects in RAM²⁶ (`mumps_memory = 'IC'`) and definitely unplug all quality controls²⁷. (`mumps_post = 'OFF'` et `kEpsilonMUMPS < 0`).

²⁶However, if there is not enough memory per processor, it is better to take advantage of the optimised management of MUMPS Out-Of-Core rather than leave the system to “swap”. This type of highly unfavourable behaviour is immediately obvious from a very large gap between CPU time and elapsed time (shown via `Imonitoring_systeme`). If system monitoring is enabled, a warning is usually issued to alert the user on this point. In order to avoid these losses of time, it is best to pre-estimate the memory consumption of each of the alternatives (via the option `LinearSolverType = 5`) and thus relaunch the calculation with full knowledge of the facts.

²⁷Part of the quality of the solution (reverse error) is ensured by the PCG stop criterion. And in any case, we chose from the outset to water down the resolution (single precision and relaxation), hence there is no need to be very precise about this deliberately approximate working problem!

15.3 Direct methods

15.3.1 Principle

15.3.1.1 Factorisation

The basic idea of direct methods is to *decompose the problem matrix \mathbf{K} into a product of special matrices* (lower and upper triangular, diagonal) that are easier to “invert”. This is called the *factorisation*²⁸ of the working matrix:

- If \mathbf{K} is SPD, it permits the unique Cholesky factorisation: $\mathbf{K} = \mathbf{L}\mathbf{L}^T$ with \mathbf{L} lower triangular.
- If \mathbf{K} is arbitrarily symmetric and regular, it permits at least one “factorisation $\mathbf{L}\mathbf{D}\mathbf{L}^T$ ”: $\mathbf{P}\mathbf{K} = \mathbf{L}\mathbf{D}\mathbf{L}^T$ with \mathbf{L} lower triangular with diagonal coefficients equal to 1, \mathbf{D} a diagonal matrix and \mathbf{P} a permutation matrix.
- If \mathbf{K} is arbitrary and regular, it allows at least one “factorisation $\mathbf{L}\mathbf{U}$ ”: $\mathbf{P}\mathbf{K} = \mathbf{L}\mathbf{U}$ with \mathbf{L} lower triangular with diagonal of 1, \mathbf{U} upper triangular \mathbf{P} a permutation matrix.

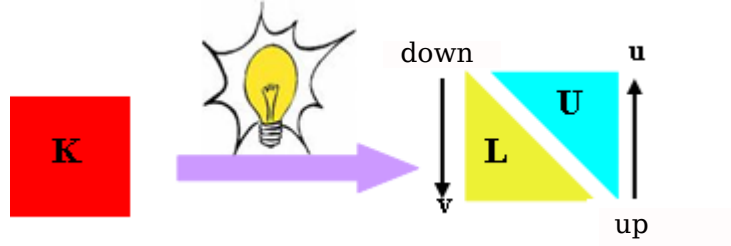


Figure 15.9: Principle of direct methods.

Remark 15.3.1 For example, the symmetric and regular matrix \mathbf{K} below decomposes into the following form $\mathbf{L}\mathbf{D}\mathbf{L}^T$ (without the need for permutation here, $\mathbf{P} = \mathbf{I}_d$)

$$\mathbf{K} := \begin{bmatrix} 10 & 20 & 30 \\ 45 & 80 & 171 \\ \text{sym} & & \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 4 & 1 \end{bmatrix}}_{\mathbf{L}} \underbrace{\begin{bmatrix} 10 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{D}} \underbrace{\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{L}^T} \quad (15.25)$$

15.3.1.2 Down- up

Once this decomposition is complete, *the resolution of the problem* is greatly facilitated. It can now only be expressed in the form of the simplest linear resolutions there are: based on triangular or diagonal matrices. These are the famous “forward/backward algorithms”. For example, in the case of a factorisation $\mathbf{L}\mathbf{U}$ the system 15.1 will be resolved by:

$$\left. \begin{array}{l} \mathbf{K}\mathbf{u} = \mathbf{f} \\ \mathbf{P}\mathbf{K} = \mathbf{L}\mathbf{U} \end{array} \right\} \Rightarrow \begin{array}{ll} \mathbf{L}\mathbf{v} = \mathbf{P}\mathbf{f} & \text{(down)} \\ \mathbf{U}\mathbf{u} = \mathbf{v} & \text{(up)} \end{array} \quad (15.26)$$

In the first lower diagonal (forward) system, we determine the intermediate solution vector \mathbf{v} . The latter then serves as the second member of the upper diagonal system (backward) for which the vector \mathbf{u} we are interested in is a solution.

²⁸By analogy with the polynomial factorisations of the small classes...

This phase is low in cost (for dense, N^2 compared with N^3 for factorisation²⁹ with N the size of the problem) and can thus be repeated many times with the same factorisation. This is very useful when resolving a *multiple second members* problem or when we want to perform *simultaneous resolutions*.

In the *first scenario*, the matrix \mathbf{K} is fixed and we successively change the second member \mathbf{f}_i to calculate as many solutions \mathbf{u}_i (the resolutions are interdependent). This allows for pooling and thus amortising this initial cost of factorisation. This policy, widely used in Code_Aster, could also benefit non-linear algorithms in code_Carmel.

In the *second scenario*, all the \mathbf{f}_i are known at the same time and the forward/backward phases are organised in blocks to simultaneously calculate the independent \mathbf{u}_i solutions. This allows more efficient high-level linear algebra routines to be used, and even to play on memory consumption by storing vectors \mathbf{f}_i and \mathbf{u}_i as sparse.

Remark 15.3.2 *The MUMPS product provides for these two types of strategy and even offers features to facilitate the construction and resolution of the Schur complement. These latter have been implemented for FEM/BEM modelling in Code_Carmel3D.*

We will now look at the process of *factorisation itself*. It is clearly explained in a good number of books³⁰. Hence, we will not deal with it in detail. We will just say that this is an iterative process organised schematically around three loops: one said to be “in i” (on the rows of the working matrix), the second “in j” (the columns respectively) and the third “in k” (the factorisation steps respectively). They iteratively construct a new matrix $\tilde{\mathbf{A}}_{k+1}$ from some of the data from the previous one, $\tilde{\mathbf{A}}_k$, using the conventional factorisation formula that is formally written:

$$\begin{array}{c} \text{Loops with i, j, k} \\ \tilde{\mathbf{A}}_{k+1}(i, j) := \tilde{\mathbf{A}}_k(i, j) - \frac{\tilde{\mathbf{A}}_k(i, k) \tilde{\mathbf{A}}_k(k, j)}{\tilde{\mathbf{A}}_k(k, k)} \end{array} \quad (15.27)$$

Initially the process is activated with $\tilde{\mathbf{A}}_0 = \mathbf{K}$ and at the last step, we recover in the square matrix $\tilde{\mathbf{A}}_N$ the triangular parts (\mathbf{L} and/or \mathbf{U}) or diagonal parts (\mathbf{D}) that interest us. For example, in the case \mathbf{LDL}^T :

$$\begin{array}{c} \text{Loops with i, j, k} \\ \text{if } i < j : \quad \mathbf{L}(i, j) = \tilde{\mathbf{A}}_N(i, j) \\ \text{if } i = j : \quad \mathbf{D}(i, j) = \tilde{\mathbf{A}}_N(i, j) \end{array} \quad (15.28)$$

Remark 15.3.3 *The formula 15.27 contains the problems inherent in direct methods: in sparse storage, the fact that the term $\tilde{\mathbf{A}}_{k+1}(i, j)$ can become non-zero whereas the term $\tilde{\mathbf{A}}_k(i, j)$ is non-zero (concept of fill-in of the factorisation, thus implying a renumbering or “ordering”); the propagation of rounding errors or the division by zero through the term $\tilde{\mathbf{A}}_k(k, k)$ (the concept of pivoting and balancing the terms of the matrix or “scaling”).*

15.3.2 The various approaches

The order of the i, j and k loops is not fixed. We can swap them and perform the same operations but in a different order. This defines six variants kij, kji, ikj, etc. which will manipulate different areas of the current matrix: “zone of new calculated terms” via 15.27, “already calculated and used zone” in TO REVIEW, “already calculated and unused zone” and “not yet calculated zone”. For example, in the jik variant, we have the following method of operation for fixed j:

²⁹For dense, Coppersmith and Winograd (1982) showed that this algorithmic complexity could be reduced at best at CN^α with $\alpha=2.49$ and C constant (for large N).

³⁰G. H. Golub & C. F. VanLoan, Matrix computations. G. W. Stewart, Matrix computations. T. A. Davis, Direct methods for sparse linear systems. G. Meurant, Computer solution of large linear systems. I. Duff, Direct methods for sparse matrix.

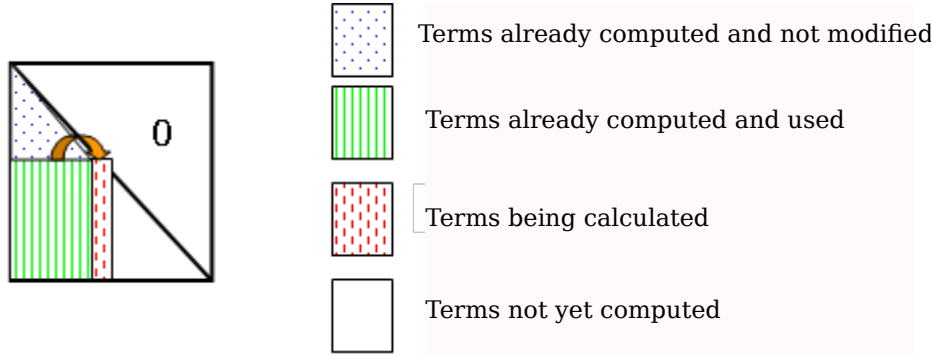


Figure 15.10: Method for constructing a “jik” (“right looking”) factorisation.

Remark 15.3.4 *The method implemented in MUMPS is column orientated (“kji”).*

Remark 15.3.5 *Some variants have special names: Crout (“jki”) and Doolittle (“ikj”) algorithms.*

Remark 15.3.6 *In papers, we often use the English terminology referring to the orientation of matrix manipulations rather than the order of the loops: “looking forward method”, “looking backward method”, “up-looking”, “left-looking”, “right-looking”, “left-right-looking”, etc.*

All these variants are available according to:

- Whether we exploit certain properties of the matrix (symmetry, positive-definite character, band, etc.) or we seek the widest scope of application;
- Whether we perform scalar processing or by blocks;
- Whether the decomposition into blocks is determined by memory aspects (see paginated \mathbf{LDL}^T method in Code `_Aster`) or rather linked to the independence of subsequent tasks (see Aster native multifrontal and MUMPS);
- Whether we re-introduce null terms in the blocks to facilitate access to data³¹ and to generate very efficient algebraic operations, often via BLAS3³² (see native Aster multifrontal and MUMPS);
- Whether we group contributions affecting a block of rows/columns (“fan-in” approach, see PaStiX) or whether they are applied as soon as possible (“fan-out”);
- Whether in parallelism, we seek to manage different levels of sequences of independent tasks, whether they are ordered statically or dynamically, whether we cover the calculation by communication, etc.
- Whether we apply pre- and post-processing to reduce fill-in and improve the quality of results: renumbering of the unknowns, scaling the terms of the matrix, partial pivoting (row) or total (row and column), scalar or diagonal blocks, iterative refinement, etc.

They are often grouped into four categories:

- Classic algorithms: Gauss, Crout, Cholesky, Markowitz (Matlab, Mathematica, Y12M, etc.) ;

³¹This sparse/dense compromise allows a reduction in indirect data addresses and thus to better use the memory hierarchy of current machines.

³²The “ calculation/memory access” ratio of Blas level 3 (matrix product/matrix) is N times better (with N the size of the problem) than other Blas levels. It is also often superior to that of “handmade” routines not optimised on these “data locality/memory hierarchy” aspects.

- Frontal methods (MA62, etc.);
- Multifrontal methods (MULT_FRONT Aster, MUMPS, SPOOLES, TAUCS, UFMPACK, WSMP, etc.);
- Supernodals (SuperLU, PaStiX, CHOLMOD, PARDISO, etc.).

15.3.3 Main steps

When dealing with sparse systems, the numerical factorisation phase (see expression 15.28) does not apply directly to the initial matrix \mathbf{K} , but to a *working matrix* $\mathbf{K}_{\text{travail}}$ resulting from a *pre-processing phase*. This is done in order to *reduce fill-in, improve calculation precision* and thus optimise subsequent CPU and memory costs. Roughly speaking, this working matrix can be written as the following matrix product:

$$\mathbf{K}_{\text{travail}} = \mathbf{P}_0 \mathbf{D}_r \mathbf{K} \mathbf{Q}_c \mathbf{D}_c \mathbf{P}_0^T \quad (15.29)$$

for which we will describe the different elements below.

We can thus break down the operation of a direct solver into four steps:

- *Pre-processing and symbolic factorisation:*³³ it inverts the order of the columns in the working matrix (via a permutation matrix \mathbf{Q}_c) to avoid division by zero of the term $\tilde{\mathbf{A}}_k(k, k)$ and reduce fill-in. In addition, it rebalances the terms to reduce rounding errors (via scaling matrices \mathbf{D}_r and \mathbf{D}_c). *This phase can also be critical for algorithmic efficiency* (sometimes a 10-fold gain) and the quality of the results (a gain of 4 or 5 decimals).

In this phase, we also create the storage structures of the sparse factorisation matrix and the auxiliaries (dynamic pivoting, communication, etc.) required in the following phases. In addition, the task dependency tree is estimated, with initial allocation based on the processors and total projected memory consumption.

- *The renumbering step:*³⁴ it interverts rows in the matrix (via the permutation matrix \mathbf{P}_0) to reduce the fill-in that factorisation implies. Indeed, in the formula 15.27, we see that the factorisation ($\tilde{\mathbf{A}}_{k+1}(i, j) \neq 0$) may contain a new non-zero term in its profile while the initial matrix did not ($\tilde{\mathbf{A}}_k(i, j) = 0$). Due to the term $\frac{\tilde{\mathbf{A}}_k(i, k) \tilde{\mathbf{A}}_k(k, j)}{\tilde{\mathbf{A}}_k(k, k)}$ not necessarily zero.

In particular, it is non-zero when we can find non-zero terms of the initial matrix of type $\tilde{\mathbf{A}}_k(i, l)$ or $\tilde{\mathbf{A}}_k(l, j)$ ($l < i$ and $l < j$). This phenomenon can lead to very large additional memory and calculation costs (the factorisation can be 100 times larger than the initial sparse matrix!).

Hence the idea of renumbering the unknowns (and hence swapping the rows of \mathbf{K}) in order to curb this phenomenon which is the real “*Achilles heel*” of *direct methods*. To do this, we often use external products (METIS, SCOTCH, CHACO, JOSTLE, PARTY, etc.) or heuristics embedded with the solvers (AMD, RCMK, etc.). Of course, these products show different levels of performance depending on the matrices processed, the number of processors, etc.

³³The code_Carmel parameter to control this step is `mumps_pre`. It is useful in both uses of the product: direct solver and preconditioner.

This stage can also provide the user with RAM, disk and flops (Floating-Point Operations per Second) pre-evaluations of MUMPS requirements. These are found in the views that the Carmel user gets when selecting the memory requirement pre-estimation option: `LinearSolverType = 5`.

³⁴The code_Carmel parameter to control this step is `mumps_renum`. With the value “AUTO”, it will choose the most appropriate of the available renumberers (MUMPS incorporates a number of simple renumberers (AMD, AMF, QAMD, PORD) and often “industrial” renumberers (METIS, SCOTCH)). It is useful in both uses of the product: direct solver and preconditioner.

Among them, *METIS*³⁵ and *SCOTCH*³⁶ are very common and “often come out best” (a gain of up to 50%).

- *Numerical factorisation phase*:³⁷ it implements the formula 15.27 through the methods seen in the preceding section. This is by far the most costly phase that will explicitly construct sparse factorisations $\mathbf{L}\mathbf{L}^T$, $\mathbf{L}\mathbf{D}\mathbf{L}^T$ or $\mathbf{L}\mathbf{U}$.
- *Resolution phase*:³⁸ it carries out the forward/backward algorithms 15.26 from which the solution \mathbf{u} “springs” (at last!). It is *low cost* and *possibly pools a later numerical factorisation* (multiple second members, simultaneous resolutions, restarting calculations, etc.).

Remark 15.3.7 Steps 1 and 2 only require knowledge of the initial matrix elimination graph. So, in the end, only data that can be stored and manipulated as integers³⁹. They only need the matrix terms if the scaling steps are engaged. In plugging in MUMPS to code_Carmel (and Code_Aster), we look more for robustness than performance and provide the product with the full matrix terms and not just their graphs.

Remark 15.3.8 Steps 1 and 4 are independent, while steps 2 and 3, in contrast, are linked. Depending on the algorithmic products/approaches, they are grouped differently: 1 and 2 are linked in MUMPS, 2 and 3 in SuperLu and 1, 2 and 3 in UMFPACK. MUMPS allows steps 1+2, 3 and 4 to be carried out separately but successively, and even their results to be pooled to carry out various sequences. For the moment, in code_Carmel, we use mainly the sequences 1+2+3+4 (direct solver), 1+2+3 then 4 several times (direct solver with pooling of the tangent matrix or preconditioner for PCG) and 1 (memory pre-estimate).

Remark 15.3.9 Some products offer to test several strategies in one or more steps and choose the most suitable: SPOOLES and WSMP for step 1, TAUCS for step 3, etc.

Remark 15.3.10 The renumbering tools used in the first phase are based on a wide variety of concepts: engineering methods, geometric or optimisation techniques, graph theory, spectral theory, taboo methods, evolutionary algorithms, memetic algorithms, algorithms based on “colonies of ants”, neural networks, etc. All moves are allowed to improve the local optimum in the form in which the renumberer problem is expressed. These tools are also often used to partition/distribute meshes. In general, METIS is the most effective. But it also competes with its Bordeaux challenger: SCOTCH.

Remark 15.3.11 In addition to the numerical steps described above, there are also steps to manage IT contingencies: initialisation or destruction of the calculation instance, filling it with data from code_Carmel, transfer of the calculated solution to code_Carmel... They are detailed in the section of this document describing the software project.

15.3.4 Main difficulties

Among the difficulties faced by “sparse direct methods” are:

- The manipulation of complex data structures that optimise storage (see matrix profile) but complicate the algorithmics (see pivoting, OOC⁴⁰...). This helps to lower the “calculation/data access” ratio.

³⁵<http://glaros.dtc.umn.edu/gkhome/views/metis/>.

³⁶http://www.labri.fr/perso/pelegrin/scotch/scotch_fr.html.

³⁷In code_Carmel, we are currently limited to the factorisations $\mathbf{L}\mathbf{U}$. The parameters to control this step are `mumps_memory`, `mumps_pivot` and `lmumps_autocorrec`. They are useful in both uses of the product: direct solver and preconditioner.

³⁸The code_Carmel parameters to control this step are `mumps_post` and `kEpsilonMUMPS`. They are most useful in the direct solver scenario. In preconditioner use, it is better to unplug them.

³⁹Only the last two steps really need the actual terms of the matrix.

⁴⁰IC for In-Core (all data structures are in RAM) and OOC for Out-Of-Core (some are dumped to disk).

- The *effective management of data in relation to the memory hierarchy* and the IC/OOC toggle. This is a recurring issue for many problems, but which is pervasive here due to the high consumption of computation.
- The management of the *sparse/dense compromise* (for frontal methods) with respect to memory consumption, ease of access to data, and the efficiency of the linear algebra building blocks.
- The choice of the *right renumbering*: this is an NP-complete problem! For problems of large size, we cannot find the optimal renumbering in a “reasonable” time . We have to settle for a “local” solution. This issue is becoming more pressing with the emergence of parallel renumberers (ParMetis and PT-Scotch).
- The effective management of *rounding error propagation* via scaling, pivoting, and error calculations on the solution (direct/reverse error⁴¹ and conditioning). This point is particularly crucial for Carmel’s singular systems.
- The *factorisation size which is often “bottleneck” $n \circ 1$* . Its distribution between processors (via distributed parallelism) and/or OOC do not always overcome this hurdle (see Figure 15.11). Given the current use of Carmel, which focuses mainly on desktop machines with little RAM, this disadvantage is particularly limiting. It will be partially lifted by the use of parallel clusters.



Figure 15.11: The “scourge” of sparse direct solvers: factorisation size; A factor of 35 between the size of the matrix and that of its factorisation (for the Code_Carmel3D test case TEAM7, on the left) or more than 100 (for the TOLE_CP1_APHI, study, on the right).

15.3.5 The MUMPS product

15.3.5.1 History

MUMPS is a multifrontal “massively” parallel package (“*Multifrontal Massively Parallel sparse direct Solver*”) developed during the European PARASOL Project (1996-1999) by teams from three laboratories: CERFACS, ENSEEIHT-IRIT and RAL (I. S. Duff, P. R. Amestoy, J. Koster and J. Y. L’Excellent). Since this finalised (MUMPS 4.04 22/09/99) and public (free of charge) version, thirty other versions have been delivered (1 or 2 per year). These developments correct anomalies, extend the scope of application, improve ergonomics and, above all, enrich functionality. MUMPS is therefore a long-term, upgradable product⁴² and maintained by teams from IRIT, CERFACS, CNRS and INRIA (half a dozen people).

The product is public and downloadable on its website: <http://graal.ens-lyon.fr/MUMPS>. There are about 1,000 direct users (of which 1/3 Europe + 1/3 USA) without counting those who use it via the libraries that include it (PETSc, TRILINOS, Matlab and Scilab). Its website offers

⁴¹Often referred to as “forward/backward errors”.

⁴²This product, initially only dedicated to one use, has gradually become a true development platform totalling more than 250,000 lines (C and F90).

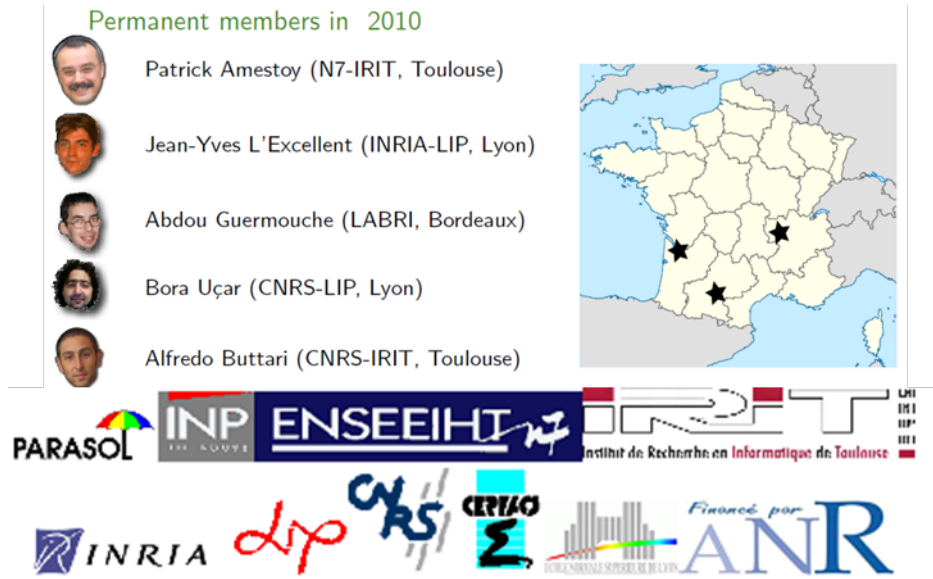


Figure 15.12: Main contributors to MUMPS: organisations, projects, and... researchers.

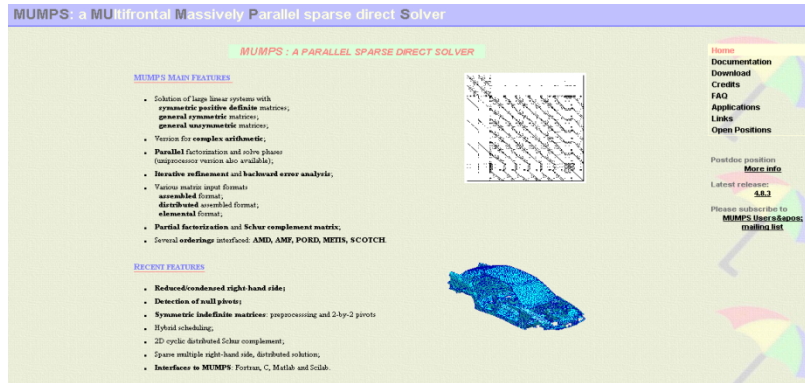


Figure 15.13: Home page of the MUMPS website.

documentation (theoretical and usage), links, examples of applications, as well as a discussion forum (in English) tracing feedback on the product (bugs, installation problems, tips, etc.).

Every year, a dozen or so algorithmic/computational projects lead to improvements in the package (theses, post-docs, research projects, etc.). It is also used regularly for industrial studies (EADS, CEA, BOEING, GeoSciences Azur, SAMTECH, Code_Aster/Telemac...).

EDF R&D has been collaborating on it since 2007⁴³ active and mature (“win-win”) with the MUMPS team. Initiated informally in the framework of ANR SOLSTICE, it was then formalised in the form of an EDF/INPT partnership on low-rank. This collaboration will evolve in 2014 and should take the legal form of a consortium.

The very good progress with this collaboration led us to organise, on the EDF Lab Clamart website, the “MUMPS Users Group Meeting 2013”⁴⁴.

⁴³Exchanges of feedback, reporting of bugs, expertise, specification assistance, independent validation, co-financing of research work, proofreading of documentation, etc. See EDF internal note H-I23-2013-03942. MUMPS Linear Solver: software project in Code_Aster, C. Weisbecker’s PhD thesis on low-rank compressions and EDF/INPT partnership.

⁴⁴http://mumps.enseeiht.fr/ud_2013.php.

15.3.5.2 Main characteristics of MUMPS

MUMPS implements a multifrontal factorisation \mathbf{LU} or \mathbf{LDL}^T (see paragraph 15.3.2). Its main characteristics are:

- *Large scope of application*⁴⁵ SPD, arbitrary symmetric, non-symmetric, real/complex, single/double precision, regular/singular matrix.
- Permits *three data distribution modes*: basic, centralised assembly or distributed assembly⁴⁶.
- *Interfaçage* in Fortran (used), C, Petsc, Matlab/Octave and Scilab.
- *Default configuration*⁴⁷ and possibility to let the package choose some of its options according to the type of problem, its nature and the number of processors.
- *Modularité*⁴⁸ (3 distinct interchangeable phases) and some of the numerical mysteries of MUMPS can be opened up. This allows the (very) advanced user to output the results of certain pre-processing operations (scaling, pivoting, renumbering), modify or replace them with others and reinsert them into the tool-specific string of calculations.
- Different *resolution strategies*: one-shot, multiple second members, simultaneous resolutions and Schur complements⁴⁹.
- Different *renumérateurs* embedded or external: METIS, AMD, QAMD, AMF, PORD, SCOTCH, “user supplied”⁵⁰.
- *Ancillary features*⁵¹: small pivot detection, rank/kernel calculation and regular solution calculation, solution error analysis.
- *Pre- and post-processing*⁵²: scaling, static and dynamic pivoting, row/column permutation and 2x2 scalar/block, iterative refinement.
- *Parallelism*⁵³: potentially at 2 levels (MPI+threads of BLAS3), asynchronous task/data flow management and dynamic reordering, calculation/communication cover; Distribution of data associated with task distribution; This parallelism only starts, for the moment, at the factorisation stage.
- *Mémoire*⁵⁴: dumping or not of the factorisation to disk (In-Core or Out-Of-Core modes) with prior estimation of RAM consumption per processor in both cases; The OOC mode only starts, for the moment, at the factorisation stage.

In terms of parallelism, MUMPS operates on two levels: an external level linked to the concurrent elimination of frontals (via MPI), the other internal, within each frontal (via “threaded” BLAS). It is this type of hybrid parallelism that is relatively flexible, efficient and “push-button” that we want to implement soon in Code_Carmel. In addition, it is sufficiently user friendly to be able to spread upstream and downstream of simple solver aspects (matrix construction, post-processing). This is to the great benefit of users and has relatively little impact for developers.

⁴⁵In real arithmetic, almost this entire scope is now regularly exploited in EDF R&D. codes

⁴⁶The last two modes are used in Code_Aster and so far, only the second one is in Code_Carmel.

⁴⁷Used in Code_Aster and Code_Carmel. This principle is even applied as much as possible to all other types of parameter (the famous “AUTO” mode for AUTOMATIC).

⁴⁸Much used in Code_Aster and Code_Carmel. This is the very heart of the sophisticated integration of MUMPS with these codes. Thanks to these properties, this tool can be used for a wide range of scenarios, and different levels of use can be arranged: standard, robust, high-performance, advanced, expert, etc.

⁴⁹The last three modes are used in Code_Aster, the first two in Code_Carmel.

⁵⁰All can potentially be called in couplings of Code_Aster and Code_Carmel with MUMPS, except the last mode.

⁵¹Used in some features of Code_Aster and often essential for Code_Carmel (non-gauged modelling).

⁵²Used (and often essential for difficult modelling) in Code_Aster and Code_Carmel.

⁵³Used, for now, only in Code_Aster.

⁵⁴Used in all ways possible by Code_Aster/Carmel: user-determined choice, automatic choice, and memory pre-estimates.

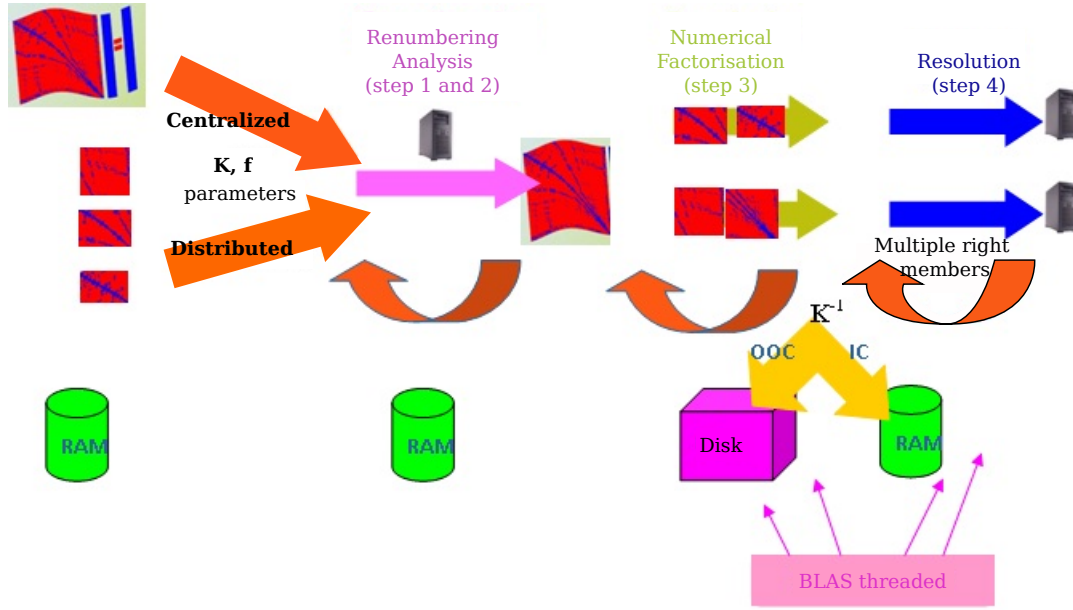


Figure 15.14: MUMPS functional flowchart: its three stages in centralised/distributed parallel and IC/OOC.

15.3.5.3 Advantages and specific features

15.3.5.3.1 Pivoting

The pivoting technique involves choosing a suitable term $\tilde{\mathbf{A}}_k(k, k)$ (in the formula 15.27) to avoid dividing by a term that is too small (which would amplify the spread of rounding errors when calculating the following terms $\tilde{\mathbf{A}}_{k+1}(i, j)$). To do this, you swap rows (partial pivoting) and/or columns (total pivoting) to find the appropriate denominator of 15.27. For example, in the case of partial pivoting, the “pivot” term $\tilde{\mathbf{A}}_k(r, k)$ is chosen such that:

$$\tilde{\mathbf{A}}_k(r, k) > u \max_i |\tilde{\mathbf{A}}_k(i, k)| \quad \text{with } u \in]0, 1[\quad (15.30)$$

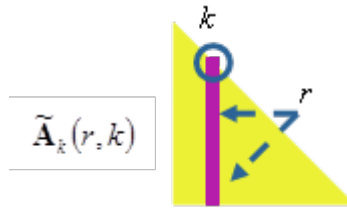


Figure 15.15: Choosing the partial pivot in step k.

This results in an amplification of rounding errors up to $(1 + \frac{1}{u})$ at this step. What is important here is not so much to choose the largest possible term in absolute value ($u=1$) as to avoid choosing the smallest! The reverse of these pivots also occurs during the forward/backward phase, so it is important to avoid these two sources of error amplification by choosing a middle u . MUMPS, like many packages, suggests $u=0.01$ by default (MUMPS parameter CNTL(1)). To pivot, we generally use scalar diagonal terms but also blocks of terms (2x2 diagonal blocks).

In MUMPS, two types of pivoting are implemented, one called “statique” (during the analysis phase), the other called “numérique” (during the numerical factorisation). They can be configured and enabled separately (see MUMPS parameters CNTL(1), CNTL(4) and ICNTL(6)). For SPD

or dominant diagonal matrices, these pivoting powers can be safely disabled (the calculation will gain speed), but in other cases, they must be initialised to manage any very small or zero pivots. This usually involves more fill-in of the factorisation but increases numerical stability.

Remark 15.3.12 *This pivoting feature makes MUMPS essential for handling Carmel’s singular models (and some in Code_Aster).*

Remark 15.3.13 *The Carmel user does not have direct access to this fine configuration. They are enabled with default values. The user can just choose to partially unplug them by setting `mumps_pre = 'OFF'`. By default it is set to `'AUTO'`.*

Remark 15.3.14 *For our industrial simulations (Carmel, Aster or TELEMAT) it is not wise to forgo numerical pivoting or even to enable static pivoting. This type of configuration is best reserved only for testing purposes.*

Remark 15.3.15 *The additional fill-in due to numerical pivoting must be ordered as soon as possible in MUMPS (from the analysis phase). This is done by arbitrarily forecasting a percentage of memory overconsumption compared with the expected profile. This number must be entered as a percentage in MUMPS parameter `ICNTL(14)`. It is accessible to the Carmel user via the key word `mumps_pivot` (20% by default).*

Remark 15.3.16 *The user may need to change this number (up to 100% or more), especially when setting the memory management mode (`mumps_memory = 'IC'` or `'OOC'`). In automatic mode (`mumps_memory = 'AUTO'`), we pre-estimate and provided MUMPS with all available RAM so that it best organises its unannounced memory over-allocations due to pivoting. As a result, this type of problem (“not enough additional space for pivoting”) occurs much less often. And when it appears, the AUTO mode increases the value, transparently for the user, and retries the numerical factorisation. In case of failure, we try again several times⁵⁵, doubling the value each time. This self-correction procedure is enabled by default (parameter `Lmumps_autocorrec = .true.`) and lets the user intervene as little as possible in management of these “computer-numerical” contingencies.*

15.3.5.3.2 Iterative refinement

At the end of resolution, having obtained the solution \mathbf{u} of the problem, we can easily evaluate its residual $\mathbf{r} := \mathbf{K} \mathbf{u} - \mathbf{f}$. Knowing the factorisation of the matrix already, this residual can then be input, at low cost, into the following iterative enhancement process (in the general non-symmetric case):

$$\begin{array}{ll}
 \text{Loop with } i & \\
 (1) & \mathbf{r}^i = \mathbf{f} - \mathbf{K} \mathbf{u}^i \\
 (2) & \mathbf{L} \mathbf{U} \delta \mathbf{u}^i = \mathbf{r}^i \\
 (3) & \mathbf{u}^{i+1} \leftarrow \mathbf{u}^i + \delta \mathbf{u}^i
 \end{array} \tag{15.31}$$

This process is “relatively”⁵⁶ painless since it costs mainly the price of the forward/backward step (2). It can thus iterate up to a certain threshold or a maximum number of iterations. If the residual calculation does not contain too many rounding errors, i.e. if the resolution algorithm is quite reliable (see next paragraph) and the conditioning of the matrix system is good, this iterative refinement process⁵⁷ is very beneficial to the quality of the solution.

In MUMPS this process is enabled or not (parameter `ICNTL(10)`) < 0) and bounded by a maximum number of iterations N_{err} (`ICNTL(10)`). The 15.31 process continues as long as the “balanced residual” \mathbf{B}_{err} is above a configurable threshold (`CNTL(2)`, set by default to $\sqrt{\varepsilon}$ where ε is the machine accuracy):

⁵⁵Up to 4 times.

⁵⁶In OOC mode this feature can be very costly depending on the speed of disk access. We often prefer to unplug it unless we are absolutely looking to give priority to the quality of the result.

⁵⁷We also call it “iterative enhancement” (‘iterative refinement’).

$$\mathbf{B}_{err} := \max_j \frac{|\mathbf{r}_j^i|}{(|\mathbf{K}||\mathbf{u}^i| + |\mathbf{f}|)_j} \quad (15.32)$$

or it does not decrease by a factor of at least 5 (not configurable). Usually one or two iterations are enough. If not, it is often indicative of other problems: poor conditioning or reverse error (see next paragraph).

Remark 15.3.17 *For the user of code_Carmel, these MUMPS parameters are not directly accessible. The functionality is only enabled if the user knowingly chooses to estimate and test the quality of the solution (see next paragraph). For example, via the parameter `post_mumps` = 'AUTO' 'FORCE'. In these pre-configured scenarios, this value is initialised either to 'OFF' ($N_{err}=0$ / $\text{threshold}=10^{+50}$), 'FORCE' ($N_{err}=10$ / $\text{threshold}=10^{-50}$) or 'AUTO' ($N_{err}=4$ / $\text{threshold}=10^{-14}$).*

Remark 15.3.18 *The number of iterations actually performed is plotted in the MUMPS display block.*

Remark 15.3.19 *This feature is present in many packages: Oblio, PARDISO, UFMPACK, WSMP, PaStiX, etc.*

15.3.5.3.3 Reliability of the calculations

To estimate the quality of a linear system's solution, MUMPS offers numerical tools derived from the theory of *reverse analysis of rounding errors initiated by Wilkinson* (1960). In this theory, rounding errors due to several factors (truncation, finite arithmetic operation, etc.) are treated as disruptions to the initial data.

This makes it possible to compare them with other sources of error (measurement, discretisation, etc.) and to manipulate them more easily via three indicators obtained in post-processing:

- *The conditioning $\text{cond}(\mathbf{K}, \mathbf{f})$* : it measures the *sensitivity of the problem to data* (unstable problem, poorly formulated/discretised, etc.). In other words, the multiplication factor that the manipulation of the data will apply to the result. To improve it, we can try to change the formulation of the problem or balance the terms of the matrix, outside of MUMPS or via MUMPS (`mumps_pre` = 'AUTO').
- *The reverse error $\text{be}(\mathbf{K}, \mathbf{f})$* ("backward error"): it measures the *propensity of the resolution algorithm to pass on/amplify rounding errors*. A tool is said to be "reliable" when this number is close to the machine accuracy. To improve it, we can try to change the resolution algorithm or modify one or more of its steps (in Code_Carmel we can adjust the parameters `mumps_post` or `mumps_renum`).
- *The direct error $\text{fe}(\mathbf{K}, \mathbf{f})$* ("forward error"): it is the product of the previous two digits and provides an *upper bound for the relative error on the solution*.

$$\frac{\|\delta \mathbf{u}\|}{\|\mathbf{u}\|} < \underbrace{\text{cond}(\mathbf{K}, \mathbf{f}) \times \text{be}(\mathbf{K}, \mathbf{f})}_{\text{fe}(\mathbf{K}, \mathbf{f})} \quad (15.33)$$

We can give a graphical representation (see Figure 15.16) of these concepts by expressing the backward error as the difference between the initial "exact" data (\mathbf{f}) and that "actually manipulated" ($\mathbf{f} + \delta \mathbf{f}$), while the forward error measures the difference between the "exact" solution (\mathbf{u}) and the solution actually obtained ($\mathbf{u} + \delta \mathbf{u}$), that of the problem disrupted by the rounding errors.

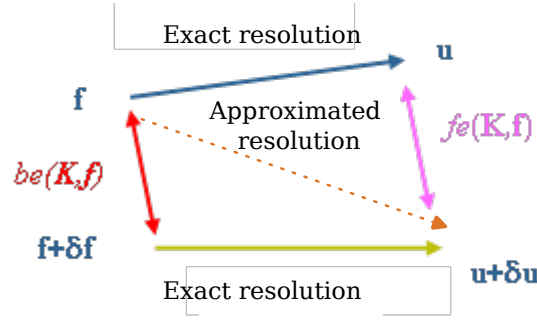


Figure 15.16: Graphical representation of the concept of forward and backward error.

For linear systems, the backward error is measured *via the balanced residual*:

$$\text{be}(\mathbf{K}, \mathbf{f}) := \max_{j \in J} \frac{|\mathbf{f} - \mathbf{K}\mathbf{u}|_j}{(|\mathbf{K}||\mathbf{u}| + |\mathbf{f}|)_j} \quad (15.34)$$

It cannot always be evaluated on all indices ($J \neq [1, N]_N$). Especially when the denominator is very small (and the numerator is non-zero), we prefer the formulation (with J^* such that $J \cup J^* = [1, N]_N$):

$$\text{be}^*(\mathbf{K}, \mathbf{f}) := \max_{j \in J^*} \frac{|\mathbf{f} - \mathbf{K}\mathbf{u}|_j}{(|\mathbf{K}||\mathbf{u}|)_j + \|\mathbf{K}_j\|_\infty \|\mathbf{u}\|_\infty} \quad (15.35)$$

where \mathbf{K}_j is the j -th row of matrix \mathbf{K} . With these two indicators we associate two estimates of the matrix conditioning (one linked to the rows selected in the set J and the other in its complement J^*): $\text{cond}(\mathbf{K}, \mathbf{f})$ and $\text{cond}^*(\mathbf{K}, \mathbf{f})$.

The theory then gives us the following results:

- The approximate solution \mathbf{u} is the exact solution to the disrupted problem:

$$\begin{aligned} (\mathbf{K} + \delta\mathbf{K})\mathbf{u} &= (\mathbf{f} + \delta\mathbf{f}) \\ \text{avec } \delta\mathbf{K}_{ij} &\leq \max(\text{be}, \text{be}^*) |\mathbf{K}_{ij}| \\ \text{et } \delta\mathbf{f}_i &\leq \max(\text{be} \cdot \mathbf{f}_i, \text{be}^* \cdot \|\mathbf{K}_i\|_\infty \|\mathbf{u}\|_\infty) \end{aligned} \quad (15.36)$$

- We have the following increase (via the forward error $\text{fe}(\mathbf{K}, \mathbf{f})$) on the relative error in solution:

$$\frac{\|\delta\mathbf{u}\|}{\|\mathbf{u}\|} < \underbrace{\text{cond} \times \text{be} + \text{cond}^* \times \text{be}^*}_{\text{fe}(\mathbf{K}, \mathbf{f})} \quad (15.37)$$

In practice, the latter estimate (\mathbf{K}, \mathbf{f}) and its components are scrutinised. Its order of magnitude indicates roughly the number of “true” decimals of the calculated solution. For Carmel’s “very” singular problems, a tolerance of 10^{-3} is not uncommon.

Remark 15.3.20 For the user of Code_Carmel these MUMPS parameters are calculated and displayed in the display block “MONITORING OF OVERALL MUMPS RESOLUTION...” as soon as this information is requested (parameter `Imonitoring_systeme` > 1 in solver, > 2 in preconditioner). Because its cost may not be negligible: between 10% and 30% of the total cost (especially in OOC).

Remark 15.3.21 Enabling this feature is not necessarily necessary when the solution sought is itself corrected by another algorithmic process (Newton’s method, etc.). In short, in non-linear, we can often do without it (`mumps_post` = ‘OFF’). Especially if we have already made the approximation to pool the linear solver aspects (factorisation or preconditioner) between several iterations of the non-linear solver (`reacprecond_methodeNL` > 0).

Remark 15.3.22 If the `Code_Carmel` user has enabled this automatic quality check (`mumps_post` other than 'OFF') and it exceeds the criterion set by `kEpsilonMUMPS` > 0 (10^{-6} by default), the calculation stops with a fatal error. If `kEpsilonMUMPS` < 0 , we estimate the quality of the solution (and possibly display it) but we do not test it and we do not stop. Warnings may appear if the values of these parameters appear suspicious (for example, `kEpsilonMUMPS` > 0 and `mumps_post` = 'OFF').

Remark 15.3.23 This type of functionality appears to be rarely found in libraries: LAPACK, Nag, HSL, etc.

15.3.5.3.4 Memory management

We have seen that the major disadvantage of direct methods is the size of the factorisation. To allow larger systems to be moved into RAM, MUMPS offers to dump this object to disk: this is the Out-Of-Core (OOC) mode as opposed to the In-Core (IC) mode where all data structures reside in RAM. This method of saving RAM is complementary to the distribution of data that parallelism naturally induces. The added value of OOC is therefore particularly significant for moderate numbers of processors (< 32 processors).

On the other hand, the MUMPS team has been very attentive to the CPU overhead generated by this practice. By reworking the algorithmics of the code, and the manipulation of the dumped entities, they have been able to keep this overhead (a few percent and above all in the resolution phase) to a minimum.

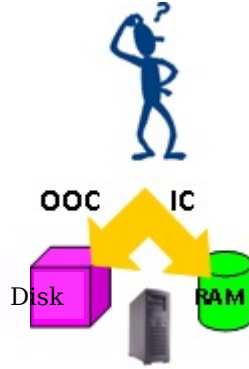


Figure 15.17: Two types of memory management: all in RAM (IC) and RAM/disk (OOC).

Remark 15.3.24 The MUMPS parameters `ICNTL(22)/ICNTL(23)` allow configuration of different memory management modes. The Carmel user only has direct access to it via the key word `mumps_memory` ('IC'/'OOC'/'AUTO'). For the value 'AUTO', the IC and OOC values are chosen dynamically according to the memory available on the current node. In case of a problem, if `Lmumps_autocorrec` = .true., we try to correct it dynamically (see section 15.3.5.3.1).

Remark 15.3.25 When automatic mode is required, if a computer problem prevents the evaluation of available RAM, we revert to the conservative choice of OOC.

Remark 15.3.26 The disk dump is fully controlled by MUMPS (number of files, dump/reload frequency, etc.). We just have to enter the memory location: this is the naturally the working directory of the executable for each processor (defined by `%OOC_TMPDIR` = '.'). These files are automatically deleted by MUMPS when the associated instance is deleted. This avoids disk overload when different systems are factorised in the same resolution.

Remark 15.3.27 Other OOC strategies would be possible or are already coded for certain packages (PaStiX, Oblío, TAUUS, etc.). We are thinking in particular of the ability to modulate the scope of dumped objects or even to reuse them on disk during another run. This last strategy would prove very valuable for certain uses of Carmel.

15.3.5.3.5 Management of singular matrices

One of the strong points of the product is its *management of singularities*. It is not only capable of *detecting numerical singularities*⁵⁸ of a matrix and summarising the information for an external use (rank calculation, warning to the user, display of expertise, etc.), but in addition, despite this difficulty, it calculates a “regular” solution⁵⁹ or even all or part of the *associated kernel*.

These new developments were one of the deliverables of ANR SOLSTICE. We had requested them from the MUMPS team (in partnership with the Algo team of CERFACS) to make this product iso-functional with respect to the other direct solvers of Code_Aster.

This feature finds a second field of application with the *potentially singular numerical modelling in Code_Carmel*. Apart from iterative solutions already integrated into the code, MUMPS is probably one of the few products equipped to resolve this type of difficulty. The tests conducted during this software project thus complete the evaluation undertaken during ANR and feedback on use in Code_Aster. On the other hand, here again, the *needs of Carmel/Aster are complementary*:

- For Code_Carmel it is a question of finding a possible solution to the problem.
- For Code_Aster, this situation is often considered pathological. In this case, we want to warn the user of a problem with the data (boundary condition, contact, etc.) or send a signal to the algorithm (time step refinement, etc.).

And in practice, how does MUMPS do it?

In broad outline, when constructing the factorisation matrix, it detects rows with pivots⁶⁰ that are very small (compared with a criterion $\text{CNTL}(3)$ ⁶¹). It lists them in the vector $\text{PIVNUL_LIST}(1:\text{INFOG}(28))$ and, as appropriate, either replaces them with a pre-set value (via $\text{CNTL}(5)$ ⁶²), or it stores them separately. The resulting (smaller) block will subsequently undergo an ad hoc QR algorithm.

And finally, iterations of iterative refinements complete this labyrinth. Since they use this “retouched” factorisation only as a preconditioner, and they benefit from the exact information of the matrix-vector product, they provide the “biased” solution⁶³ back on the right track!

Remark 15.3.28 *The MUMPS parameters $\text{ICNTL}(13)$ / $\text{ICNTL}(24)$ / $\text{ICNTL}(25)$ and $\text{CNTL}(3)$ / $\text{CNTL}(5)$ allow configuration of these features. They cannot be changed by a standard use of Code_Carmel. Out of caution, the functionality is always enabled.*

Remark 15.3.29 *This functionality can also be valuable in domain decomposition (FETI linear solver, preconditioner) and modal calculation (rigid mode filtering).*

15.3.6 Implementing MUMPS in code_Carmel

15.3.6.1 Version compatibility and copyright

The copyright of the MUMPS product (reproduced in section K) must be attached to the theoretical documentation and/or the Code_Carmel. user manual. It reminds the user of the authorship of the product and the conditions of its use.

⁵⁸The MUMPS parameters $\text{ICNTL}(13)$ / $\text{ICNTL}(24)$ / $\text{ICNTL}(25)$ and $\text{CNTL}(3)$ / $\text{CNTL}(5)$ allow these features to be configured. They are not modifiable in standard use of Code_Carmel. Out of caution, the feature is kept permanently enabled.

⁵⁹This is a possible solution of the problem at the time the second member $\mathbf{f} \in \ker(\mathbf{K}^T)^T$. Which in our symmetric case is \mathbf{f} element of the image space.

⁶⁰Strictly, this is the infinite norm of the row of the working matrix with the pivot.

⁶¹By default it is set to 10^{-8} (in double precision) and 10^{-4} (in single) because these numbers represent (empirically) a loss of at least half the level of precision if the factorisation is continued.

⁶²This value must be large enough to limit the impact of this change on the rest of the factorisation. In Carmel, it is set to $10^6 \|\mathbf{K}_{\text{travail}}\|$.

⁶³This is the same mechanism as for static pivoting.

The main features of MUMPS have just been described in the preceding sections. Their links with configuration specific to Code_Carmel and comparisons with other EDF R&D uses (Code_Aster, TELEMAC, etc.) or comparable tools (PaStiX, Pardiso, etc.) were also mentioned.

These parameters, which can be explicitly changed in the configuration file.F90⁶⁴, are only permissible if Code_Carmel has been linked to MUMPS beforehand (see makefile and USE_MUMPSvariable).

The *compatible versions of MUMPS* are *v4.9.2* and *v4.10.0*. There is no need to go too far in backward compatibility. The oldest version of MUMPS for which we ensure compatibility already dates back to November 2009!

And in any case, it is only from this version that MUMPS has stabilised features crucial for the needs of Code_Carmel: management of singular systems, Out-Of-Core memory management and pre-allocation of memory requirements.

This compatibility is tested every time a linear system is initialised (see InitializeOccMUMPS routine) but is only displayed in the .log the first time⁶⁵. In case of version incompatibility, a dedicated message is sent.

15.3.6.2 Ergonomic choices

In fact, *MUMPS is more of a tool kit than a product dedicated to a single task*. Its entry points are numerous (about fifty parameters) and its interactions multiple (combined parameter sets, a hundred messages and warnings). In short, the range of possible choices remains considerable. It is therefore preferable to relieve the user of Code_Carmel and to group these parameters in order to provide a *minimal API*.

As a result, we have simplified and grouped these MUMPS settings into just eight Code_Carmel parameters. This choice to group this rich configuration into sub-categories (often pre-configured and dynamically enabled), allows *users to gradually access this functionality*.

Depending on their expertise, needs and appetite, they can choose the mode of operation that best suits them: *from “push-button” use through to very advanced use*.

The priority when using MUMPS in Code_Carmel being:

- “Maximum scope of application” and “Service continuity”. Thus not restricting the uses of the product while trying to self-correct as many internal MUMPS problems as possible.
- Robustness of the numerical process.
- Precision of the results.
- Time and memory consumption.

When using MUMPS as a preconditioner, the last two objectives are swapped. Priority is given to completing the preconditioning step as quickly as possible. It doesn't matter how precise it is, since we are already working on an approximate system (single precision and relaxed) and it is inserted into an iterative and corrective process (PCG algorithm).

The internal MUMPS parameters can thus be broken down into four categories:

- Those *prefixed “hard”* in Code_Carmel⁶⁶ because we know in principle the characteristics of the calculation and the priority of users when they use MUMPS (system type, handling singularities, pre and post-processing, etc.): a robust and safe calculation.

ICNTL: 1, 2, 3, 4, 13, 24 etc.
CNTL: 3, 5 etc.

- Those that can *evolve dynamically* according to the calculation and resources of the machine (memory management, additional space dedicated to pivoting, etc.).

ICNTL: 14, 22, 23.

⁶⁴As usual in Code_Carmel.

⁶⁵So as not to overload the .log.

⁶⁶In the InitializeOccMUMPS routine.

- Those that are *driven* (often collectively) *by the user*. They have a default value or, at the very least, an “AUTO” mode which lets the product or the Code_Carmel- MUMPS integration decide according to the situation, the resources and the peripheral products installed⁶⁷.

ICNTL: 7, 6, 8, 10, 11, 12, 14, 22, 23 etc.
CNTL: 2.

In addition, when possible, we implement *self-correcting procedures* (if `Lmumps_autocorrec = .true.`) to dynamically change this MUMPS configuration when a numerical or computer problem occurs in the background of the external product. The priority is to ensure the continuity of the calculation so as, for example, not to “crash” at the 500th linear system resolution just because of not enough pivoting space!

Given the embryonic functionality of Code_Carmel in terms of intermediate restarts and backup of computation results, it is essential to provide the user with this *guarantee of “service continuity”*.

Remark 15.3.30 *Pre-estimation, self-configuration, self-correction... we could eventually extend this type of operation to other numerical tools of the code: ODE solver and non-linear solver, etc.*

15.3.6.3 Code_Carmel parameters to use MUMPS

These Carmel parameters are grouped in the overview below (see Figure 5.4.1). The allowable, recommended, and default values are summarised in the comments in the configuration file.



Figure 15.18: Parameters for using MUMPS in Code_Carmel.

In addition, the new `Imonitoring_systeme` parameter allows tracking of different information about this linear system resolution step in the `.log`.

The new option `LinearSolverType = 5` allows pre-evaluation of memory requirements (RAM and possibly disk) of Code_Carmel based on the user parameters (in `configuration.F90` and `structureDonnées.F90`), the characteristics of the problem processed and the available linear solvers (PCG Crout/Jacobi with or without MUMPS direct solver and preconditioner).

Users can thus, initially and for a given study, calibrate their set of parameters with respect to the available memory resources of their machine. They then start the calculation with the most suitable configuration.

This mode of operation can even be used, quite simply, to *pre-test their dataset and their installation*. If Code_Carmel manages to pre-evaluate all memory consumption, this is very good sign! The code is most likely functional on this platform and the dataset probably legal! Because the data was successfully read, the matrix was constructed and, if applicable (if MUMPS is installed), the matrix was analysed by MUMPS.

This is a (light) deployment of this feature which can be very useful in practise: It is often less costly than a portion of the study.

⁶⁷METIS or SCOTCH renumberers.

15.3.6.4 MUMPS warnings and error reporting

The MUMPS product is likely to send its user a hundred error messages and warnings. Of course, there is no question of providing a “dry” MUMPS return code to the Carmel user, nor of “customising”⁶⁸ each of the possibilities in detail.

We have thus chosen a middle road:

- *Auto-corriger*, as many parameters as possible (see previous section);
- *Group these messages by category* so as to send the final Carmel user a message that is *simple, clear and usable*. Generally, either some advice is given to restart the calculation, or it is suggested contacting the development team: the implication is that this study may have highlighted a bug that needs to be investigated⁶⁹.

15.4 Organisation of calculations with MUMPS

For the implementation of calculations with MUMPS, the priority has been on modularity, readability and QA. Ten new routines have been created to this end. The main features of this integration are described below.

15.4.1 Initialisation

When initialising the Carmel data structure⁷⁰ containing data specific to the linear system (matrix, RHS⁷¹...), we also initialise the associated MUMPS instance⁷² (matrix, RHS...). This ensures dialogue with the numerical features of the product. It contains the control parameters for its numerical features: 15 real and 44 integer.

This initialisation takes place by a call on

Call InitializeOccMUMPS(systeme)

 (15.38)

Once this instance has been finished with, it is supposed to be destroyed (via CleanOccMUMPS) to gain memory space (RAM and disk) and avoid possible confusion. For the sake of simplicity, there must be a bijection between a Carmel “system” and a MUMPS instance. The latter cannot be enlarged, for example, once it has been created. If the Carmel system changes, the MUMPS instance must be destroyed and recreated.

Remark 15.4.1 *If there is already a MUMPS instance associated with the system, the routine stops with an error. It must be first be destroyed via a CleanOccMUMPS.*

Remark 15.4.2 *This routine should be called before filling the matrix and the MUMPS RHS.*

Remark 15.4.3 *Many warnings may be issued when performing this routine: checks on configuration, versions, installation, etc.*

Remark 15.4.4 *This initialisation occurs only if Code_Carmel has been linked to the external product and the configured functionality requires it (`LinearSolverType` = 3, 4, or 5).*

Remark 15.4.5 *Unlike Code_Aster, , there is no limit on the number of simultaneous instances that can exist in memory. This is not an issue (for example, not enough memory) at this time, because only one linear system appears to be used in Code_Carmel.*

⁶⁸Unnecessary and quickly unmanageable when you want to be compatible with several versions.

⁶⁹In MUMPS (or its dependencies), in the Code_Carmel-MUMPS integration or in Code_Carmel.

⁷⁰A data structure called system. It is initialised in initialiserSysteme.

⁷¹Right-Hand-Side or second member.

⁷²This instantiates a MUMPS derived type smumps_struc or dmumps_struc, depending on whether we are in single or double precision.

15.4.2 Filling

Then we read the *Carmel matrix* and analyse its characteristics. Next we initialise the associated *MUMPS instance* and fill it with the appropriate matrix terms. Any outliers are filtered out⁷³ along with extra-diagonal terms that are too small (if constructing a relaxed preconditioner).

CallInitFillMatrixMUMPS(systeme) (15.39)

Remark 15.4.6 *If there is already a MUMPS matrix associated with this system, the routine stops with an error. It must be first be destroyed via a CleanMatrixMUMPS.*

Remark 15.4.7 *During filtering, if the matrix has illicit values, depending on the case, the calculation is stopped with a fatal error.*

Remark 15.4.8 *The results of the filtering (Carmel / MUMPS profile size, number of “outrange” terms and number of relaxed terms) is plotted in the .log file if Imonitoring_systeme > 1 (direct solver) or 2 (preconditioner).*

We do the same with the second member via

Call InitFillRhsMUMPS(systeme,B) (15.40)

Remark 15.4.9 *If there is already a MUMPS RHS associated with the system, the routine will stop with an error. It must be first be destroyed via a CleanRhsMUMPS.*

Remark 15.4.10 *Filling of the second member of the MUMPS instance can be deferred and done just before the forward/backward step (DoSolveMUMPS).*

Remark 15.4.11 *This modularity is very convenient when handling a preconditioner or implementing a Newton method: we can easily and effectively pool the same matrix for many RHSs.*

15.4.3 Calculation steps

The three calculation steps themselves (described in section 15.3.3) are performed through the following calls:

- Pre-processing, symbolic factorisation and renumbering

Call DoAnalyseMUMPS(systeme,ramIC,ramOOC,diskOOC,objetMUMPS) (15.41)

- Numerical factorisation

Call DoFactorizationMUMPS(systeme,ramIC,ramOOC) (15.42)

- Forward/backward

Call DoSolveMUMPS(System) (15.43)

Then of course the MUMPS solution must be provided to the “Carmel world”. This is done using the following utility:

Call GiveSolutionMUMPS(system,X) (15.44)

that fills the Code_Carmel X vector with the much sought-after solution!

⁷³Depending on the arithmetic considered (single or double precision) and the desired function (direct solver or single preconditioner).

Remark 15.4.12 *To use these routines, the MUMPS instance must of course have been created and filled.*

Remark 15.4.13 *In theory, the analysis phase can be pooled for several factorisations. At least so long as only the values of the matrix terms are changed. However, in order to preserve the effectiveness of the pre-processing and the relevance of the “integration tricks” (pre-allocation of memory, self-correction, etc.), it is better to repeat this step before each numerical factorisation. The gain (in time and RAM) from pooling is often small in sequential mode. This will be much less true in MPI parallel mode.*

Remark 15.4.14 *We can insert between 15.42 and 15.43 an RHS filling step (see 15.40). It is not required before that. This results in “multiple-RHS” calculation methods for which we can pool the numerical factorisation for many different RHSs.*

Remark 15.4.15 *Depending on the level of monitoring (via `Imonitoring_systeme`) and the intended use (direct solver or preconditioner), each of these steps plots different elements: estimates of memory requirements, time consumed, self-correction procedure enabled, numerical expertise, system characteristics, etc.*

Remark 15.4.16 *The very large MUMPS objects containing the factorisation⁷⁴ are not destroyed until the instance is destroyed (`CleanOccMUMPS`) or a factorisation is re-attempted (`DoFactorisationMUMPS`). As they are likely to be reused on new RHSs.*

15.4.4 Cleaning

Destroying the objects and the MUMPS instance is done through the following calls:

- MUMPS matrix

Call `CleanMatrixMUMPS(systeme)` (15.45)

- MUMPS RHS

Call `CleanRhsMUMPS(systeme)` (15.46)

- MUMPS instance

Call `CleanOccMUMPS(systeme)` (15.47)

Remark 15.4.17 *To use these routines, the MUMPS instance must of course have been created and filled.*

Remark 15.4.18 *Destroying the MUMPS instance must be the last step in a calculation method using this external product.*

⁷⁴In RAM and on disk (if OOC has been enabled).

Part V

Post-processing

Chapter 16

Force calculations

Abstract

Magnetic forces exerted on the moving parts of an electromagnetic system constitute important values for the study of its operation. This is the case for pulsating torque in rotating machines or forces that act on the surface of a magnetic material.

As a whole, these values can also be used for coupling with mechanical equations (calculation of speed and position) [Vassent 1990] [Ren, Razek 1994]. At the local level, the results are used to predict possible system deformations [Ren, Razek 1992], [Ren et al 1992], [Henneberger, Hadrys 1993].

To determine these forces, several calculation methods may be used. These methods include those based on the calculation of force density (distribution of local force at the surface). Overall force is then obtained through a sum of local forces. These methods use concepts based on equivalent sources (loads or magnetic currents), the derivative of the magnetic energy or the Maxwell stress tensor [Coulomb 1983], [Ren, Razek 1992], [Sadowski et al 1992], [Ren 1994]. In a finite element calculation, these techniques use the distribution of local values in the domain under study.

Here, we will focus on the overall calculation of force and torque by two methods: Maxwell stress tensor and virtual work. These calculations are presented for 2D and 3D structures with a discretisation, by the finite element method, of the electromagnetic field equations [Boualem, Piriou 1996].

16.1 Maxwell stress tensor method

16.1.1 Principle

We want to calculate the forces and torque that act on a part \mathcal{D}' of the domain under study \mathcal{D} . This region is bounded by a surface Γ . The system may contain magnetic materials (linear or non-linear) conductors or inductor regions with uniform current density (see Figure 16.1).

This calculation is performed by applying the Maxwell stress tensor \mathbf{T} that is defined in a vacuum (or equivalent medium) by the following expression [Durand 1968]:

$$\mathbf{T}_{i,j} = \mu_0 \left(h_i h_j - \frac{1}{2} \delta_{ij} h^2 \right) \quad (16.1)$$

The Maxwell stress tensor allows calculation of the force acting on domain \mathcal{D}' by an integral extended to a surface Γ' surrounding it:

$$\mathbf{F} = \int_{\mathcal{D}'} \text{div} \mathbf{T} d\mathcal{D}' = \oint_{\Gamma'} \mu_0 \left((\mathbf{h} \cdot \mathbf{n}) \mathbf{h} - \frac{1}{2} |\mathbf{h}|^2 \mathbf{n} \right) d\Gamma' \quad (16.2)$$

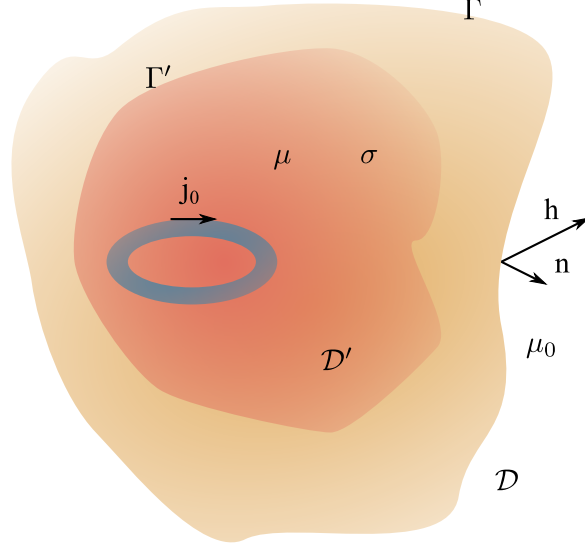


Figure 16.1: Application of the Maxwell stress tensor.

The surface Γ' can be arbitrary provided it is completely defined in a vacuum (or equivalent medium). It must also cover the whole region of interest. \mathbf{n} represents the outgoing normal to this surface. From the previous expression, we can also deduce the value of the torque C :

$$C = \mathbf{r} \times \mathbf{F} = \oint_{\Gamma'} \mu_0 \left((\mathbf{h} \cdot \mathbf{n}) (\mathbf{r} \times \mathbf{h}) - \frac{1}{2} |\mathbf{h}|^2 (\mathbf{r} \times \mathbf{n}) \right) d\Gamma' \quad (16.3)$$

\mathbf{r} is the vector that connects the integration element “ $d\Gamma'$ ” to the axis of rotation.

16.1.2 Discretisation

For the computational electromagnetics of electrotechnical systems, we are required to approach the integrals giving the values of the force (16.2) or torque (16.3) by a finite sum effected on surface Γ' . As a result, the latter is represented by an assembly of “ N_e ” surface elements in 3D or linear elements in 2D. Such entities are obtained by the intersection of the finite element mesh with surface Γ' . As a result, only one part of the mesh elements is affected by the force or torque calculation. For the magnetic field \mathbf{H} , it is replaced by its numerical approximation on each element.

We seek to determine the components of force $\mathbf{F}^T (F_x, F_y, F_z)$ and the torque. The torque is calculated for a rotation of domain \mathcal{D}' around the Oz axis. It thus has only one component, denoted C_z . The use of first-order tetrahedral elements requires a linear variation of the potential considered. The magnetic field is then constant in each element for tetrahedra. For a surface element Γ'^e , the field and outgoing normal components are, respectively, $h^{eT} (h_x^e, h_y^e, h_z^e)$ and $n^{eT} (n_x^e, n_y^e, n_z^e)$. It is noted that the choice of a flat elemental surface implies a single outgoing normal. Using the previous notations in relation (16.2), we can then write the components of the force \mathbf{F} in the following matrix form (see annex M):

$$F_s = \frac{\mu_0}{2} \sum_{e=1}^{N_e} \Gamma'^e \mathbf{H}^{eT} \mathbf{M}_s \mathbf{H}^e \quad s = x, y, z \quad (16.4)$$

For each component of force \mathbf{F} , matrix \mathbf{M}_s^e is written:

$$\mathbf{M}_x^e = \begin{bmatrix} n_x^e & 0 & 0 \\ 2n_y^e & -n_x^e & 0 \\ 2n_z^e & 0 & -n_x^e \end{bmatrix} \quad \mathbf{M}_y^e = \begin{bmatrix} -n_y^e & 2n_x^e & 0 \\ 0 & n_y^e & 0 \\ 0 & 2n_z^e & -n_y^e \end{bmatrix} \quad \mathbf{M}_z^e = \begin{bmatrix} -n_z^e & 0 & 2n_x^e \\ 0 & -n_z^e & 2n_z^e \\ 0 & 0 & n_z^e \end{bmatrix} \quad (16.5)$$

For the calculation of torque, the vector \mathbf{r} is defined by the projections of the barycentre of the surface element on the Oxy plane, i.e. $\mathbf{r}^{eT} = (r_x^e, r_y^e, 0)$. From relation 16.3, the component C_z of the torque can be written:

$$C_z = \frac{\mu_0}{2} \sum_{e=1}^{N_e} \Gamma'^e (\mathbf{h}^{eT} \mathbf{M}_c^e \mathbf{h}^e) \quad (16.6)$$

Matrix \mathbf{M}_c^e is given by the following expression:

$$\mathbf{M}_c^e = r_x^e \mathbf{M}_y^e - r_y^e \mathbf{M}_x^e = \begin{bmatrix} -r_x^e n_y^e - r_y^e n_x^e & 2r_x^e n_x^e & 0 \\ -2r_y^e n_y^e & r_x^e n_y^e + r_y^e n_x^e & 0 \\ -2r_y^e n_z^e & 2r_x^e n_z^e & -r_x^e n_y^e + r_y^e n_x^e \end{bmatrix} \quad (16.7)$$

In the case of rotating machines, the torque calculation is performed using a surface placed in the air gap. This is a cylinder of axis Oz and radius R. Using the cylindrical coordinates in the previous expression, we obtain:

$$\mathbf{M}_c^e = 2R \begin{bmatrix} -\cos \theta^e \sin \theta^e & \cos^2 \theta^e & 0 \\ -\sin^2 \theta^e & \cos \theta^e \sin \theta^e & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (16.8)$$

where θ^e is the angle that vector \mathbf{r}_e makes with the Ox axis.

In theory, this value of force or torque does not depend on the choice of the integration surface Γ' . This is not always verified in computational electromagnetics [Coulomb, Meunier 1984], [Sadowski 1993]. In order to have sufficient precision, this surface must be placed so as to connect the middles of the edges of the tetrahedra (see Figure 16.2). These elements belong to a layer surrounding the modelled object. Depending on the layout of these elements in this layer, two types of surface elements can be distinguished: triangles and quadrangles. Two algorithms are thus required to calculate the surface and the barycentre.

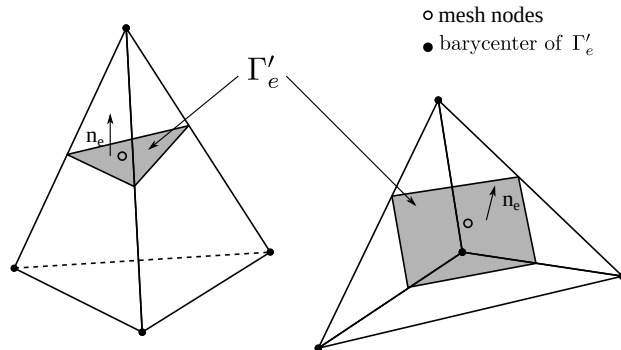


Figure 16.2: Surface element types - Intersection between a tetrahedral mesh and the integration surface

16.2 Virtual work method

16.2.1 Principle

We consider the same domain \mathcal{D}' as defined above (see Figure 16.1). We want to calculate the forces and torque by the virtual work method. This approach is based on the principle of converting magnetic energy into mechanical energy. We can show that the total force, in a direction “s”, is calculated from the variation of the magnetic energy “w” of the system after a move in this same direction. This movement is at constant flux, i.e. constant \mathbf{B} [Coulomb 1983], [Ren, Razek 1992].

$$F_s = -\partial_s w|_{\mathbf{B}=\text{cte}} \quad s = x, y, z \quad \text{avec } w = \int_{\mathcal{D}'} \int_0^{\mathbf{b}} \mathbf{H} \cdot d\mathbf{B} dv \quad (16.9)$$

A similar expression can be established using the co-energy “w’” at constant current, i.e. constant \mathbf{H} .

$$F_s = -\partial_s w'|_{\mathbf{H}=\text{cte}} \quad s = x, y, z \quad \text{avec } w' = \int_{\mathcal{D}'} \int_0^{\mathbf{H}} \mathbf{B} \cdot d\mathbf{H} dv \quad (16.10)$$

Assuming that there are no changes of \mathbf{H} or \mathbf{B} on the boundary Γ during the move, the calculation of w or w' is performed only in domain \mathcal{D}' . Given the considerations on the field and the flux density, the force calculation is obtained using: the derivative of the energy (see expression 16.9) for the vector potential \mathbf{A} and the derivative of the co-energy (see expression 16.10) for the scalar potential Ω [Coulomb 1983].

We can also calculate the torque by differentiation of w or w' with respect to the angle of rotation θ . We thus obtain the following relations:

$$C_z = -\partial_\theta w|_{\mathbf{B}=\text{cte}} \quad C_z = -\partial_\theta w'|_{\mathbf{H}=\text{cte}} \quad (16.11)$$

16.2.2 Discretisation

The goal is to obtain the overall value of the force or torque from a solution of the problem by the finite element method. To do this, it is possible to use a finite differences approach that consists in evaluating the energy w (see expression 16.9) or co-energy w' (see expression 16.10) for two positions s_0 and s_1 of the region \mathcal{D}' . The value of the force is then given by the expressions:

$$F_s = -\frac{w_1 - w_0}{s_1 - s_0} \quad F_s = -\frac{w'_1 - w'_0}{s_1 - s_0} \quad (16.12)$$

This approach thus requires two solutions to the problem. On the one hand, it introduces rounding errors on the calculation of forces [Coulomb, Meunier 1984]. For these reasons, we prefer a method based on the local differentiation of the energy or co-energy. In this case, the calculation of force or torque is obtained by direct differentiation of the energy functions w or w' by using a single resolution by the finite element method. This method gives a general and easy to implement algorithm. Its introduction into a calculation code is achieved by the derivative of the Jacobian matrix [Coulomb 1983].

The calculation is performed by a volume integral in 3D or surface integral in 2D. Usually we choose a layer of elements located in the air and surrounding domain \mathcal{D}' . The movement of this layer results in deformation of the layer elements. The surface nodes are then virtually moved [Coulomb 1983], [Ren, Razek 1992], [Sadowski 1993]. In the case of rotating machines, such a layer is placed in the air gap.

16.2.2.1 Local derivative of the magnetic energy

The use of the vector potential \mathbf{A} , discretised by the edge elements, implies that a flux can be kept constant by the flows of \mathbf{A} . As a result, the calculation of forces is obtained by the derivative of the energy. For each element, the magnetic induction is given by:

$$\mathbf{b}^e = \mathbf{rot} \mathbf{w}^e c_a^e$$

The magnetic energy is then approached by a finite sum on the N_e elements of the previously defined layer:

$$w = \sum_{e=1}^{N_e} w^e = \frac{1}{2} \sum_{e=1}^{N_e} c_a^{eT} S_a^e c_a^e \quad \text{avec } S_a^e = \int_{\mathcal{D}_e} \frac{1}{\mu_0} \mathbf{rot} \mathbf{w}^{eT} \cdot \mathbf{rot} \mathbf{w}^e dv \quad (16.13)$$

w^e is the magnetic energy for a virtually moved element. We note that S_a^e is the elemental stiffness matrix calculated in air. It is recalled that the terms of the stiffness matrix in air S_a and the source vector F_a are written:

$$S_{a\,i,j} = \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{w}^i \cdot \mathbf{rot} \mathbf{w}^j d\mathcal{D} \quad i, j = 1, \dots, N_a \quad (16.14)$$

$$F_{a\,j} = \int_{\mathcal{D}} \mathbf{w}^j \cdot \mathbf{J}_0 d\mathcal{D} \quad (16.15)$$

Because the flow values are kept constant during the movement and the magnetic energy of domain \mathcal{D}' changes, the differentiation is performed only on the terms of matrix S_a^e . The force is thus written as follows:

$$F_s = \frac{1}{2} \sum_{e=1}^{N_e} c_a^{eT} \partial_s S_a^e c_a^e \quad (16.16)$$

The expression for matrix $\partial_s S_a^e$ is more easily obtained by passing from the real element \mathcal{D}_e to the reference element $\hat{\mathcal{D}}_e$ (see annex **L** and **M**):

$$\partial_s S_a^e = \int_{\hat{\mathcal{D}}_e} \frac{1}{\mu_0} \mathbf{rot} w^{eT} [(\partial_s \mathbf{J}^T) \mathbf{J}'^T - (\partial_s \mathbf{J}') \mathbf{J}] \mathbf{rot} w^e d\hat{v} \quad (16.17)$$

\mathbf{J} is the Jacobian matrix and \mathbf{J}' represents the co-matrix (co-factors matrix) transposed from matrix \mathbf{J} .

16.2.2.2 Local derivative of the magnetic co-energy

For the scalar potential Ω , the force is calculated by the derivative of the co-energy. We can have a constant current by fixing the potential values. The latter is discretised by nodal elements. In this case, the magnetic field \mathbb{H} is written $\mathbf{h}^e = -\mathbf{grad} \lambda^e \Omega^e$ for each element. The co-energy (see expression 16.10) is thus given by the following relation:

$$w' = \sum_{e=1}^{N_e} w'^e = \frac{1}{2} \sum_{e=1}^{N_e} \Omega^{eT} S_\Omega^e \Omega^e \quad \text{avec } S_\Omega^e = \int_{\mathcal{D}_e} \mu_0 \mathbf{grad} \lambda^{eT} \cdot \mathbf{grad} \lambda^e dv \quad (16.18)$$

We note that S_Ω^e is the elementary stiffness matrix expressed in air.

It is recalled that the terms of the stiffness matrix in air S_a and the source vector F_a are written:

$$S_{a\,i,j} = \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{rot} \mathbf{w}^i \cdot \mathbf{rot} \mathbf{w}^j d\mathcal{D} \quad i, j = 1, \dots, N_a \quad (16.19)$$

$$F_{a\,j} = \int_{\mathcal{D}} \mathbf{w}^j \cdot \mathbf{J}_0 d\mathcal{D} \quad (16.20)$$

During the movement, the potential values Ω are fixed, but the co-energy of domain \mathcal{D}' changes. As a result, the force is calculated from the derivative of matrix S_Ω^e :

s	node i moved			node i fixed		
	$\partial_s x_i$	$\partial_s y_i$	$\partial_s z_i$	$\partial_s x_i$	$\partial_s y_i$	$\partial_s z_i$
x	1	0	0	0	0	0
y	0	1	0	0	0	0
z	0	0	1	0	0	0
θ	y_i	$-x_i$	0	0	0	0

Table 16.1: derivatives of the coordinates

$$F_s = \frac{1}{2} \sum_{e=1}^{N_e} \Omega^{eT} \partial_s S_\Omega^e \Omega^e \quad (16.21)$$

Using the same notation as before, we obtain (see annex M):

$$\partial_s S_\Omega^e = \int_{\hat{D}_e} \mu_0 \mathbf{grad} \lambda^{eT} [(\partial_s \mathbf{J}^T) \mathbf{J}'^T - (\partial_s \mathbf{J}') \mathbf{J}] \mathbf{grad} \lambda^e d\hat{v} \quad (16.22)$$

The matrix expression in brackets:

$$[(\partial_s \mathbf{J}^T) \mathbf{J}'^T - (\partial_s \mathbf{J}') \mathbf{J}]$$

is the same for both relations 16.17 and 16.22. This represents a considerable advantage. A single algorithm suffices to develop the calculations for both formulations.

16.2.2.3 Derivative of the Jacobian matrix

The calculation of force or torque by the virtual work method is reduced to differentiation of the Jacobian matrix \mathbf{J} . This latter allows moving from the real element, of arbitrary shape, to a reference element of a unique shape (see annex L). The expression of \mathbf{J} is given in annex L, we repeat it here in order to calculate its derivative. For a tetrahedron, whose nodes are identified by the Cartesian coordinates $((x_i, y_i, z_i) \quad i = 1, 4)$, we have the following relation:

$$\mathbf{J} = \mathbf{grad} \hat{\lambda} \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_4 & y_4 & z_4 \end{bmatrix} \quad (16.23)$$

where $\hat{\lambda}$ are the nodal approximation functions defined in the reference system.

These functions are linear, hence their gradient is constant. As a result, the differentiation of \mathbf{J} is performed only on the coordinates of the nodes of the element:

$$\partial_s \mathbf{J} = \mathbf{grad} \hat{\lambda} \begin{bmatrix} \partial_s x_1 & \partial_s y_1 & \partial_s z_1 \\ \partial_s x_2 & \partial_s y_2 & \partial_s z_2 \\ \partial_s x_3 & \partial_s y_3 & \partial_s z_3 \\ \partial_s x_4 & \partial_s y_4 & \partial_s z_4 \end{bmatrix} \quad (16.24)$$

Table 16.1 summarises the derivative values for the different components of the force along x , y and z . For calculation of the torque, we take:

$$s = \theta$$

The calculation of the force thus requires knowledge of the layer elements as well as the virtually displaced nodes (see Figure 16.3).

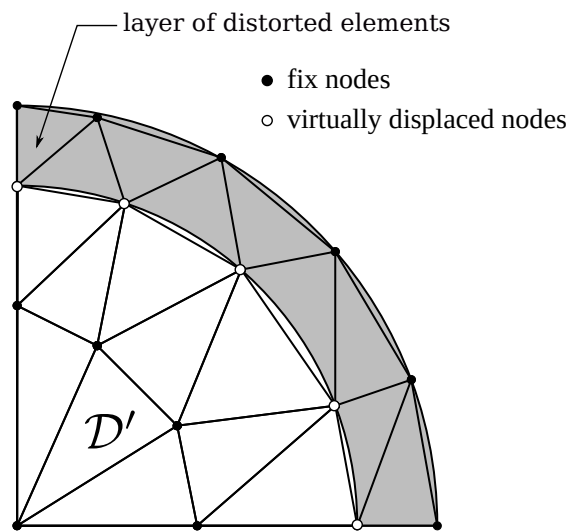


Figure 16.3: Deformed elements and displaced nodes for a triangle mesh.

Chapter 17

Calculating local magnetic flux

17.1 Introduction

It may be interesting to calculate a flux more locally across a surface inside the domain. For formulations using the vector magnetic potential \mathbf{A} , the calculation of such a value is not a problem since it is sufficient to calculate the flow of \mathbf{A} on a contour of a surface. However, for discrete formulations in scalar magnetic potential, since the normal component of the magnetic induction is not preserved, we cannot define the concept of magnetic flux. However, there are several methods to determine an image of such a value.

17.2 Presentation of the problem

Below, we consider a contractible surface S supported by the mesh facets (see Figure 17.1).

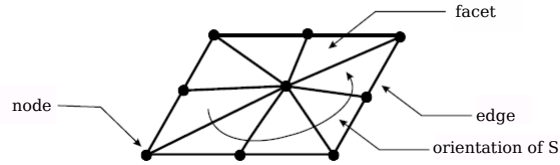


Figure 17.1: Definition of the surface S .

We denote n_a^S the number of edges forming the boundary ∂S and n_f^S the number of facets of S . The orientation of S is fixed arbitrarily, and this implicitly orients its contour ∂S (see Figure 17.1).

To simplify the presentation, we consider the magnetostatic case knowing that this method can be extended to the magnetodynamic case.

17.3 Case of formulation \mathbf{A}

In the case of formulation \mathbf{A} , the magnetic flux Φ_A through the surface S can be directly obtained by the expression of the magnetic induction \mathbf{B}_A .

$$\Phi_A = \int_S \mathbf{B}_A \cdot \mathbf{n} \, ds \quad (17.1)$$

with \mathbf{n} the normal unit vector to S for which the direction is fixed by the orientation of S .

As the flux density \mathbf{B}_A is discretised in the facet element space, its normal component is then preserved across all mesh facets. Expressing the flux density \mathbf{B}_A as a function of the potential \mathbf{A} ($\mathbf{A} \in \mathbf{W}^1$) in equation 17.1, the magnetic flux crossing S is obtained by a simpler expression:

$$\Phi_A = \int_S \mathbf{B}_A \cdot \mathbf{n} ds = \int_S \mathbf{rot} \mathbf{A} \cdot \mathbf{n} ds = \oint_{\partial S} \mathbf{A} \cdot d\partial s \quad (17.2)$$

The path direction of ∂S is determined by the orientation chosen for S . Each edge a of n_a^S is associated with an incidence number δ_a , if the orientations of a and ∂S are the same then $\delta_a = 1$ otherwise $\delta_a = -1$. In these conditions, the expression of Φ_A according to the flow of \mathbf{A} along the edges is equal to:

$$\Phi_A = \sum_{a=1}^{n_a^S} A_a \delta_a \quad (17.3)$$

An example is given in Figure 17.2 where the contour of the surface S is composed of 8 edges.

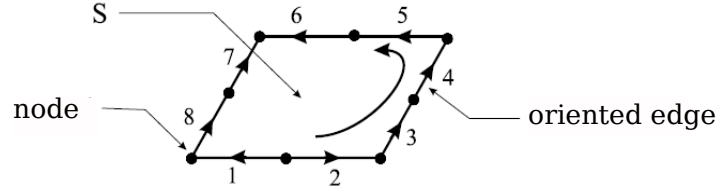


Figure 17.2: Example of calculation of a local magnetic flux by formulation A .

The flux Φ_A through S is thus equal to:

$$\Phi_A = -A_1 + A_2 + A_3 + A_4 - A_5 + A_6 - A_7 - A_8 \quad (17.4)$$

This conventional method is very advantageous in terms of calculation time and simplicity of implementation. Only the nodes belonging to ∂S are to be determined (the edges of ∂S are easily deduced from it) and not the surface S itself.

17.4 Case of formulation Ω

Below, formulation Ω will be assumed to be resolved on the primal mesh. We thus obtain a magnetic field \mathbf{H}_Ω that verifies Ampère's circuital law. On the other hand, the flux density \mathbf{B}_Ω obtained through the constitutive relation does not verify the conservation of flux law and does not have a continuous normal component throughout the domain. To obtain such a field would require use of the dual mesh where the flux density $\hat{\mathbf{B}}_\Omega$ has conservative flux (see Chapter I of [Henneron 2004]).

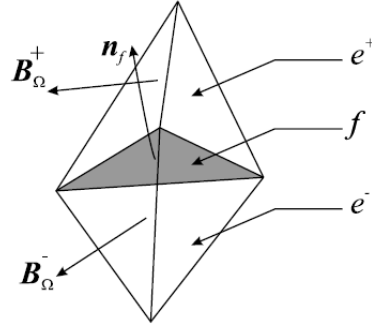
17.4.1 First approach

Since the normal component of \mathbf{B}_Ω is not preserved, the value of the surface integral of the normal component of the flux density through a facet f common to two elements e^+ and e^- is not the same as the expression of \mathbf{B}_Ω on e^+ and e^- (see Figure 17.3).

Two values homogeneous to fluxes Φ_f^+ and Φ_f^- can be calculated:

$$\Phi_f^+ = \int_f \mathbf{B}_\Omega^+ \cdot \mathbf{n}_f df \quad \text{et} \quad \Phi_f^- = \int_f \mathbf{B}_\Omega^- \cdot \mathbf{n}_f df \quad (17.5)$$

with \mathbf{B}_Ω^+ and \mathbf{B}_Ω^- the magnetic induction in two elements e^+ and e^- having the common facet f and \mathbf{n}_f the normal vector to f for which the orientation depends on that of S . Two values Φ_{Cl}^+

Figure 17.3: Example of a facet contained in surface S .

and Φ_{Cl}^- homogeneous to a flux through a surface S are defined as the sum of the two values Φ_f^+ and Φ_f^- :

$$\Phi_{Cl}^+ = \sum_{f=1}^{n_f^S} \Phi_f^+ \quad \text{et} \quad \Phi_{Cl}^- = \sum_{f=1}^{n_f^S} \Phi_f^- \quad (17.6)$$

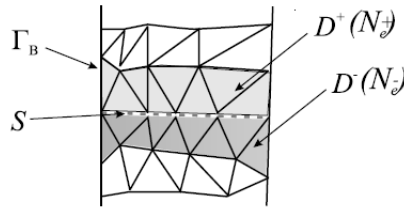
17.4.2 Second approach

The second approach is based on the relation giving the flux through a surface belonging to the boundary of domain \mathcal{D} :

$$\Phi_D = \int_{\mathcal{D}} \mathbf{B}_\Omega \cdot \mathbf{grad} \alpha \, d\mathcal{D} \quad (17.7)$$

with α defined in section 3.2.

If we consider a surface S inside \mathcal{D} but whose boundary ∂S belongs to Γ_B . We can use a relation similar to 17.7. We denote by N_{e+} and N_{e-} the two sets of elements on each side of S and having at least one node belonging to S (see Figure 17.4). These two sets of elements form two domains \mathcal{D}^+ and \mathcal{D}^- .

Figure 17.4: Example of domains resulting from surface S .

We define a function α^+ (α^- respectively) zero on $\mathcal{D} \setminus \mathcal{D}^+$ ($\mathcal{D} \setminus \mathcal{D}^-$ respectively) and defined as follows on \mathcal{D}^+ (\mathcal{D}^- respectively)

$$\alpha^+ = \sum_{n \in S} w_n \quad (17.8)$$

where w_n is the nodal function associated with node n .

In these conditions, two values homogeneous to a flux through the surface S can be calculated by:

$$\Phi_D^+ = \int_{\mathcal{D}^+} \mathbf{B}_\Omega \cdot \mathbf{grad} \alpha^+ d\mathcal{D}^+ \quad \text{et} \quad \Phi_D^- = \int_{\mathcal{D}^-} \mathbf{B}_\Omega \cdot \mathbf{grad} \alpha^- d\mathcal{D}^- \quad (17.9)$$

We show that Φ_D^+ and Φ_D^- are equal because they correspond to fluxes through the dual facets of the edges on each side of S and having a single node on S (see Figure 17.5).

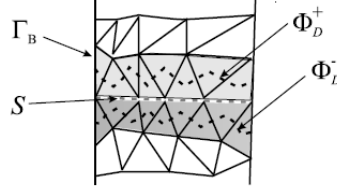


Figure 17.5: Flux through the dual facets.

We recall that this method is only applicable with surfaces S supported by boundary conditions of type $\mathbf{B} \cdot \mathbf{n} = 0$. Next, we will propose an extension to this for surfaces that do not rely on Γ_B .

17.4.3 Third approach

We define an exploratory coil using the boundary ∂S formed by the edges. The magnetic field produced only by the coil traversed by a current of 1 A in a domain \mathcal{D} assumed to be of uniform permeability μ_0 is denoted \mathbf{K}_{sp} .

This field is calculated by the Biot-Savart law at any point M of the domain (see Figure 17.5) by:

$$\mathbf{K}_{sp}(M) = \frac{1}{4\pi} \int_{\partial S} \frac{d\mathbf{l} \times \mathbf{r}}{\|\mathbf{r}\|^2} \quad (17.10)$$

with \mathbf{u} the unit vector of \mathbf{r} and $d\mathbf{l}$ an elemental movement along the edges of ∂S .

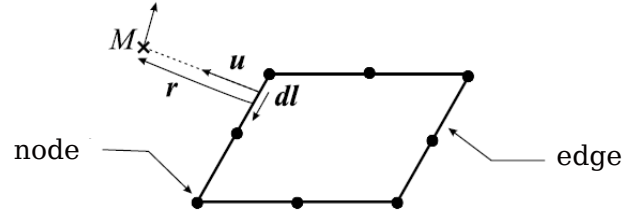


Figure 17.6: Calculation of \mathbf{K}_{sp} at a point M .

The magnetic flux Φ_{sp} is then calculated by:

$$\Phi_{sp} = \int_{\mathcal{D}} \mathbf{B}_\Omega \cdot \mathbf{K}_{sp} d\mathcal{D} \quad (17.11)$$

Vector \mathbf{K}_{sp} cannot be projected into any of the discrete spaces shown above. The numerical calculation of expression 17.11 must be performed as carefully as possible to be precise. The Gauss integration method was used for this calculation, but the selected integration points should not be placed on the boundary of the surface (vector \mathbf{K}_{sp} is not defined on the surface) but inside the elements. The numerical technique used to determine \mathbf{K}_{sp} in one Gauss point is detailed in annex D.

Chapter 18

Calculation of iron losses

In the current context of developing devices that meet sustainable development and energy efficiency criteria, research into the need to save energy, the efficient use of materials in electrical devices and the development of new materials with superior properties are of paramount importance. Recent advances in the electrotechnical industry are due, in large part, to the improvement of the technology for the manufacture of magnetic materials. Rotating and static electrical machines, of all sizes, are usually constructed with soft magnetic materials (sheet). For their appropriate design, it is important to have a good knowledge of the properties of these magnetic sheets.

The magnetic material represents the heart of the operation of an electrical machine and the properties of the material, such as the $B(H)$ magnetic constitutive relation and the iron losses, influence the performance and efficiency of the machine.

In this chapter, we will first introduce the definitions of the various magnetic values that will allow us to explain the physics of a magnetic material. Secondly, the mechanisms behind iron losses will be described using the Bertotti theory. Next, the main difficulties in estimating these iron losses in an electrical machine will be presented. In a final section, we will address the main models used to estimate these losses and present the approach we have chosen for this work.

18.1 Magnetic materials

18.1.1 Magnetic values

A sample of matter is basically described, from the point of view of magnetic properties, as a set of magnetic moments, resulting from the movement of electrons. Conventionally, electrons orbiting the atomic nucleus have a magnetic moment also called the orbital moment:

$$\mathbf{m} = -(e/2m_e) \mathbf{L}$$

where:

- e is the charge;
- m_e is the mass of the electron;
- \mathbf{L} is the angular moment.

In addition to this orbital magnetic moment, electrons have an intrinsic magnetic moment called spin magnetic moment. We thus define the magnetic moment of an atom as the vector sum of those two moments.

At the macro scale, a volume element of a magnetic material is a set of magnetic moments and we can define the magnetisation \mathbf{M} [A/m] of the material such that:

$$\mathbf{M} = \frac{\partial \mathcal{M}}{\partial v} \quad (18.1)$$

where:

- \mathcal{M} is the sum of the magnetic moments;
- ∂v is the volume element considered.

The general relation between magnetic induction \mathbf{B} [T], magnetic field \mathbf{H} [A/m] and magnetisation \mathbf{M} is written as follows:

$$\mathbf{B} = \mu_0 (\mathbf{H} + \mathbf{M}) \quad (18.2)$$

where:

- $\mu_0 = 4\pi \cdot 10^{-7}$ [H/m] is the magnetic permeability of a vacuum;

In a vacuum, the magnetisation \mathbf{M} being zero, the relation $\mathbf{B} = \mu_0 \mathbf{H}$ allows us to consider the flux density and the magnetic field as equivalent quantities, because they are simply linked by the proportionality constant μ_0 . In the presence of magnetic material, the contribution $\mu_0 \mathbf{M}$ reflects the response of the material to external solicitation. This contribution is called magnetic polarisation \mathbf{J} , a quantity with the same unit as \mathbf{B} [T] and the same properties as magnetisation \mathbf{M} . Equation 18.2 is then conventionally written as follows:

$$\mathbf{B} = \mu_0 \mathbf{H} + \mathbf{J} \quad (18.3)$$

The magnetic constitutive relation can also be expressed as:

$$\mathbf{B} = \mu_0 \mu_r \mathbf{H} \quad \text{et} \quad \mathbf{M} = \chi \mathbf{H} \quad (18.4)$$

where:

- μ_r is the relative permeability
- χ is the magnetic susceptibility.

These parameters are linked by the following equation:

$$\mu_r = 1 + \chi \quad (18.5)$$

Based on this general representation of magnetic behaviour, it is possible to describe the behaviour of the three major categories of magnetic materials:

- paramagnetic materials;
- diamagnetic materials;
- ferromagnetic materials.

Below, we will briefly explain the magnetic properties of each category of material.

18.1.2 Classification of magnetic materials

18.1.2.1 Diamagnetism

Diamagnetism is reflected in the appearance within the material of a magnetic field opposed to the applied field. Its origin is the modification of the orbital movement of the electrons around the atomic nucleus following the application of an external magnetic field. As a result, diamagnetic magnetisation is present in all materials, but its contribution to the total magnetisation remains very small compared with other types of magnetisation.

Among the diamagnetic materials (which have only diamagnetic magnetisation) are Cu, Au, Ag, Zn, Pb, etc. These materials thus have a negative magnetic susceptibility, independent of temperature, in the order of 10^{-5} . As a result, the constitutive relation of this type of material can be assimilated to that in a vacuum in the study of electrical machines.

18.1.2.2 Paramagnetism

From the microscopic point of view, paramagnetism is linked to the existence of a permanent magnetic moment that can be carried by atoms or molecules. In the absence of an external magnetic field, the magnetic moments are randomly orientated due to thermal agitation, so the material does not show spontaneous magnetisation. Paramagnetic materials (e.g. Al, Cr, Mn, Na) nevertheless have a low but positive magnetic susceptibility in the order of 10^{-3} to 10^{-5} .

The constitutive relation of these materials can thus be considered as linear and close to that in a vacuum in electrotechnical fields of application.

18.1.2.3 Ferromagnetism

In the case of ferromagnetism, at the microscopic scale, the magnetic moments of spin show strong coupling. Thus, at the scale of a Weiss domain (defined below), there is magnetisation even in the absence of an external field, the magnetisation being qualified as spontaneous. This is due to the fact that atomic moments tend to align spontaneously and parallel to each other, forming an ordering that can be compared to the geometric ordering characteristic of the solid state.

It should be recalled that the theory of paramagnetism considers atoms to be independent of each other, which is not the case for ferromagnetism. There is an exchange energy between the magnetic moments carried by the atoms which tend, by a collective effect, to align in the same direction. The exchange energy may be written, taking into account the magnetic moments \mathbf{S}_i and \mathbf{S}_j the two neighbouring atoms, in the following form:

$$W_{ij} = -2 J_{ij} \mathbf{S}_i \mathbf{S}_j \quad (18.6)$$

In this expression, proposed by Heisenberg, J_{ij} is the exchange integral. The value of this coupling factor favours the appearance of a ferromagnetic order if $J_{ij} > 0$ or an antiferromagnetic order if $J_{ij} < 0$. In the case of a ferromagnetic material, the magnetisation tends to orient along the preferred directions (easy axis of magnetisation) determined by the crystal structure or by the shape of the sample. To change the direction of a magnetic moment, we can either apply a magnetic field or provide energy by raising the temperature. It should be noted that increasing the temperature above a threshold temperature, called the Curie temperature, leads to a reversible collapse of the spontaneous magnetisation making the system paramagnetic.

Ferromagnetic materials (e.g. Fe, Co, Ni and their alloys) show high susceptibility in the order of 10^3 and are the main materials used in electrotechnical energy conversion devices. Ferromagnetic materials can be subdivided into two groups: soft materials and hard materials (permanent magnets). Soft magnetic materials can be easily magnetised with weak magnetic fields; they are used in electrical machines to focus and channel the magnetic flux.

At industrial frequencies, FeSi sheet of 0.35 to 0.65 mm thickness is generally used and for frequencies above 10 kHz, amorphous materials are used which have saturation flux density, thickness and losses that are lower than for conventional sheet. Hard magnetic materials (permanent magnets) are used as a source of magnetic field in electrical machines.

Here, we are only interested in soft ferromagnetic materials.

18.1.3 Configuration in magnetic domains

The first modern theory of ferromagnetism, which remains valid today, was proposed by Pierre Weiss in 1906-1907 [Weiss 1907], [Brissoneau 1997] and the first experimental work was conducted in the 1930s. On a macro scale, the spontaneous magnetisation observed on a microscopic scale disappears. P. Weiss's theory explains the existence of a demagnetised state and states that a ferromagnetic material is subdivided into several domains called Weiss domains inside which the magnetisation is uniform and aligned in the same direction for each domain but different from one domain to another. These domains are separated by walls (Bloch walls) of small thickness compared with the size of the domain, from a few hundred to a few thousand Angström. In these walls, the orientation of the magnetisation varies rapidly from one direction in one domain to another in the neighbouring domain.

18.1.3.1 Weiss domains

In section 18.1.2.3 we introduced the concept of exchange energy between the magnetic moments of the neighbouring atoms which, despite the thermal agitation, allows the magnetic moments to align. This implies that the overall moment of the system would be the saturation moment. However, there are two other types of energy that oppose the exchange energy: magnetostatic energy and magnetocrystalline anisotropy energy. It is the appearance of Weiss domains in the ferromagnetic body that effectively allows minimisation of the sum of the three types of magnetic energy.

18.1.3.1.1 Anisotropy energy

In a crystalline structure, there are easy axes of magnetisation along which the energy required to magnetise the material is less than in other directions. For example, for a monocrystalline sample, if the excitation field is applied along the easy axis of magnetisation, the polarisation \mathbf{J} reaches saturation almost instantly.

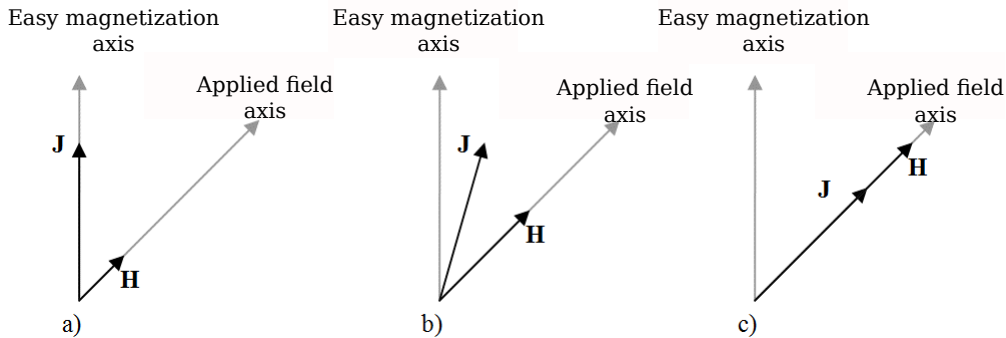


Figure 18.1: Polarisation behaviour \mathbf{J} when applying a field \mathbf{H} .

If, on the other hand, an excitation field is applied on an axis other than the easy axis of magnetisation, the polarisation \mathbf{J} does not behave as in the previous case. As shown in Figure 18.1a, if a field is applied in a direction outside the easy axis of magnetisation, the materials initially polarise along the nearest easy axis of magnetisation. In this case, depending on the axis of application of the magnetic field, the magnetisation contribution corresponds to the projection

of the polarisation on this same axis. If the amplitude of the magnetic field continues to grow, there is a rotation of the polarisation \mathbf{J} and a slow increase of the projection on the axis of application of the field that approaches the saturation level (18.1b). The field needed to change the polarisation direction is called the “anisotropy field”. Then, if the magnetic field continues to increase, the polarisation will align with the magnetic field and the material will be saturated in the same direction as the excitation field (18.1c). As a result, the volume energy required to reach saturation in a direction other than the easy axis of magnetisation will be higher.

18.1.3.1.2 Magnetostatic energy

This energy results from the magnetic interactions between the magnetic moments, since each magnetic moment is subject to a local field created by all other magnetic moments. P. Brissoneau [Brissoneau 1997] suggested an expression for magnetostatic energy by representing magnetised matter as a set of magnetic moments in a vacuum.

$$W_m = \frac{1}{2} \iiint_V \mathbf{M} \cdot \mathbf{H}' dv \quad (18.7)$$

where:

- V is the system volume;
- \mathbf{H}' is the local field.

In the absence of external field, \mathbf{H}' is due to the presence of the demagnetising field created by the moments of the structure. These are the result of the appearance of fictitious magnetic masses within the material due to the local divergence of the magnetisation.

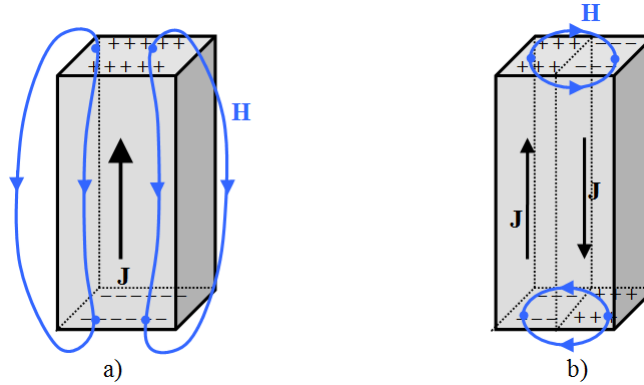


Figure 18.2: System with uniform magnetisation a); Fractional structure in two domains with anti-parallel magnetisations b).

In Figure 18.2a, the presence of the fictitious magnetic poles gives rise to a magnetic field which, according to equation 18.7, will introduce significant magnetostatic energy. However, as the magnetic moments are aligned in one direction, the easy axis of magnetisation, exchange energy and anisotropy energy are minimised. In the second configuration (Figure 18.2b), the structure is divided into two domains with anti-parallel magnetic moments. The magnetic field thus re-loops in the ends of the domains, thus limiting the magnetic field in comparison to the first configuration. As a result, this configuration minimises the magnetostatic energy but the exchange energy increases because there are moments anti-parallel to the interface between the domains. In addition, the contribution of anisotropy energy favours the orientation of magnetic moments along a preferred direction of the crystal to minimise the overall energy of the system.

Thus, overall, the total energy of the material (sum of the three contributions mentioned above) is minimised by the division of the material into Weiss domains. The size of these domains varies

depending on the material and the metallurgical quality. The order of magnitude of the domains can range from a few tens of nanometers to a few hundred microns.

18.1.3.2 Bloch walls

As mentioned above, a ferromagnetic material is subdivided into several domains. This structure shows transition zones (Bloch walls) between neighbouring domains where the orientation of magnetic moments changes from one domain to the neighbouring domain.

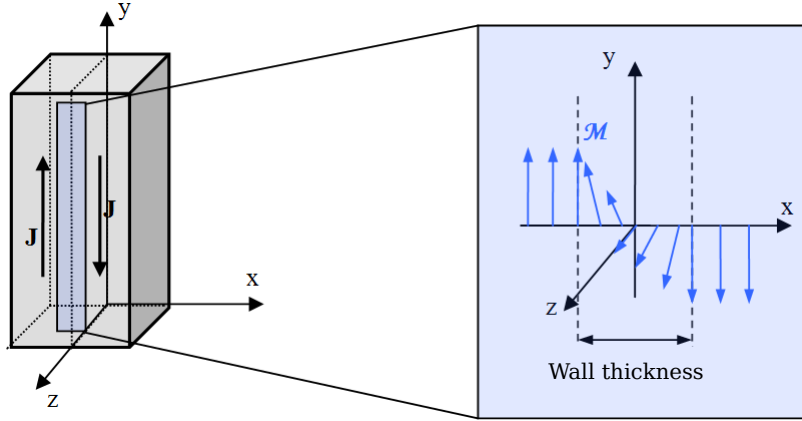


Figure 18.3: Rotation of magnetic moments between two domains at 180° .

As shown in the figure above, the change of orientation of the magnetic moments is not sudden and takes place gradually in the thickness of the wall. Thus, the exchange energy needed for a gradual transition is less than for a sudden transition [Brissonneau 1997]. The change in exchange energy is thus inversely proportional to the size of the wall.

However, if we think in terms of anisotropy energy, a large wall thickness implies an increase in anisotropy energy because there are several magnetic moments aligned in unfavourable directions. In fact, the optimal width of this wall is obtained for the configuration of minimum overall energy.

18.1.4 Magnetisation process - First magnetisation curve

If a ferromagnetic material is demagnetised, the magnetisations associated with Weiss domains show random orientations, resulting in a zero total magnetisation. Note that, in practice, this demagnetised state can be obtained by natural relaxation of the material or by application of an alternative field of initially high amplitude (to saturate the material) then becoming weaker until the excitation is cancelled out. If an increasing magnetic field is then applied to the material, the magnetic moments will tend to align with the direction of the applied field. This means that the Bloch walls move within the material.

However, the movement of the Bloch walls is hindered by the imperfections present within the material. These imperfections are due in particular to non-magnetic and ferromagnetic impurities as well as to dislocation stresses, grain boundaries and metallurgical treatments. These defects have the direct consequence, as will be seen later, of a reduction in permeability and an increase in magnetic losses.

Thus, depending on the intensity of the magnetic field applied, the magnetisation mechanism can be described, as a first approach, as the succession of three main mechanisms (Figure 18.4):

- Region A: this is the area of the fields where the movement of the walls can be considered as an elastic deformation. Conceptually, because they are not rigid, they can deform on the anchor sites. Thus, if the increase in the external field is not sufficient to dislodge the wall, it will deform without causing a sudden change in the magnetisation. This process is reversible: if the external field is removed, the system returns to its initial state.

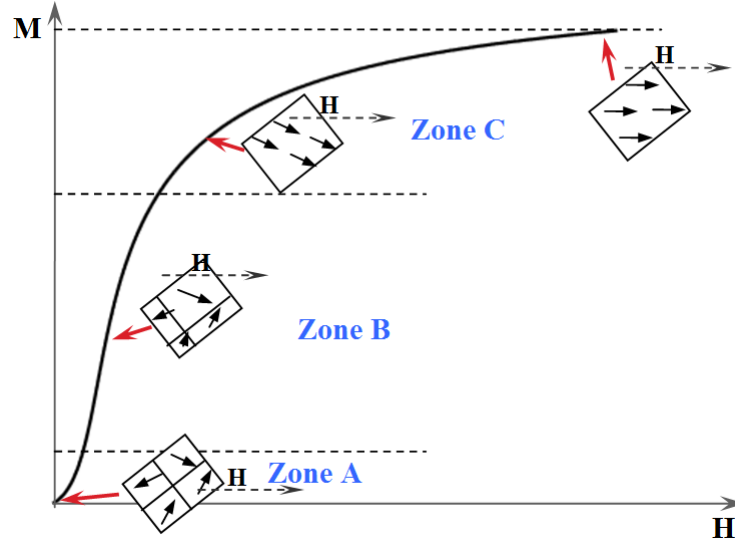


Figure 18.4: First magnetisation curve.

- Region B: in this region, the external magnetic field intensity reaches a level allowing the walls to break free of the anchor sites. Hence, domains for which the initial magnetisation was in the same direction, or close to the external magnetic field direction, will expand in volume at the expense of others.
- Region C: To reach this region, the intensity of the magnetic field must be very high. Magnetisation then begins to saturate and the Bloch walls disappear. We basically have a structure with a single magnetic domain where the magnetic moments start to align in the same direction as the applied magnetic field. This process of rotation of magnetic moments is reversible.

18.2 Magnetic losses

When a ferromagnetic material is subjected to a variable field over time, it is the site of energy dissipation, more commonly known as magnetic losses or iron losses. According to the approach proposed by Bertotti, [Bertotti 1988] these losses can be broken down into three contributions:

- Losses by hysteresis
- Induced current losses (or conventional losses)
- Anomalous losses.

Remark 18.2.1 *In reality, these three components are due to the induced currents that develop in the material, but at different scales (microscopic and macroscopic).*

Below, we briefly present these three contributions to the total losses. We take the case of a ferromagnetic sheet whose length and width are much greater than its thickness, and in conditions of excitation dynamics (frequency) such that the thickness of the skin remains compared with the thickness of the sheet. The magnetic field can then be considered, at first approach, uniform in the thickness of the sheet. In addition, we will now work with the usual magnetic induction value \mathbf{B} linked to the magnetisation \mathbf{M} by equation 18.2.

18.2.1 Hysteresis losses

Hysteresis losses are associated with the movement of the Bloch walls (see section 18.1.3), which is mostly irreversible and causes the magnetic induction \mathbf{B} to lag behind the excitation field \mathbf{H} . This delay is observed at the macro scale in the form of a material-specific hysteresis cycle.

In addition, from thermodynamic considerations [Bertotti 1998] it can be shown that the area described by this cycle corresponds to the volume energy dissipated over a period. Thus, as mentioned above, the movement of the walls is not continuous, but by sudden jumps from one anchor site to another (Barkhausen jumps, see Figure 18.5)

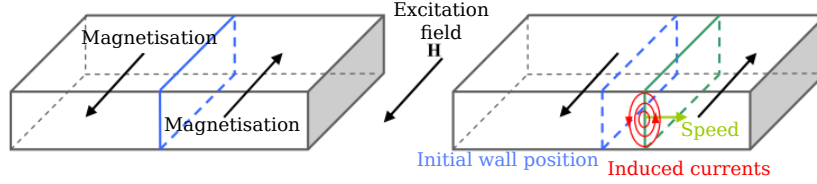


Figure 18.5: Microscopic induced currents when moving a wall by 180° .

These jumps are associated with local flux variations, giving rise to microscopic induced currents in the Bloch wall region.

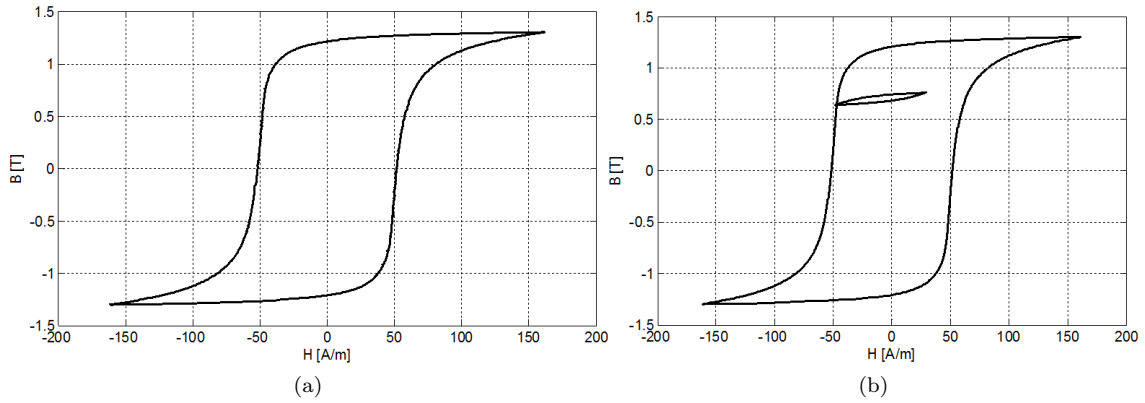


Figure 18.6: a) Major centred hysteresis cycle b) Major centred hysteresis cycle with a minor cycle

In addition, depending on the waveform of the magnetic induction, hysteresis cycles may have minor, non-centred cycles (Figure 18.6). These minor cycles induce additional losses also determined by their area. Generally, the energy supplied to the material to cover a complete cycle is written:

$$W = \oint \mathbf{H} \cdot d\mathbf{B} \quad [\text{J/m}^3] \quad (18.8)$$

This energy is converted to heat during the magnetisation process and represents the volume losses by hysteresis in the static case (low frequency or dynamic).

$$P_h = f \oint \mathbf{H} \cdot d\mathbf{B} \quad [\text{W/m}^3] \quad (18.9)$$

18.2.2 Induced current losses

In the dynamic state, in addition to steady state losses, losses due to macroscopic induced currents, linked to the conductivity σ of the material, become non-negligible. In Figure 18.7 we can see the

induced currents that develop in the thickness of the sheet.

In this figure, the magnetic field and flux density are orientated along axis (Oz); the electric field \mathbf{E} and the induced current density \mathbf{J} are directed along axis (Ox). We assume that the excitation field dynamics \mathbf{H} are weak enough to have a uniform field in the sheet and thus neglect the skin effect.

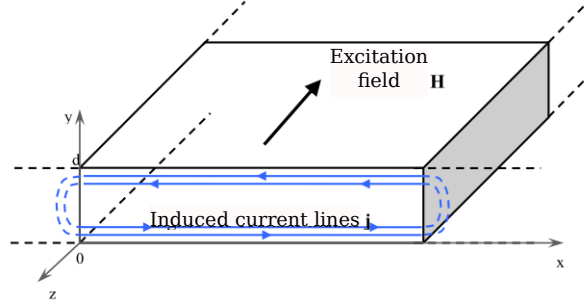


Figure 18.7: Development of induced currents in the thickness of a sheet.

In the case of a sheet of dimensions, in the plane, that are infinite in relation to its thickness, the expression of the volume losses by induced currents is given by [Bertotti 1998]:

$$p_{ci} = \frac{1}{d} \int_0^d \frac{j^2(y, t)}{\sigma} dy = \frac{\sigma d^2}{12} \left(\frac{dB}{dt} \right)^2 \quad (18.10)$$

The mean value over an excitation field period is thus expressed as follows:

$$P_{ci} = \frac{\sigma d^2}{12} \frac{1}{T} \int_0^T \left(\frac{dB}{dt} \right)^2 dt \quad [\text{W/m}^3] \quad (18.11)$$

where:

- T is the time period of the magnetic induction \mathbf{B} ;
- d is the thickness of the sheet.

In the sinusoidal case, the above expression can be written as follows:

$$P_{ci} = 2\pi^2 \left(\frac{\sigma d^2}{12} \right) f^2 B_m^2 \quad [\text{W/m}^3] \quad (18.12)$$

We observe that the induced current losses are proportional to the square of the thickness of the sheet d and to the square of the frequency and induction field \mathbf{B} . These losses also change linearly with the conductivity of the material.

From the point of view of the magnetisation cycle, in the dynamic state, the induced currents produce a swelling of the $B(H)$ cycle as shown in Figure 18.8. This is referred to as a loss cycle, in particular because the cycle includes static losses and macroscopic induced current losses.

In the case of electrical machines, these losses can be non-negligible for several reasons.

- Today, machines are largely powered by static converters, which introduce time harmonics of currents that translate directly into magnetic field harmonics.
- The layout of the coils introduces space harmonics. The magnetomotive force of the air gap is thus not sinusoidal, hence the space harmonics of the magnetic field.
- The stator and/or rotor notches introduce a variation in the air gap reluctance that also induces changes in the magnetic field.

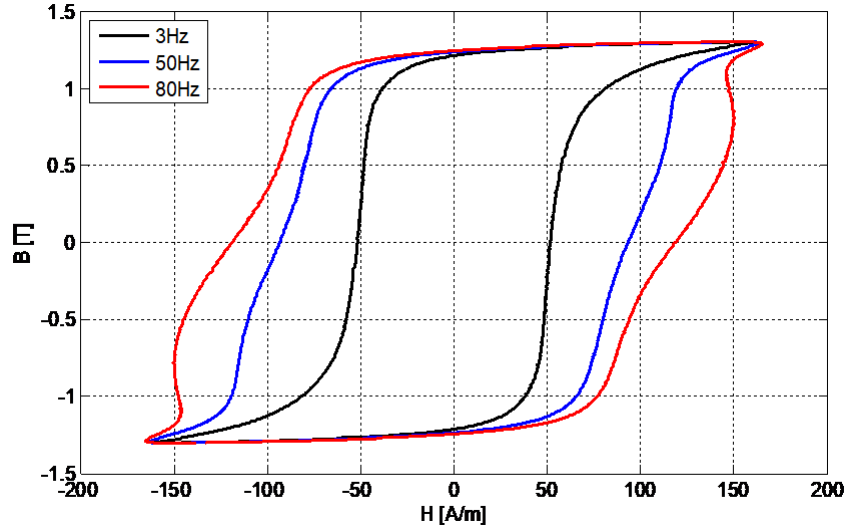


Figure 18.8: Swelling of the $B(H)$ magnetisation cycle in the dynamic state.

- Finally, there are additional end losses introduced by the winding overhang of stators and sometimes rotors, which create additional induced current losses in the magnetic materials located at the ends of the electrical machine.

18.2.3 Anomalous losses

These losses are caused by the movement of the Bloch walls in the dynamic state. These movements are not independent and interact, leading to the appearance of localised induced currents in the vicinity of the walls. This phenomenon can be considered uniform over the whole material and depends strongly on the frequency of the excitation field [Bertotti 1998].

In 1990, Fiorillo and Novikov [Fiorillo, Novikov 1990], based on Bertotti's theory, showed that the average value of anomalous losses, in the case of laminated material and over an electrical period, can be expressed as follows:

$$P_{exc} = \sqrt{\sigma G V_0 S} \frac{1}{T} \int_0^T \left| \frac{dB}{dt} \right|^{1.5} dt \quad [\text{W/m}^3] \quad (18.13)$$

where:

- G is the coefficient of friction between the magnetic domains;
- V_0 is a parameter that characterises the statistical distribution of the local coercive field;
- S is the transverse surface of the laminated material.

If the magnetic induction is sinusoidal, the expression for the anomalous losses becomes:

$$P_{exc} = 8,764 \sqrt{\sigma G V_0 S} f^{1.5} B_m^{1.5} \quad [\text{W/m}^3] \quad (18.14)$$

These losses are influenced by the conductivity of the material, the intensity and frequency of excitation, or the level of impurities present in the material.

18.2.4 Rotational field losses

In electrotechnical applications, the magnetic field is not always unidirectional and orientated according to the easy axis of magnetisation or the transverse axis. In the yoke of electrical machines, for example, or in the T-joints of the magnetic circuits of three-phase transformers, the combination of fields associated with the different phases leads to the appearance of a locally rotating flux density. In general, the flux density module describes a more or less ellipsoidal, even circular shape. Thus, if we consider a circular flux density regime of amplitude B and constant angular velocity ω , the flux density can be broken down into two axes in the sheet plane x and y in the form:

$$\begin{cases} B_x(t) &= B \cos \omega t \\ B_y(t) &= B \sin \omega t \end{cases} \quad (18.15)$$

The rotational losses over a cycle can then be expressed by the following relation [Moses 1992]:

$$P_{rot} = \frac{1}{T} \int_0^T \frac{d\theta}{dt} |\mathbf{H}| \cdot |\mathbf{B}| \sin \alpha dt \quad (18.16)$$

where:

- α is the angle between \mathbf{H} and \mathbf{B} ;
- θ is the angle between \mathbf{B} and a given direction.

In practice, it can be seen that the iron losses in the rotating field and the uni-directional field evolve differently. The difference is explained by the complex behaviour during the magnetisation mechanism involved. In the case of a uni-directional field, the flux density undergoes continuous variation, during which the Bloch walls and the magnetic domains are continuously modified. In the case of a circular field, however, the amplitude of the flux density remains constant and only the projections of the field vary in amplitude.

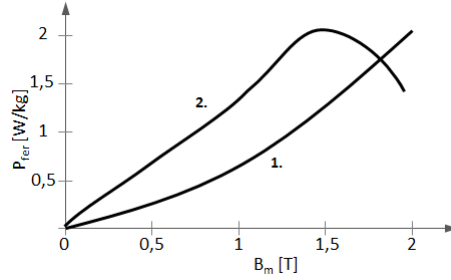


Figure 18.9: Magnetic losses 1) in a uni-directional field and 2) in a rotating field.

For weak fields, rotational field losses, for FeSi-type sheet with non-orientated grain (N.O.), can have values double those in a uni-directional field [Moses 1992], [Enokizono et al 1990]. In a rotating field, these losses can be approximated by the sum of the uni-directional field losses along the rolling direction and along the transverse direction. Conversely, for fields of very high amplitude, rotational field losses decrease rapidly as a function of the amplitude of B while uni-directional field losses continue to increase as a function of B (see Figure 18.9). This is generally observed for flux density values close to saturation.

18.3 Description of the iron loss calculation procedure

Since the calculation of iron losses and the modelling of soft materials are highly interdependent, the calculation of losses can be considered in two ways: either by modelling the $B(H)$ magnetic

constitutive relation by a hysteresis cycle, in this case the losses are calculated directly in the code, or by neglecting the effect of hysteresis on the flux distribution in the device and subsequently calculating the losses from theoretical or experimental formulas. In `code_Carmel`, the second method of calculating losses has been chosen.

The procedure for estimating iron losses in post-processing is described in a simplified manner in Figure 18.10 and detailed in the following sections. The `code_Carmel` calculation code has two versions. A first version is dedicated to permanent states. The calculations are performed in complex mode, harmonic by harmonic. A second version, called the time-based version, where calculations are carried out step by step over time. In the latter, physical values can vary freely over time.

The loss calculation procedure has been implemented in both versions. The approach used in the time-based version of the code is detailed below.

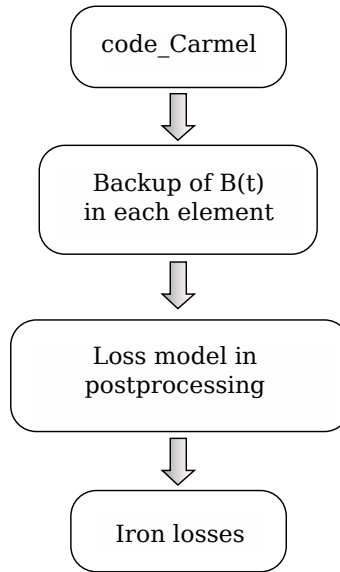


Figure 18.10: Simplified diagram of the iron loss calculation procedure.

As a first step, resolution of the electromagnetic problem is performed, with `code_Carmel`, using one of the two formulations. For a magnetostatic problem, the calculation can be performed with the vector potential formulation \mathbf{A} or with the scalar potential formulation Ω . In the case of a magnetodynamic problem, one of the two mixed formulations $\mathbf{A} - \varphi$ or $T - \Omega$ is used.

Secondly, by means of a procedure that we have implemented in post-processing, we save, over a period of time, the waveform of the flux density $\mathbf{B}(t)$ in each element of the media subject to iron loss. The magnetic induction, although predominantly pulsing in electrotechnical devices, may exhibit local rotating-field behaviour as a result of the local combination of fluxes from different phases. More generally, this behaviour can be described as ellipsoidal and induces additional losses that need to be taken into account. As a result, when calculating the iron losses, it is necessary to consider the two spatial components of the flux density in the plane of the sheet [Bastos, Sadowski 2003]. Hence, the waveforms of the flux density are stored in two files that correspond to the decomposition of field \mathbf{B} along the two spatial axes (in the plane of the sheets).

Lastly, a final procedure was coded, in order to make use of the two files created at the end of the electromagnetic calculation. First, the procedure determines, for each element, the direction in which the flux density module is maximum (called the “Large axis”) and locally associates it with a new coordinate system as shown in Figure 18.11. Thus, at each time step, vector \mathbf{B} is decomposed in order to extract the time waveforms along the large axis and the small axis.

Next, it is possible to choose from several subroutines of the iron loss calculations, implemented

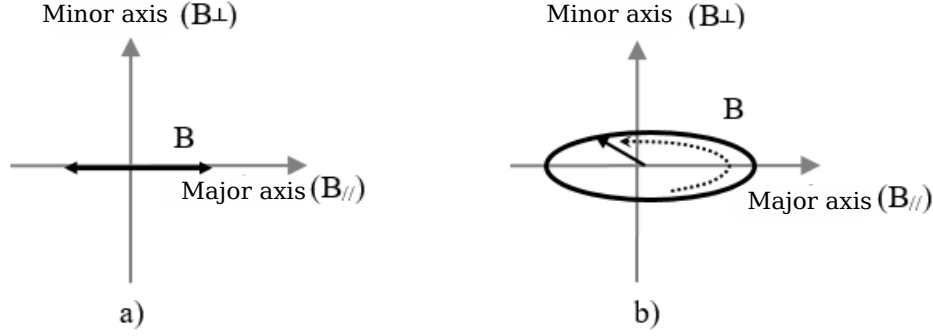


Figure 18.11: Flux density locations a) alternating b) rotational.

subsequently, which provide the density of iron losses in $[\text{W}/\text{m}^3]$ for each element and along the two axes. Finally, the loss density of each element is multiplied by the volume (V_i) of the element under consideration and then summed with losses of all other elements of the system to result in the total iron losses. The various subroutines implemented are introduced below.

The first subroutine calculates iron losses, expressed in W , based on the peak value of the flux density B_m (**model M1**):

$$P_{tot} = \sum_{i=1}^n \left[k_h f \left(B_{\perp,m}^{\alpha} + B_{//,m}^{\alpha} \right) + k_{ci} f^2 \left(B_{\perp,m}^2 + B_{//,m}^2 \right) + k_{exc} f^{1,5} \left(B_{\perp,m}^2 + B_{//,m}^2 \right)^{\frac{3}{4}} \right] V_i \quad (18.17)$$

With:

- k_h , k_{ci} and k_{exc} the iron loss coefficients;
- f the frequency;
- n the total number of elements.

The second subroutine calculates iron losses from the equation below (**model M2**);

$$P_{tot} = \sum_{i=1}^n \left\{ k_h f \left[\left(\frac{\Delta B_{\perp}}{2} \right)^{\alpha} + \left(\frac{\Delta B_{//}}{2} \right)^{\alpha} \right] + \frac{k_{ci}}{2\pi^2 T} \int_0^T \left[\left(\frac{dB_{\perp}}{dt} \right)^2 + \left(\frac{dB_{//}}{dt} \right)^2 \right] dt + \frac{k_{exc}}{8,76 T} \int_0^T \left[\left(\frac{dB_{\perp}}{dt} \right)^2 + \left(\frac{dB_{//}}{dt} \right)^2 \right]^{\frac{3}{4}} dt \right\} V_i \quad (18.18)$$

The third subroutine calculates the quasistatic term based on the peak flux density value of each harmonic k obtained by the Fourier series decomposition and the dynamic components according to the time derivative of the flux density (**model M3**).

$$\begin{aligned}
P_{tot} = \sum_{i=1}^n \left[\sum_{j=1}^k k_h f_j \left(B_{\perp,m,j}^{\alpha} + B_{//,m,j}^{\alpha} \right) \right] V_i \\
+ \sum_{i=1}^n \left\{ \frac{k_{ci}}{2\pi^2} \frac{1}{T} \int_0^T \left[\left(\frac{dB_{\perp}}{dt} \right)^2 + \left(\frac{dB_{//}}{dt} \right)^2 \right] dt \right. \\
\left. \frac{k_{exc}}{8,76} \frac{1}{T} \int_0^T \left[\left(\frac{dB_{\perp}}{dt} \right)^2 + \left(\frac{dB_{//}}{dt} \right)^2 \right]^{\frac{3}{4}} dt \right\} V_i \quad (18.19)
\end{aligned}$$

Finally, the last two subroutines calculate the static component of the losses using a hysteresis model (Jiles-Atherton or Preisach) and the dynamic components according to the time derivative of the flux density (**models M4 and M5**).

$$\begin{aligned}
P_{tot} = & \text{Modèle d'hystérésis} \\
& \text{statique} \\
& + \sum_{i=1}^n \left\{ \frac{k_{ci}}{2\pi^2} \frac{1}{T} \int_0^T \left[\left(\frac{dB_{\perp}}{dt} \right)^2 + \left(\frac{dB_{//}}{dt} \right)^2 \right] dt \right. \\
& \left. \frac{k_{exc}}{8,76} \frac{1}{T} \int_0^T \left[\left(\frac{dB_{\perp}}{dt} \right)^2 + \left(\frac{dB_{//}}{dt} \right)^2 \right]^{\frac{3}{4}} dt \right\} V_i \quad (18.20)
\end{aligned}$$

For all embedded approaches, we consider the contributions given by the two spatial components to be independent of each other and the losses are given by the sum of the two contributions. Hereafter, for the sake of simplicity, we will refer to the different iron loss models by their notation (models M1, M2, M3, M4 and M5).

In addition to the calculation of iron losses, the program developed also provides us with the density map of iron losses and tracking of the locations in the elements wanted.

In the methodology presented, data on the values of the magnetic induction in the elements of the magnetic media, as a function of time, are stored as files. Storing the magnetic induction $\mathbf{B}(t)$ in this way can cause a problem with file sizes, which can be significant depending on the number of mesh elements. On the other hand, the major advantage lies in the fact that, once the electromagnetic problem has been solved, which is often a time-consuming step, the files are saved and can be manipulated to calculate the losses using either of the iron loss approaches with, possibly, different values for the coefficients. Because iron loss calculation procedures do not require significant execution time, this may be the best option.

Chapter 19

Exploratory points

This paragraph describes how to locate a point, *e.g.*, forming part of a section line, in a finite element and the conversion of its coordinates in the reference element system. We illustrate this problem below on two test cases, using the solution of reference calculated using the time-based version of code_Carmel¹. As it stands, tetrahedron, prism, and hexahedron elements of the first order are possible. We are having trouble in obtaining the coordinates of the point in the reference hexahedron, independent of the re-orientation of the elements practised in the code.

19.1 Search method

In code_Carmel, when exploratory points are defined, we first seek the elements containing each of these points, by checking, for all “orientated” faces of the element if the point is on the inside of the element (see annex I). This operation is carried out from the original mesh, because we know the order of the nodes provided for the element.

After re-orienting the elements, *i.e.*, the node indices, comes the step of finding the coordinates of the point in the reference element. These coordinates will be used directly for interpolation of the field at the appropriate location.

The transformation of geometric coordinates, in a finite element, to change the coordinates of a point in the reference element to the coordinates of this point in real space, is very well defined in the literature. It uses **nodal interpolation functions** and poses no practical difficulties because the transformation is analytical [Dhatt, Thouzot 1984, Sec 1.5]. The reverse transformation to change the coordinates from real space to coordinates in the reference element, is not obvious for non-tetrahedral elements, because this transformation is not linear and this several notations for this transformation are possible. These notations are equivalent on paper but do not give the same results in practice. Although this is not mentioned in the literature, we show, case by case, that it is possible to use a method based on the **Jacobian matrix of the geometric transformation**². This matrix is natively available in a finite element code and this method works in practice³. An equivalent method uses **barycentric coordinates** [Dhatt, Thouzot 1984, Sec. 2.5.1]. The latter is used in practice in the harmonic version of Code_Carmel3D [Bereux 2008]. After a discussion with Patrick Dular (University of Liège), this second method does not work well with elements of order 2 and higher.

¹code_Carmel (<http://code-carmel.univ-lille1.fr>) is co-developed by the LAMEL laboratory resulting from a partnership between the L2EP laboratory (<http://l2ep.univ-lille1.fr>) and EDF R&D (<http://www.edf.fr>). The version used to calculate section lines is in development.

²The use of the Jacobian matrix was suggested by Thomas Henneron and Yvonnick le Ménach (L2EP).

³It gives good results for an extruded mesh.

19.1.1 Nodal function method

Here we take up the results and definitions of [Dhatt, Thouzot 1984]. We know that code `_Carmel` uses finite elements *isoparamétriques*. In these conditions, we wish to express the coordinates of a point, known in real space $\mathbf{x} = (x, y, z)$, in the system of the reference element $\boldsymbol{\xi} = (\xi, \eta, \zeta)$. This geometric transformation from reference element to real element calls on n interpolation functions $N_i(\xi, \eta, \zeta)$ where $i \in [1, n]$ and n is the number of nodes in the element. The geometric transformation is thus written:

$$\mathbf{x} = \sum_{i=1}^n N_i(\xi, \eta, \zeta) \mathbf{x}_i \quad (19.1)$$

where \mathbf{x}_i represents the coordinates of node i in real space.

A more detailed notation leads to:

$$\begin{aligned} x &= \sum_{i=1}^n N_i(\xi, \eta, \zeta) x_i \\ y &= \sum_{i=1}^n N_i(\xi, \eta, \zeta) y_i \\ z &= \sum_{i=1}^n N_i(\xi, \eta, \zeta) z_i \end{aligned}$$

For first order elements, the interpolation functions N_i involve linear, bi-linear or tri-linear polynomials in ξ , η and ζ .

Expression (19.1) is difficult to reverse to express $\vec{\xi} = (\xi, \eta, \zeta)$ from $\vec{x} = (x, y, z)$, as we wish. We propose a reformulation below, based on the Jacobian matrix, which allows us to find $\vec{\xi} = (\xi, \eta, \zeta)$.

19.1.2 Jacobian matrix method

The Jacobian matrix, denoted J in [Dhatt, Thouzot 1984], allows the deformation of the reference element to be expressed as a real element, *i.e.* its expansion and rotation with respect to a coordinate system linked to the element. This is the matrix 3×3 defined by:

$$J = \begin{pmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{pmatrix} \quad (19.2)$$

In order to express the coordinates in the real coordinate system of the domain under study, this transformation must be completed by translation into the real coordinate system, of the reference element, *i.e.* the translation of centre O reference frame, *i.e.*, the real coordinates (x_0, y_0, z_0) of the centre of the reference frame. The geometric transformation (19.1) is then written, in matrix form:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} + {}^t J \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} \quad (19.3)$$

where ${}^t J$ is the transposed Jacobian matrix.

It suffices to invert (19.3) to obtain the coordinates (ξ, η, ζ) of the point sought in the reference element. The Jacobian matrix J generally depends on these coordinates (ξ, η, ζ) . We thus denote it $J(\xi)$ and the inverted equation (19.3) is written:

$$\begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} = {}^tJ^{-1}(\xi) \left\{ \begin{pmatrix} x \\ y \\ z \end{pmatrix} - \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} \right\} \quad (19.4)$$

In the case where the Jacobian matrix depends on the coordinates (ξ, η, ζ) , equation (19.4) to be solved is non-linear. An iterative resolution method is thus required. The substitution method works very well in practice on first order elements⁴. During the iteration, the first calculation of the Jacobian matrix can use any point, *e.g.* the centre of the reference element. Then this point will be updated with the result of the previous iteration.

This Jacobian matrix is generally constructed using interpolation functions N_i of the reference element (see equation 19.1). More precisely by the matrix product of the transposed gradient of the set of N_i by the matrix made up of the real coordinates of the n nodes of the element:

$$J = \begin{pmatrix} \frac{\partial N_1}{\partial \xi} & \frac{\partial N_2}{\partial \xi} & \dots & \frac{\partial N_n}{\partial \xi} \\ \frac{\partial N_1}{\partial \eta} & \frac{\partial N_2}{\partial \eta} & \dots & \frac{\partial N_n}{\partial \eta} \\ \frac{\partial N_1}{\partial \zeta} & \frac{\partial N_2}{\partial \zeta} & \dots & \frac{\partial N_n}{\partial \zeta} \end{pmatrix} \times \begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{pmatrix} \quad (19.5)$$

where (x_1, y_1, z_1) and (x_n, y_n, z_n) are the coordinates of the first and last nodes of the element, respectively.

Note that it is not necessary to use the centre O of the reference frame in (19.4). Tests show that it is possible to use any nodes of the element, or a linear combination of these nodes such as the centre of the element. This result has not been established analytically. For any point P meeting this criterion, for which the coordinates are (x_P, y_P, z_P) and (ξ_P, η_P, ζ_P) in the real coordinate system and the reference element coordinate system, respectively, equation (19.4) can be written again:

$$\begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} = \begin{pmatrix} \xi_P \\ \eta_P \\ \zeta_P \end{pmatrix} + {}^tJ^{-1}(\xi) \left\{ \begin{pmatrix} x \\ y \\ z \end{pmatrix} - \begin{pmatrix} x_P \\ y_P \\ z_P \end{pmatrix} \right\} \quad (19.6)$$

The previous remark makes sense in practice, because it takes more time to calculate the centre of the item rather than use a known point, *e.g.*, the first node⁵.

This method is easy to use because this Jacobian matrix J is defined in any finite element code using the reference element. It only requires, whatever the type of element, 3×3 matrix inversion, which can be done analytically, *i.e.*, without requiring the use of external libraries such as LAPACK.

We will now show, on a case-by-case basis, under what conditions this method is equivalent or otherwise to the nodal function method.

It is very important to note that this localisation must be performed **after orientation of the elements**.

19.1.3 Barycentric coordinate method

This method is based on the principle of conservation of the barycentric coordinates of any point in an element. These coordinates are the same in the reference element coordinate system and that specific to the real element.

⁴On Rubinacci's cube with prisms or hexahedra, the convergence is ensured after a maximum of 2 steps.

⁵It is the first node that is used in code_Carmel

Any point in an element can be uniquely defined by its barycentric coordinates⁶ λ_i . There are as many as there are nodes n , but only 3 of them are independent, of course. Each barycentric coordinate is defined by the relative volume of the tetrahedron defined by the point sought and 3 of the nodes of the element, *e.g.*, $\lambda_1 = V_1/V$ where V_1 is the volume of the tetrahedron defined by the point and nodes 2, 3 and 4 of the element and V is the volume of the element [Dhatt, Thouzot 1984, Sec. 2.5]. The sum of these barycentric coordinates is 1 because the volume of the element is the sum of the n volumes of the tetrahedra concerned by the point [Dhatt, Thouzot 1984, Sec. 2.5]. Hence a first dependency between the barycentric coordinates.

The calculation, which we will only detail by example, consists first in finding the barycentric coordinates from the real coordinates (x, y, z) of the point. By inverting a matrix $n \times n$. This matrix can depend on (x, y, z) and find the barycentric coordinates then become a non-linear problem. The coordinates (ξ, η, ζ) of the point sought are then obtained by a matrix-vector product of a matrix $n \times n$ and the vector formed by the barycentric coordinates.

This method is implemented in Code_Carmel3D and code_spectral, by N. Béreux (EDF R&D) for tetrahedra and D. Laval (EDF R&D) for the other elements⁷.

19.2 Tetrahedra

Figure 19.1 shows the reference tetrahedron with 4 nodes, of coordinates $(0,0,0)$, $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$ in the numbering order 1, 2, 3, and 4 of the nodes [Dhatt, Thouzot 1984, Sec. 2.5]. Any point in the reference tetrahedron must satisfy the inequalities: $\xi \geq 0$, $\eta \geq 0$, $\zeta \geq 0$ and $1 - \xi - \eta - \zeta \geq 0$. This definition is the same in code_Carmel, Code_Carmel3D and code_Carmel spectral.

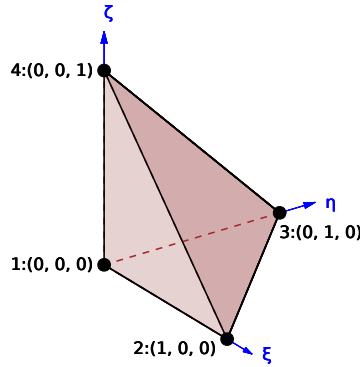


Figure 19.1: Reference tetrahedron.

⁶These barycentric coordinates are denoted L_i in [Dhatt, Thouzot 1984] and λ_i in [Bereux 2008]. We will use the latter notation in the rest of the text.

⁷Only the implementation for the prisms works correctly, albeit after correction of a bug identified on the code-carmel website.

19.2.1 Nodal function method

The 4 interpolation functions are: $N_1(\xi, \eta, \zeta) = 1 - \xi - \eta - \zeta$, $N_2(\xi, \eta, \zeta) = \xi$, $N_3(\xi, \eta, \zeta) = \eta$ and $N_4(\xi, \eta, \zeta) = \zeta$ [Dhatt, Thouzot 1984, Sec. 2.5.2]. The relation (19.1) it thus written:

$$\begin{aligned} x &= N_1x_1 + N_2x_2 + N_3x_3 + N_4x_4 \\ &= (1 - \xi - \eta - \zeta)x_1 + \xi x_2 + \eta x_3 + \zeta x_4 \\ &= x_1 + (x_2 - x_1)\xi + (x_3 - x_1)\eta + (x_4 - x_1)\zeta \end{aligned}$$

The same goes for the relations for y and z . The whole can be written in the following matrix form:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} + \begin{pmatrix} x_2 - x_1 & x_3 - x_1 & x_4 - x_1 \\ y_2 - y_1 & y_3 - y_1 & y_4 - y_1 \\ z_2 - z_1 & z_3 - z_1 & z_4 - z_1 \end{pmatrix} \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} \quad (19.7)$$

the calculation of the coordinates (ξ, η, ζ) of the point sought from its coordinates (x, y, z) is not difficult in this precise case, because the matrix multiplying the unknown does not depend on this unknown.

19.2.2 Jacobian matrix method

The Jacobian matrix J is written, from (19.5) and the 4 interpolation functions above [Dhatt, Thouzot 1984, Sec. 2.5.2.] :

$$J = \begin{pmatrix} -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_4 & y_4 & z_4 \end{pmatrix} = \begin{pmatrix} x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \\ x_4 - x_1 & y_4 - y_1 & z_4 - z_1 \end{pmatrix} \quad (19.8)$$

and the expression (19.3), where the centre of the element is the first node, brings us back to (19.7). This shows the equivalence of this reformulation in this specific case⁸.

This Jacobian matrix does not depend on the point sought, and the calculation of the coordinates (ξ, η, ζ) of the point sought from its coordinates (x, y, z) does not pose any difficulties with the aid of the relation (19.4).

19.2.3 Barycentric coordinates method

Since the tetrahedron has 4 nodes, there are 4 barycentric coordinates [Dhatt, Thouzot 1984, Sec. 2.5]. Tetrahedra are the simplest of all types of elements because their interpolation function is linear as a function of the coordinates. The search for coordinates in the reference element, does not pose any difficulty, therefore. First we have to find the 4 barycentric coordinates of the point in the real element, by resolving a 4x4 linear system involving the coordinates in the real element of the 4 nodes of the tetrahedron, *e.g.*, (x_1, y_1, z_1) for the first node (see Eq. 19.9).

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \\ 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \quad (19.9)$$

This is done in Code_Carmel3D using the DGESV routine (LU factorisation) from the LAPACK library. We then find the coordinates of the point (ξ, η, ζ) in the reference element by a

⁸I think this reformulation with the Jacobian matrix is true for all elements of the 1st order, where interpolation functions involve only linear, bi-linear or tri-linear polynomials in ξ , η and ζ . Because the derivative of these interpolation functions returns the function in part when multiplied by the variable of differentiation This would not be the case with polynomials of higher order.

matrix-vector product involving the barycentric coordinates previously found and the coordinates of the 4 nodes in the reference element *e.g.*, (ξ_1, η_1, ζ_1) for the first node (see Eq. 19.10). For the tetrahedron, the coordinates of the nodes in the reference element are $(0,0,0)$, $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$ respectively.

$$\begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} = \begin{pmatrix} \xi_1 & \xi_2 & \xi_3 & \xi_4 \\ \eta_1 & \eta_2 & \eta_3 & \eta_4 \\ \zeta_1 & \zeta_2 & \zeta_3 & \zeta_4 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{pmatrix} \quad (19.10)$$

Equations (19.9) and (19.10) are taken from the note [Bereux 2008] (see Secs. 7.1.1 and 7.2), without justification or bibliographic reference. In this case, the system (19.10) is easy to resolve: we find $\xi = \lambda_2$, $\eta = \lambda_3$, $\zeta = \lambda_4$ and, by definition (the sum of these barycentric coordinates is 1 because the volume of the element is the sum of the 4 volumes of the tetrahedra involving the point), $\lambda_1 = 1 - \lambda_2 - \lambda_3 - \lambda_4 = 1 - \xi - \eta - \zeta$ [Dhatt, Thouzot 1984, Sec. 2.5].

19.2.4 Proof of the equivalence of the last two methods

Here we show how to return to the Jacobian matrix method from the barycentric coordinate method.

By expressing the 4 barycentric coordinates $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ as a function of the coordinates (ξ, η, ζ) , the system (19.9) is written:

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \\ 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 - \xi - \eta - \zeta \\ \xi \\ \eta \\ \zeta \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \quad (19.11)$$

or, by developing,

$$\begin{aligned} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} &= \begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} + \begin{pmatrix} (x_2 - x_1)\xi + (x_3 - x_1)\eta + (x_4 - x_1)\zeta \\ (y_2 - y_1)\xi + (y_3 - y_1)\eta + (y_4 - y_1)\zeta \\ (z_2 - z_1)\xi + (z_3 - z_1)\eta + (z_4 - z_1)\zeta \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} + \begin{pmatrix} x_2 - x_1 & x_3 - x_1 & x_4 - x_1 \\ y_2 - y_1 & y_3 - y_1 & y_4 - y_1 \\ z_2 - z_1 & z_3 - z_1 & z_4 - z_1 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} \end{aligned} \quad (19.12)$$

which becomes, by deleting the last line that has become useless,

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} + \begin{pmatrix} x_2 - x_1 & x_3 - x_1 & x_4 - x_1 \\ y_2 - y_1 & y_3 - y_1 & y_4 - y_1 \\ z_2 - z_1 & z_3 - z_1 & z_4 - z_1 \end{pmatrix} \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} \quad (19.13)$$

which is indeed written in the form (19.3) with the transposed Jacobian defined by (19.8) :

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} + {}^t J \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} \quad (19.14)$$

where the centre of the reference element coordinate system is indeed the first node (x_1, y_1, z_1) .

19.3 Prisms

Figure 19.2 shows the reference prism with 6 nodes, of coordinates $(0,0,-1)$, $(1,0,-1)$, $(0,1,-1)$, $(0,0,1)$, $(1,0,1)$, and $(0,1,1)$ in the numbering order 1, 2, 3, 4, 5 and 6 of the nodes [Dhatt, Thouzot 1984, Sec. 2.7].

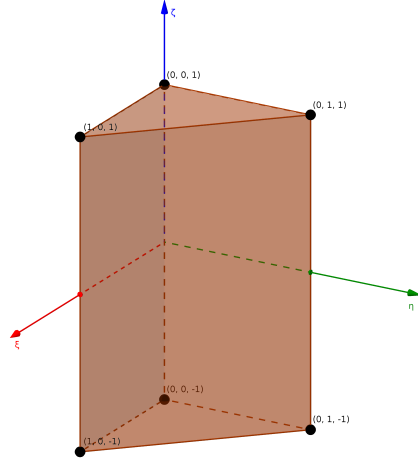


Figure 19.2: Reference prism.

Any point in the reference prism must satisfy the inequalities: $\xi \geq 0$, $\eta \geq 0$, $-1 \leq \zeta \leq 1$ and $1 - \xi - \eta \geq 0$. This definition is the same in code_Carmel, Code_Carmel3D and code_Carmel spectral.

19.3.1 Nodal function method

The 6 interpolation functions are: $N_1(\xi, \eta, \zeta) = \lambda a$, $N_2(\xi, \eta, \zeta) = \xi a$, $N_3(\xi, \eta, \zeta) = \eta a$, $N_4(\xi, \eta, \zeta) = \lambda b$, $N_5(\xi, \eta, \zeta) = \xi b$, $N_6(\xi, \eta, \zeta) = \eta b$ where $\lambda = 1 - \xi - \eta$, $a = (1 - \zeta)/2$ and $b = (1 + \zeta)/2$ [Dhatt, Thouzot 1984, Sec. 2.7.1]. They can also be written in the short form:

$$N(\xi, \eta, \zeta) \equiv (N_1(\xi, \eta, \zeta), N_2(\xi, \eta, \zeta), \dots, N_6(\xi, \eta, \zeta)) \quad (19.15)$$

$$= (\lambda a, \xi a, \eta a, \lambda b, \xi b, \eta b) \quad (19.16)$$

The relation (19.1) is written as follows:

$$\begin{aligned} x &= \sum_{i=1}^6 N_i x_i \\ &= \lambda a x_1 + \xi a x_2 + \eta a x_3 + \lambda b x_4 + \xi b x_5 + \eta b x_6 \\ &= a(\lambda x_1 + \xi x_2 + \eta x_3) + b(\lambda x_4 + \xi x_5 + \eta x_6) \\ &= \frac{1}{2} \{ (1 - \zeta) [(1 - \xi - \eta)x_1 + \xi x_2 + \eta x_3] + (1 + \zeta) [(1 - \xi - \eta)x_4 + \xi x_5 + \eta x_6] \} \\ &= \frac{1}{2} \{ x_1 + x_4 + \xi [(1 - \zeta)(x_2 - x_1) + (1 + \zeta)(x_5 - x_4)] + \eta [(1 - \zeta)(x_3 - x_1) + (1 + \zeta)(x_6 - x_4)] \\ &\quad + \zeta(x_4 - x_1) \} \end{aligned} \quad (19.17)$$

$$\begin{aligned} &= \frac{1}{2} \{ x_1 + x_4 + \xi(x_2 - x_1 + x_5 - x_4) + \eta(x_3 - x_1 + x_6 - x_4) \\ &\quad + \zeta[x_4 - x_1 + \xi(x_1 - x_2 + x_5 - x_4) + \eta(x_1 - x_3 + x_6 - x_4)] \} \end{aligned} \quad (19.18)$$

where there are two possible expressions, (19.17) and (19.18), to factorise (ξ, η, ζ) .

In practice the first expression (19.17) has poor non-linear convergence when the elements are not extruded, and we retain the second expression (19.18), inspired by the Jacobian matrix method, which has always given good results in our trials⁹. A similar relation exists for y and z . Equation (19.18) can be expressed in the following matrix form:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \frac{1}{2} \begin{pmatrix} x_1 + x_4 \\ y_1 + y_4 \\ z_1 + z_4 \end{pmatrix} + M \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} \quad (19.19)$$

where the translation is relative to the barycentre of nodes 1 and 4 and matrix M is:

$$M = \frac{1}{2} \begin{pmatrix} x_2 - x_1 + x_5 - x_4 & x_3 - x_1 + x_6 - x_4 & x_4 - x_1 + \xi(x_1 - x_2 + x_5 - x_4) + \eta(x_1 - x_3 + x_6 - x_4) \\ y_2 - y_1 + y_5 - y_4 & y_3 - y_1 + y_6 - y_4 & y_4 - y_1 + \xi(y_1 - y_2 + y_5 - y_4) + \eta(y_1 - y_3 + y_6 - y_4) \\ z_2 - z_1 + z_5 - z_4 & z_3 - z_1 + z_6 - z_4 & z_4 - z_1 + \xi(z_1 - z_2 + z_5 - z_4) + \eta(z_1 - z_3 + z_6 - z_4) \end{pmatrix} \quad (19.20)$$

To find the coordinates (ξ, η, ζ) from coordinates (x, y, z) , it is necessary to resolve the non-linear system :

$$\begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} = M(\vec{\xi})^{-1} \left\{ \begin{pmatrix} x \\ y \\ z \end{pmatrix} - \frac{1}{2} \begin{pmatrix} x_1 + x_4 \\ y_1 + y_4 \\ z_1 + z_4 \end{pmatrix} \right\} \quad (19.21)$$

where $\vec{\xi} = (\xi, \eta, \zeta)$ is the point sought.

One iterative method is substitution, which expresses the solution $\vec{\xi}_{n+1}$ at iterate $n+1$ as a function of the solution $\vec{\xi}_n$ at the previous iterate, from an initial approximation of the solution. Eq. 19.21 it thus written: $\vec{\xi}_{n+1} = M(\vec{\xi}_n)^{-1} \vec{b}$ where $\vec{b} = \vec{x} - (\vec{x}_1 + \vec{x}_4)/2$. The iteration continues until the residual of the equation $\|\vec{\xi}_{n+1} - \vec{\xi}_n\|$ is close enough to 0. This method is not difficult in practice, using the centre of the reference element $(1/3, 1/3, 0)$ for example as an initial point, when the mesh is extruded (maximum 2 iterations). The convergence is slower for a mesh that is not extruded (up to 30 iterations to reach a residual of 1e-12).

The non-linear Newton-Raphson method is defined below in order to possibly improve this convergence on the basis of knowledge of the first derivative of matrix M , analytic. It is recalled that the Newton-Raphson method seeks to find the solution x to the problem $f(x) = 0$ using an iterative method based on the development of $f(x)$ in the first order: $f(x) \simeq f(x_0) + f'(x_0)(x - x_0) = 0$, which here amounts to expressing the solution to iteration $n+1$: $x_{n+1} = x_n + f(x_n)/f'(x_n)$ from the solution to the previous iteration, from an initial estimate of the solution.

Our matrix problem (19.19) is thus written: $f(\vec{\xi}) = M(\vec{\xi})\vec{\xi} - \vec{b}$ not depending on $\vec{\xi} = (\xi, \eta, \zeta)$. The derivative of f is written thus $f'(\vec{\xi}) = M(\vec{\xi}) + \vec{\xi}dM/d\vec{\xi} = M + \xi dM/d\xi + \eta dM/d\eta + \zeta dM/d\zeta$, which gives:

$$\vec{\xi} \frac{dM}{d\vec{\xi}} = \frac{1}{2} \begin{pmatrix} 0 & 0 & \xi(x_1 - x_2 + x_5 - x_4) + \eta(x_1 - x_3 + x_6 - x_4) \\ 0 & 0 & \xi(y_1 - y_2 + y_5 - y_4) + \eta(y_1 - y_3 + y_6 - y_4) \\ 0 & 0 & \xi(z_1 - z_2 + z_5 - z_4) + \eta(z_1 - z_3 + z_6 - z_4) \end{pmatrix} \quad (19.22)$$

and the iterative linear system to resolve is written:

$$\begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix}_{n+1} = \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix}_n + dM(\vec{\xi}_n)^{-1} \left[\left\{ \begin{pmatrix} x \\ y \\ z \end{pmatrix} - \frac{1}{2} \begin{pmatrix} x_1 + x_4 \\ y_1 + y_4 \\ z_1 + z_4 \end{pmatrix} \right\} - M(\vec{\xi}_n) \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix}_n \right] \quad (19.23)$$

⁹Tests on Rubinacci's cube composed of pure prisms of which one is deformed, i.e., with one of its nodes moved along the 3 directions of space.

where the derivative matrix dM , to be inverted, is written, from (19.20) and (19.22):

$$dM = \frac{1}{2} \begin{pmatrix} x_2 - x_1 + x_5 - x_4 & x_3 - x_1 + x_6 - x_4 & x_4 - x_1 + 2\xi(x_1 - x_2 + x_5 - x_4) + 2\eta(x_1 - x_3 + x_6 - x_4) \\ y_2 - y_1 + y_5 - y_4 & y_3 - y_1 + y_6 - y_4 & y_4 - y_1 + 2\xi(y_1 - y_2 + y_5 - y_4) + 2\eta(y_1 - y_3 + y_6 - y_4) \\ z_2 - z_1 + z_5 - z_4 & z_3 - z_1 + z_6 - z_4 & z_4 - z_1 + 2\xi(z_1 - z_2 + z_5 - z_4) + 2\eta(z_1 - z_3 + z_6 - z_4) \end{pmatrix} \quad (19.24)$$

In practice, the Newton-Raphson method provides no improvement on an extruded mesh, and it even converges more slowly most of the time than the substitution method on a non-extruded mesh¹⁰.

Finally, it is easy to see why the convergence is also good for an extruded mesh. There is thus a relation between the node coordinates of an element that makes this problem linear. For example, for prisms orientated along the Oz axis, we have $x_5 = x_2$, $x_4 = x_1$ and $x_6 = x_3$ which allows us to write $x_1 - x_2 + x_5 - x_4 = 0$ and $x_1 - x_3 + x_6 - x_4 = 0$. The same goes for the relation in y for the same reasons. This is also valid for the relation in z because distance Δz between the two triangular faces of the prisms is the same at any point, *i.e.* for all its nodes. Thus $z_5 - z_2 = z_6 - z_3 = z_4 - z_1 = \Delta z$, which allows us to write $z_1 - z_2 + z_5 - z_4 = 0$ and $z_1 - z_3 + z_6 - z_4 = 0$. In the end, matrix M only depends on the coordinates of the nodes and no longer the unknowns ξ or ζ . It is written:

$$M = \frac{1}{2} \begin{pmatrix} x_2 - x_1 + x_5 - x_4 & x_3 - x_1 + x_6 - x_4 & x_4 - x_1 \\ y_2 - y_1 + y_5 - y_4 & y_3 - y_1 + y_6 - y_4 & y_4 - y_1 \\ z_2 - z_1 + z_5 - z_4 & z_3 - z_1 + z_6 - z_4 & z_4 - z_1 \end{pmatrix} \quad (19.25)$$

The Newton-Raphson method is then equivalent, analytically, to the substitution method, but adds a possible rounding error.

19.3.2 Jacobian matrix method

The Jacobian matrix is written:

$$\begin{aligned} J &= \begin{pmatrix} \frac{\partial N_1}{\partial \xi} & \frac{\partial N_2}{\partial \xi} & \dots & \frac{\partial N_6}{\partial \xi} \\ \frac{\partial N_1}{\partial \eta} & \frac{\partial N_2}{\partial \eta} & \dots & \frac{\partial N_6}{\partial \eta} \\ \frac{\partial N_1}{\partial \zeta} & \frac{\partial N_2}{\partial \zeta} & \dots & \frac{\partial N_6}{\partial \zeta} \end{pmatrix} \times \begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_6 & y_6 & z_6 \end{pmatrix} \\ &= \begin{pmatrix} -a & a & 0 & -b & b & 0 \\ -a & 0 & a & -b & 0 & b \\ -\lambda/2 & -\xi/2 & -\eta/2 & \lambda/2 & \xi/2 & \eta/2 \end{pmatrix} \times \begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_4 & y_4 & z_4 \\ x_5 & y_5 & z_5 \\ x_6 & y_6 & z_6 \end{pmatrix} \quad (19.26) \end{aligned}$$

where $\lambda = 1 - \xi - \eta$, $a = (1 - \zeta)/2$ and $b = (1 + \zeta)/2$ have been defined above (see Sec. 19.3.1).

We will detail below the expression of x , and show in which cases it is equivalent to the nodal function method. From (19.3) and (19.26), we can write $x = x_0 + J_{11}\xi + J_{21}\eta + J_{31}\zeta$ where $J_{11} = a(x_2 - x_1) + b(x_5 - x_4)$, $J_{21} = a(x_3 - x_1) + b(x_6 - x_4)$ and $J_{31} = \frac{1}{2} [\lambda(x_4 - x_1) + \xi(x_5 - x_2) + \eta(x_6 - x_3)]$. The whole is thus written:

$$\begin{aligned} x &= x_0 \\ &+ \xi [(1 - \zeta)(x_2 - x_1) + (1 + \zeta)(x_5 - x_4)] \\ &+ \eta [(1 - \zeta)(x_3 - x_1) + (1 + \zeta)(x_6 - x_4)] \\ &+ \zeta [x_4 - x_1 + \xi(x_1 - x_2 + x_5 - x_4) + \eta(x_1 - x_3 + x_6 - x_4)] \end{aligned} \quad (19.27)$$

¹⁰Tested on Rubinacci's cube with 500 extruded prisms along the axis Ox, with one node moved along Oz to create two non-extruded elements.

The same for y and z . We see that (19.27) is equivalent to (19.17 if and only if:

1. The translation point x_0 is indeed the centre of the reference element $(x_1 + x_4)/2$,
2. The dependence in ξ and η of the last line is zero, *i.e.*, $x_1 - x_2 + x_5 - x_4 = 0$ and $x_1 - x_3 + x_6 - x_4 = 0$.

These two conditions must also be met by the other components y and z .

Concerning condition 2) on cancellation, it is possible for any extruded mesh because there is a relation between the coordinates of the nodes (see Sec. 19.3.1). This is still true if the mesh is rotated in any way with respect to $Oxyz$ ¹¹, because this rotation maintains the relations below by simply mixing the relations in x , y and z . This is no longer true, however, if an element is deformed. In this case the Jacobian matrix method does not give good results compared with the nodal functions method¹².

19.3.3 Barycentric coordinates method

This method was programmed by Damien Laval (EDF R&D). The code does not contain comments and largely reproduces the notation introduced for tetrahedra (see Sec. 19.2.3). Algorithm 19.1 is decoded below, using the notation in the preceding sections. We solve the matrix problem (size 4x4):

$$A \begin{pmatrix} \xi \\ \eta \\ \zeta \\ \mu \end{pmatrix} = b = 2 \left\{ \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} - \frac{1}{2} \begin{pmatrix} x_1 + x_4 \\ y_1 + y_4 \\ z_1 + z_4 \\ 1 \end{pmatrix} \right\} \quad (19.28)$$

where the 4x4 matrix: $A = A_{LIN} + A_{NLIN}$ is decomposed into its linear and non-linear parts:

$$A_{LIN} = \begin{pmatrix} x_2 - x_1 + x_5 - x_4 & x_3 - x_1 + x_6 - x_4 & x_4 - x_1 & 1 \\ y_2 - y_1 + y_5 - y_4 & y_3 - y_1 + y_6 - y_4 & y_4 - y_1 & 1 \\ z_2 - z_1 + z_5 - z_4 & z_3 - z_1 + z_6 - z_4 & z_4 - z_1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (19.29)$$

and

$$A_{NLIN} = \zeta \begin{pmatrix} x_1 - x_2 + x_5 - x_4 & x_1 - x_3 + x_6 - x_4 & 0 & 0 \\ y_1 - y_2 + y_5 - y_4 & y_1 - y_3 + y_6 - y_4 & 0 & 0 \\ z_1 - z_2 + z_5 - z_4 & z_1 - z_3 + z_6 - z_4 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad (19.30)$$

The resolution of the system (19.28), non-linear in ζ , is performed iteratively, from the initial value $(\xi, \eta, \zeta, \mu) = (1/4, 1/4, 1/4, 1/4)$, until the residual of this system is close to 0. The 4th unknown μ is not really unknown because the system (19.28) leads to the solution $\mu = 1$. The algorithm notation is : ALIN for A_{LIN} , ANLIN for A_{NLIN} , lambda_tot(1:4) for (ξ, η, ζ, μ) . The equivalence between λ_{tot} and the coordinates of the point in the reference element is not obvious. It is proved below. Algorithm 19.1 expresses these coordinates according to the coordinates of the nodes of the prism in the reference element: xi(i), eta(i) and zeta(i), and a lambda vector of size 6. These relations are written:

¹¹We tested it on Rubinacci's cube turned by 45° relative to the Ox axis.

¹²We tested it by moving a node in the 3 directions of space, then making a section line along the deformed edge. The nodal function method finds the coordinates $\xi = 0$, $\eta = 1$ and $\zeta \in [-1, 1]$ as it should. The Jacobian matrix method finds $\zeta \in [-0.666..., 0.666]$. The field path is much better for the nodal functions method.

$$\begin{aligned}
\xi &= \sum_{i=1}^6 \xi_i \lambda_i = \lambda_2 + \lambda_5 = \lambda_{tot}(1) \\
\eta &= \sum_{i=1}^6 \eta_i \lambda_i = \lambda_3 + \lambda_6 = \lambda_{tot}(2) \\
\zeta &= \sum_{i=1}^6 \zeta_i \lambda_i = \lambda_4 + \lambda_5 + \lambda_6 - (\lambda_1 + \lambda_2 + \lambda_3) = \lambda_{tot}(3)
\end{aligned} \tag{19.31}$$

using the known coordinates ξ_i , η_i and ζ_i of the nodes in the reference element (see Sec. 19.3). All this is equivalent to the nodal function method using the first expression (19.17), which causes convergence problems when the mesh is not not extruded, as we found. Although the algorithm 19.1 is in a form appropriate to the non-linear Newton-Raphson method, the inverted matrix is not the correct one and the method used is substitution. Probably with rounding errors coming from this shape and perhaps from the unnecessary fourth dimension.

Algorithm 19.1 Barycentric coordinates method programmed for prisms.

```

ALIN(1:4,1:4) = 0
ALIN(1,1) = - x1 + x2 - x4 + x5
... the same holds for ALIN(2,1) and ALIN(3,1) by replacing x with y and z, respectively
ALIN(1,2) = - x1 + x3 - x4 + x6
... the same holds for ALIN(2,2) and ALIN(3,2) by replacing x with y and z, respectively
ALIN(1,3) = - x1 + x4
... the same holds for ALIN(2,3) and ALIN(3,3) by replacing x with y and z, respectively
ALIN(4,:) = 1
b(1) = 2 * x - ( x1 + x4 )
... the same holds for b(2) and b(3) by replacing x with y and z, respectively
b(4) = 1
lambda_tot(1:4) = 0.25
ANLIN(1:4,1:4) = 0
ANLIN(1,1) = lambda_tot(3) * ( x1 - x2 - x4 + x5 )
... the same holds for ANLIN(2,1) and ALIN(3,1) by replacing x with y and z, respectively
ANLIN(1,2) = lambda_tot(3) * ( x1 - x3 - x4 + x6 )
... the same holds for ANLIN(2,2) and ALIN(3,2) by replacing x with y and z, respectively
ANLIN = ANLIN + ALIN
res(1:4) = b(1:4) - ANLIN(1:4,1:4) * lambda_tot(1:4)
err = 1
iter = 0
while iter < 50 et err > 1e-12 do
    lambda2(1:4) = res(1:4)
    inversion of ANLIN by subroutine LAPACK DGESV -> lambda2 = inverse(ANLIN)*lambda2
    lambda_tot(1:4) = lambda_tot(1:4) + lambda2(1:4)
    updating of ANLIN with the new value of lambda_tot
    ANLIN = ANLIN + ALIN
    updating of res and err
end while
lambda(1) = ( 1 - lambda_tot(1) - lambda_tot(2) ) * ( 1 - lambda_tot(3) ) / 2
lambda(2) = lambda_tot(1) * ( 1 - lambda_tot(3) ) / 2
lambda(3) = lambda_tot(2) * ( 1 - lambda_tot(3) ) / 2
lambda(4) = ( 1 - lambda_tot(1) - lambda_tot(2) ) * ( 1 + lambda_tot(3) ) / 2
lambda(5) = lambda_tot(1) * ( 1 + lambda_tot(3) ) / 2
lambda(6) = lambda_tot(2) * ( 1 + lambda_tot(3) ) / 2
xi = sum on i from 1 to 6 of xi(i) * lambda(i)
eta = sum on i from 1 to 6 of eta(i) * lambda(i)
zeta = sum on i from 1 to 6 of zeta(i) * lambda(i)

```

Bibliography

- [Albanese, Rubinacci 2000] A. Albanese, G. Rubinacci *Magnetostatic field computations in terms of two component vector potentials* Int. J. Numer. Meth. Engng., Vol. 49, pp 573-598, 2000.
- [Alotto, Perugia 2004] P. Alotto, I. Perugia. *Matrix properties of a vector potential cell method for magnetostatics*. IEEE Trans. Mag, Vol. 40, No 3, 2004.
- [Ammari et al 2000] Habib Ammari, Annalisa Buffa, Jean-Claude Nédélec *A justification of eddy currents models for the Maxwell's equations* SIAM J. Appl. Math., 60(5), pp. 1805-1823, May 2000.
- [Antunes et al 2005] O. J. Antunes, J. P. A. Bastos, N. Sadowski, A. Razek, L. Santandrea, F. Bouillault, F. Rapetti, *Using hierarchic interpolation with mortar element method for electrical machines*, IEEE Trans. Magn., vol. 41, n°5, pp 1472-1475, 2005.
- [Antunes et al 2006] O.J. Antunes, J. P. A. Bastos, N. Sadowski, A. Razek, L. Santandrea, F. Bouillault and F. Rapetti, *Comparison between non-conforming movement methods*, IEEE Transactions on Magnetism, vol. 42, n°4, pp 599-x, 2006.
- [Badics, Cendes 2007] Zsolt Badics and Zoltan J. Cendes, *Source Field Modeling by Mesh Incidence Matrices*, Ansoft Corporation, Pittsburgh, 2007
- [Bastos, Sadowski 2003] J. P. A. Bastos and N. Sadowski *Electromagnetic Modeling by Finite Element Methods* 1st ed. CRC Press, 2003.
- [Beddek 2012] K. Beddek *Propagation d'incertitudes dans les modèles éléments finis en électromagnétisme – Application au contrôle non destructif par courants de Foucault* PhD thesis, Université des Sciences et Technologie de Lille, 2012
- [Bereux 2008] N. Béreux, *code_Carmel3D : implementation of version 1.0*, EDF R&D, technical note H-R25-2008-03705-FR, 2008.
- [Bertotti 1988] G. Bertotti *General properties of power losses in soft ferromagnetic materials* IEEE Transactions on Magnetism, vol. 24, no. 1, pp. 621–630, Jan. 1988.
- [Bertotti 1998] G. Bertotti *Hysteresis in magnetism: for physicists, materials scientists, and engineers*, Gulf Professional Publishing, 1998.
- [Biddlecombe et al 1988] Biddlecombe, C.S.; Simkin, J.; Jay, A.P.; Sykulski, J.K.; Lepaul, S. *Transient electromagnetic analysis coupled to electric circuits and motion*, IEEE Trans. Magn., vol. 34, Issue 5, Part 1, 3182 - 3185, 1998.

- [Biro et al 1993b] O. Biro, K. Preis, G. Vrisk, K. R. Richter, I. Tîcar *Computation of 3D magnetostatic fields using a reduced scalar potential*, IEEE Trans. Mag., vol. 29, no. 2, pp. 1329-1332, 1993.
- [Biro et al 1993] O. Biro, K. Preis, W. Renhart, G. Vrisk, K. R. Richter, *Computation of a 3D current driven skin effect problems using a current vector potential*, IEEE Trans. Mag., vol. 29, pp 1325-1328, 1993.
- [Biro, Preis 2000] O. Biro, K. Preis. *An edge finite element eddy current formulation using magnetic and a current vector potential*. IEEE Trans. Mag., vol. 36, No 5, pp. 3128-3130, 2000.
- [Boiteau 2014] O. Boiteau, *Amélioration des fonctionnalités solveurs linéaire de Code_Carmel v1.12.0: manuel théorique, manuel utilisateur et descriptif informatique*, RDF R&D, technical note H-I23-2014-00544-FR, date to be confirmed.
- [Bossavit, Vêrité 1983] A. Bossavit, J.-C. Vêrité *The Trifou code: solving the 3D eddy current problem by using h as a state variable* I.E.E.E. Trans. on Magnetism, Vol. 19, no 6, pp 2465-2471, 1983
- [Bossavit 1993] A. Bossavit. *Électromagnétisme en vue de la modélisation* Édition Springer-Verlag, 1993.
- [Bossavit 2003] A. Bossavit, *Mixed-hybrid methods in magnetostatics: complementarity in one stroke*, IEEE Trans. Mag, Vol. 39, No 3, pp 1099-1102, 2003.
- [Bossavit, Kettunen 2000] A. Bossavit, L. Kettunen. *Yee-like schemes on staggered cellular grids: a synthesis between FIT and FEM approaches*. IEEE Trans. Mag, Vol. 36, No 4, pp 861-867, 2000.
- [Boualem 1997] B. Boualem *Contribution à la modélisation des systèmes électrotechniques à l'aide des formulations en potentiels : application à la machine asynchrone* PhD thesis, Université des sciences et technologies de Lille, 1997.
- [Boualem, Piriou 1996] B. Boualem, F. Piriou *On the use of potentials to study of 3D magnetostatic problems*, International Workshop on electric and magnetic fields, Liège 1996.
- [Boualem, Piriou 1998] B. Boualem, F. Piriou *Numerical models for rotor cage induction machines using finite element method* IEEE Trans. Magn. vol. 34, n° 5, pp 3202-3205, 1998.
- [Boualem, Piriou 1998b] B. Boualem, F. Piriou *Modélisation 3D du circuit électrique et du mouvement: application à la machine asynchrone* European Physical Journal Applied Physics, vol. 1 n°1, pp 67-71, 1998.
- [Boukari 2000] N. Boukari *Modélisation du mouvement à l'aide de codes de calcul par éléments finis en 3D : application à la machine homo polaire et au micro actionneur électrostatique*, PhD thesis, Institut National Polytechnique, Toulouse, July 2000.
- [Bouissou 1994] S. Bouissou *Comparaison des formulations en potentiel, pour la résolution numérique en 3D des équations magnétiques couplées aux équations du circuit électrique*, PhD thesis, Université de Paris VI, July 1994.

- [Brissonneau 1997] P. Brissonneau *Magnétisme et matériaux magnétiques pour l'électrotechnique*, Hermes Sciences Publicat., 1997
- [Cahouet 1992] J. Cahouet *Modélisations simplifiées des champs électromagnétiques basses et moyennes fréquences en présence d'un être humain*, EDF R&D, technical note HI72/7705, August 1992.
- [Chaitin-Chatelin et Frayssé 1996] F. Chaitin-Chatelin, V. Frayssé *Lectures on Finite Precision Computations*, Society for Industrial and Applied Mathematics, 1996.
- [Chavanne 1988] J. Chavanne *Contribution à la modélisation des systèmes statiques à aimants permanents*, PhD thesis, Institut National Polytechnique de Grenoble, 1988.
- [Clemens, Weiland 1987] M. Clemens, T. Weiland. *Discrete electromagnetics: Maxwell's equations tailored to numerical simulations*. Compumag 1987, Graz, Austria, pp 13-20, 1987.
- [Costabel] Costabel M, Dauge M *Espaces fonctionnels Maxwell : Les gentils, les méchants et les singularités*", <http://perso.univrennes1.fr/monique.dauge/publis/CoDaZmax.pdf>
- [Coulomb 1983] J. - L. Coulomb *A methodology for the determination of global electromechanical quantities from the finite element analysis and its application to the evaluation of magnetic forces, torques and stiffness*, I.E.E.E. Trans. on Magnetics, vol. 19, no. 6, pp. 2514-2519, 1983
- [Coulomb, Meunier 1984] J. - L. Coulomb, G. Meunier *Finite Element Implementation of virtual work principle for magnetic or electric force and torque computation*, I.E.E.E. Trans. on Magnetics, vol. 20, no. 5, pp. 1894-1896, 1984
- [Daveau, Rioux-Damidaou 1999] C. Daveau, F. Rioux-Damidaou, *New (e,h) formulation coupling a finite element method and a boundary integral method for the computation of the interaction of waves with a conducting domain*, IEEE Trans. Mag, Vol. 35, No 2, pp 1014-1018, 1999.
- [Deliège 2003] G. Deliège *Flexible implementation of the finite element method applied to 3D coupled problems considering convective effects*, PhD thesis, Katholieke Universiteit Leuven, December 2003.
- [Dembo,Steihaug1983] R. S. Dembo, T. Steihaug *Truncated-Newton algorithms for large-scale unconstrained optimization*, Math. Programming, 26(2): pp.190-212, 1983
- [Demenko et al 2006] A. Demenko, K. Hameyer, L. Nowak, K. Zawirski, X. Shi, Y. Le Menach, J.-P. Ducreux, F. Piriou *Comparison of slip surface and moving band techniques for modelling movement in 3D with FEM COMPEL-The international journal for computation and mathematics in electrical and electronic engineering*, Emerald Group Publishing Limited, Vol. 25, No 1, pp17-30, 2006.
- [Dhatt, Thouzot 1984] G. Dhatt, G. Thouzot, *Une présentation de la méthode des éléments finis*, Coll. Université de Compiègne, Ed. Maloine S.A., Paris, 2nd ed., 1984.

- [Dreher et al 1996] T. Dreher, R. Perrin-Bit, G. Meunier and J. L. Coulomb *A three dimensional finite element modelling of rotating machines involving movement and external circuit*, IEEE Trans. Magn. vol. 32, n° 3, pp1070-1073, 1996.
- [Dular 1994] P. Dular. *Modélisation du champ magnétique et des courants induits dans des systèmes tridimensionnels non linéaires*, PhD thesis, Université de Liège - Faculté des Sciences Appliquées, 1994.
- [Dular et al 1996] P. Dular, F. Robert, J.F. Remacle, M. Umé, W. Legros *Computation of the source current density in inductors of any shape using a mixed formulation*, Third International Workshop on Electric and Magnetic Fields, pp. 107-112, Liège 1996.
- [Dular, Legos 1998] P. Dular, W. Legros *Coupling of local and global quantities in various finite element formulations and its application to electrostatics, magnetostatics and magnetodynamics* IEEE Trans. Mag, Vol. 34, No 5, pp. 3078-3081, 1998.
- [Durand 1968] E. Durand, *Magnétostatique*. Edition Masson et Cie, 1968.
- [Enokizono et al 1990] M. Enokizono, T. Suzuki, J. Sievert, and J. Xu *Rotational power loss of silicon steel sheet* IEEE Transactions on Magnetics, vol. 26, no. 5, pp. 2562-2564, Sep. 1990.
- [Féliachi 1981] M. Féliachi *Contribution au calcul du champ électromagnétique par la méthode des éléments finis en vue d'une modélisation dynamique de machines électriques*, Engineering PhD thesis, LGEP, 1981.
- [Fiorillo, Novikov 1990] F. Fiorillo and A. Novikov *An improved approach to power losses in magnetic laminations under nonsinusoidal induction waveform* IEEE Transactions on Magnetics, vol. 26, no. 5, pp. 2904-2910, Sep. 1990.
- [Fournet 1985] F. Fournet, *Électromagnétisme à partir des équations locales*. Édition Masson, 1985.
- [Fujiwara et al 1993] K. Fujiwara, T. Nakata, H. Fusayasu. *Acceleration of convergence characteristic of the ICCG method* IEEE Trans. Mag., vol. 29, No 2, pp. 1958-1961, 1993.
- [Gasmi 1996] N. Gasmi *Contribution à la modélisation des phénomènes électriques- magnétiques couplés et du mouvement, pour les systèmes électromagnétiques en 3D*, PhD thesis, Université Paris-VI, October 1996.
- [Geuzaine 2001] C. Geuzaine *High order hybrid finite element schemes for Maxwell's equations taking thin structures and global quantities into account*, PhD thesis, Université de Liège, October 2001.
- [Girault 2006] V. Girault *Aproximations variationnelles des EDP*, Cours de DEA, 2005-2006.
- [Goby 1987] F. Goby *Utilisation d'une méthode couplée: élément finis - élément de frontière, pour le calcul des forces dans des dispositifs électromagnétiques. Application au calcul du couple d'une machine à réluctance variable*, PhD thesis, Université Paris-VI, LGEP, September 1987.

- [Golias, Tsiboukis 1994] N.A.Golias, T.D. Tsiboukis *Magnetostatics with edge element: a numerical investigation in the choice of the tree*, IEEE Trans. Mag., vol. 30, no .5, pp. 2877-2880, 1994.
- [Golovanov 1997] C. Golovanov *Développement de formulations éléments finis 3D en potentiel vecteur magnétique : application à la simulation de dispositifs électromagnétiques en mouvement*, PhD Thesis, INPG, 1997.
- [Golovanov et al 1998] C. Golovanov, Y. Marechal, G. Meunier *3D edge element based formulation coupled to electric circuits* IEEE Trans. Mag., vol. 34, No 5, pp. 3162-3165, 1998.
- [Gondran, Minoux 1995] M. Gondran, M. Minoux *Graphes et algorithmes*, Collection des études et recherches de EDF, Editions Eyrolles, 1995
- [Goursaud 2015] B. Goursaud *Note de principe de Code_Carmel3D version 2.5.0*, EDF R&D, technical H-R26-2015-05615-FR, November 2015
- [Gradinaru 1999] V Gradinaru and R Hiptmair *Whitney elements on pyramids*, Electronic Transactions on Numerical Analysis, 8 :pp. 154–168, 1999.
- [Henneberger, Hadrys 1993] G. Henneberger, W. Hadrys, *On the nature of different approaches for the calculation of magnetic forces and their mechanical displacement*, Compumag, pp.2-3, 1993.
- [Henneron 2004] T. Henneron *Contribution à la prise en compte des Grandeurs Globales dans les Problèmes d'Électromagnétisme résolu avec la Méthode des Éléments Finis*, PhD thesis, Université des Sciences et Technologies de Lille, defended on 15 December 2004.
- [Henneron et al 2005] T. Henneron, S. Clenet, P. Dular, F. Piriou *Discrete Finite Element characterisations of source fields for volume boundary constraints in electromagnetic problems* ACOMEN 2005.
- [Higham 2002] N. J. Higham *Accuracy and Stability of Numerical Algorithms* Society for Industrial and Applied Mathematics, second edition, 2002
- [Johnson 1987] C. Johnson *Numerical solution of partial differential equations by the finite element method* Cambridge University Press, Cambridge, 1987.
- [Kameari, Koganezawa 1997] A.Kameari, K.Koganezawa *Convergence of ICCG method in FEM using edge elements without gauge condition*, IEEE Trans. Mag., vol 33, pp. 1223-1226, 1997.
- [Kawase et al 1995] Y. Kawase, T. Yamaguchi and Y. Hayashi *Analysis of Cogging Torque of Permanent Magnet Motor by 3-D finite Element Method* IEEE Trans. Magn. vol. 31, n°3, pp 2044-2047, 1995.
- [Kawase et al 1998] Y. Kawase, T. Mori, T. Ota. *Magnetic field analysis of coupling transformers for electric vehicle using 3-D finite element method*. IEEE Trans. Mag., vol. 34, No 5, pp. 3186-3189, 1998.
- [Kelley 2003] C. T. Kelley *Solving nonlinear equations with Newton's method. Fundamentals of Algorithms*, SIAM, 2003.

- [Kettunen et al 1999] L. Kettunen, K. Forsman, A. Bossavit *Gauging in Whitney spaces* IEEE Trans. Mag., vol. 35, No 3, pp 1466-1469, 1999.
- [Kladas, Tegopoulos 1992] A.G. Kladas and J.A. Tegopoulos *A new potential formulation for 3D magnetostatic necessitating no source field computation*, IEEE Trans. Mag., vol 28, no 2, pp. 1103-1106, 1992.
- [Korecki 2009] J. Korecki *Contribution à la modélisation 3D des systèmes électromagnétiques basse fréquence à l'aide de la méthode d'intégration finie(FIT)* PhD thesis, Université des Sciences et Technologies de Lille, defended on 15 May 2009.
- [Kuczmam 2010] M. Kuczmam *Technique to Solve Nonlinear Static Magnetic Field Problems Using the Newton-Raphson Method in the Polarization*, IEEE Transactions on Magnetics, 46(3) : pp. 875–879 2010.
- [Le Floch 2002] Y. Le Floch. *Développement de formulations 3D éléments finis pour la prise en compte de conducteurs massifs et bobinés avec un couplage circuit*. PhD thesis, Institut National Polytechnique de Grenoble, 2002.
- [Le Menach et al 1998] Y.Le Menach, S.Clénet, F.Piriou, *Determination and utilization of the source field in 3D magnetostatic problems*, IEEE Trans. Mag., vol. 34, pp. 2509-2512, 1998.
- [Le Menach 1999] Y. Le Menach, *Contribution à la modélisation numérique tridimensionnelle des systèmes électrotechniques* PhD thesis, Université des Sciences et Technologies de Lille, defended on 1 February 1999.
- [Le Menach et al 2000] Y. Le Menach, S. Clénet, F. Piriou. *Numerical model to discretize source fields in the 3D finite element method*. IEEE Trans. Mag, Vol. 34, No 4, 2000.
- [Le Menach 2012] Y. Le Menach, *Contribution à la modélisation numérique des phénomènes électromagnétiques 3D en basse fréquence* Summary report with a view to obtaining Authorisation to Supervise Research, Université des Sciences et Technologies de Lille, defended on 11 December 2012.
- [Lepaul et al 1999] S. Lepaul, J. K. Sykulski, C. S. Biddlecombe, A. P. Jay, J. Simkin *Coupling of motion and circuits with electromagnetic analysis* IEEE Trans. Magn., vol., n°35, pp 1602-1605, 1999.
- [Maréchal 1991] Y. Maréchal *Modélisation des phénomènes magnéto-statiques avec terme de transport: Application aux ralentisseurs électromagnétiques*, PhD thesis, INPG, February 1991.
- [Marrocco 1977] A. Marrocco *Analyse numérique des problèmes en électrotechniques*, Ann. Sc. Math, Québec, vol. 1, pp. 271-296, 1977.
- [Marrone 2004] M. Marrone. *Properties of constitutive matrices for electrostatic and magnetostatic problems*. IEEE Trans. Mag., vol. 40, pp 1045-1048, 2004.
- [Mayergoyz 1983] I. D. Mayergoyz. *A new approach to the calculation of three-dimensional skin effect problems*. IEEE Trans. Mag., vol. 19, No 5, pp 2198-2200, 1983.

- [Miellou, Spiteri 1985] J. C. Miellou, P. Spiteri, *Un critère de convergence pour des méthodes générales de point fixe*, 1985.
- [Moreau 2012] O. Moreau *Note de principe de Code_Carmel3D*, EDF R&D, technical note H-R26-2011-02244-FR, April 2012
- [Montier 2018] L. Montier "Applications de méthodes de réduction de modèles aux problèmes d'électromagnétisme basse fréquence", Thesis, L2EP, 2018.
- [Moses 1992] A. Moses *Importance of rotational losses in rotating machines and transformers* Journal of Materials Engineering and Performance, vol. 1, no. 2, pp. 235–244, Mar. 1992.
- [Nakata et al 1988] T. Nakata, N. Takahashi, K. Fujiwara and Y. Okada *Improvement of $\mathbf{T}-\Omega$ method for 3D eddy currents analysis*, IEEE Trans. Mag., vol. 24, no 1, pp. 274-277, 1988.
- [Nakata et al, 1988] T. Nakata, N. Takahashi, K. Fujiwara, Y. Okada. *A new potentiel formulation for 3D magnetostatic necessitating no field computation*. IEEE Trans. Mag., vol. 24, No 1, pp. 274-277, 1988.
- [Nakata et al 1995] T. Nakata, N. Takahashi, K. Fujiwara *Summary of results for TEAM Workshop problem 13 (3-D nonlinear magnetostatic model)* 14(2) : pp. 91–101, 1995.
- [Nédélec 1992] J.-C. Nédélec *Notion sur les techniques d'éléments finis*, Édition Springer, 1992
- [Pérez et al 1990] J-P. Pérez, R. Carles, R. Fleckinger. *Électromagnétisme*, Édition Masson, 1990.
- [Perrin-Bit 1992] R. Perrin-Bit, *Modélisation des machines électriques tournantes par la méthode des éléments finis tridimensionnels : calcul des grandeurs magnétiques avec prise en compte du mouvement*, PhD thesis, Institut National Polytechnique, Grenoble, March 1992.
- [Pierquin 2011] A. Pierquin, *Imposition d'un courant uniforme par section dans un conducteur quelconque sous code_Carmel_3D*, internship report, Master 2 Scientific Calculation - USTL, March 2011 - September 2011.
- [Preston et al 1988] T. W. Preston, A. B. J. Reece, P. S. Sangha *Induction motor analysis by time-stepping techniques* IEEE Trans. Magn. vol. 24, n°1, pp 471-474, 1988.
- [Rapetti 2000] F. Rapetti *Approximation des équations de la magnétodynamique en domaine tournant par la méthode des éléments avec joints*, PhD thesis, Université Pierre et Marie Curie, May 2000.
- [Rapetti et al 2000] F. Rapetti, F. Bouillault, L. Santendra, A. Buffa, Y. Maday, A. Razek *Calculation of Eddy currents with edge elements on non-matching grids in moving structures*, IEEE Trans. Magn., vol. 36, n°4, pp 1351-1355, 2000.
- [Rapetti, Rousseau 2011] F. Rapetti, G. Rousseaux *Implications of Galilean Electromagnetism in Numerical Modeling*, ACES Journal, Vol. 26, n°. 9, pp. 784-791, September 2011.

- [Razek et al 1982] A. Razek, J. L. Coulomb, M. Féliachi, J. C. Sabonnadière *Conception of an air gap element for dynamic analysis of the electro-magnetic field of electric machine*, IEEE Trans. Magn., vol. 18, n°2, pp 655-659, 1982.
- [Ren 1994] Z. Ren *Comparison of different force calculation methods in 3D finite element modelling*, I.E.E.E. Trans. on Magnetism, vol. 30, no.5, pp. 3471-3474, 1994.
- [Ren 1996] Z. Ren *Auto-gauging of vector potential by iterative solver numerical evidence*, International workshop on electric and magnetic fields, pp 119-124, Liège 1996
- [Ren 1996b] Z. Ren *Influence of R.H.S on the convergence behaviour of curl-curl equation*, IEEE Trans. Mag., vol 32, pp. 655-658, 1996.
- [Ren et al 1990] F. Bouillault, A. Razek, A. Bossavit, J-C. Vérité *A new hybrid model using electric field formulation for 3-D eddy current problems*, IEEE Trans. Mag, Vol. 26, No 2, pp 470-473, 1990.
- [Ren et al 1992] Z. Ren, M. Besbes, S. Boukhtache *Calculation of local magnetic forces in magnetized materials*, International Workshop on electric and magnetic fields, pp. 64.1-64.6, Liège 1992.
- [Ren, Razek 1992] Z. Ren, A. Razek *On the magnetic forces calculation by equivalent source method*, International Workshop on electric and magnetic fields, pp. 21.1-21.5, Liège 1992.
- [Ren, Razek 1992] Z. Ren, A. Razek *Local force computation in deformable bodies using edge elements*, I.E.E.E. Trans. on Magnetism, vol. 20, no.2, pp. 1212-1215, 1992
- [Ren, Razek 1994] Z. Ren, A. Razek *A strong coupled model for analysing dynamic behaviours of non linear electromechanical systems*, I.E.E.E. Trans. on Magnetism, vol. 30, no.5, pp. 3252-3255, 1994.
- [Rodger et al 1990] D. Rodger, H. C. Lai and P. J. Leonard *Coupled element for problems involving movement*, IEEE, Trans. Magn., vol 26, no2, pp548-550, March 1990.
- [Sadowski et al 1992] N. Sadowski, Y. Lefèvre, M. Lajoie-Mazenc, J.-P. A. Bastos *Sur le calcul des forces magnétiques*, Journal Physique III, France, pp. 859-870, 1992.
- [Sadowski 1993] N. Sadowski *Contribution à la modélisation des machines électriques par la résolution simultanée des équations du champ et des équations du circuit électrique d'alimentation*, PhD thesis, I.N.P. Toulouse, December 1993.
- [Swift et al, 2001] G. Swift, D. A. Tziouvaras, P. McLaren, G. Alexander, D. Dawson, J. Esztergalyos, C. Fromen, M. Glinkowski, I. Hasenwinkele, M. Kezunovic, L. Kojovic, B. Kotheimer, R. Kuffel, J. Nordstrom, S. Zocholl, *Discussion of Mathematical models for current, voltage, and coupling capacitor voltage transformers and closure*, IEEE Transactions on Power Delivery, Vol. 16, no 4, pp. 827-828, 2001.

- [Tarhasaari et al 1999] T. Tarhasaari, L. Kettunen, A. Bossavit. *Some realizations of a discrete Hodge operator: a reinterpretation of finite element techniques*. IEEE Trans. Mag, Vol. 35, No 3, pp 1494-1497, 1999.
- [Tittarelli 2016] R. Tittarelli *Estimateurs d'erreur a posteriori pour les équations de Maxwell en formulation temporelle et potentielle* PhD thesis, Université des Sciences et Technologies de Lille, defended on 27 September 2016.
- [Tonti 2000] E. Tonti. *Algebraic topology and computational electromagnetism*. In International Workshop on Electric and Magnetic Fields, pp 20-21, 2000.
- [Tonti 2001a] E. Tonti. *A discrete formulation of field laws: The cell method*. CMES, vol. 1, No 1, 2001.
- [Tonti 2001] E. Tonti. *Finite Formulation of Electromagnetic fields*. In ICS Newsletter, vol. 8, No 1 pp 5-12, 2001.
- [Tsukerman 1992] I. A. Tsukerman *Overlapping finite elements for problems with movement*, IEEE Trans. on Magn., vol. 28, n°5, pp 2247-2249, 1992.
- [Vassent 1990] E. Vassent *Contribution à la modélisation des moteurs asynchrones par la méthode des éléments finis*, PhD thesis, I.N.P.G Grenoble, November 1990.
- [Vérité et al 2007] Jean - Claude Vérité, Jean - Pierre Ducreux, Gérard Tanneau, Philippe Baraton, Bernard Paya *Calcul de champ électromagnétique*, Lavoisier, 2007.
- [Webb, Forghani 1989] J.P.Webb and B.Forghani *A single scalar potential method for 3D magnetostatics using edge elements*, IEEE Trans. Mag., vol 25, no 5, pp. 4126-4128, 1989.
- [Weiss 1907] P. Weiss *L'hypothèse du champ moléculaire et la propriété ferromagnétique* 1907.
- [Ypma 1995] T. J. Ypma *Historical Development of the Newton-Raphson Method* SIAM Review, 37(4) : pp. 531–551, 1995
- [Ren et al 1996] Z.Ren, "Influence of R.H.S on the convergence behaviour of curl-curl equation", in *IEEE Trans. Mag.*, vol. 32, pp 655-658, 1996
- [Dlotko et al 2011] P. Dlotko and R. Specogna, "Efficient generalized source field computation for h-oriented magnetostatic formulations", in *Eur. Phys. J. Appl. Phys.*, 53, 20801, 2011
- [Le Menach et al 1998] Y. Le Menach, S. Clénet and F. Piriou, "Determination and utilization of the source field in 3D magnetostatic problems", in *IEEE Trans. Mag.*, vol. 34, no. 5, sept. 1998
- [Golovanov et al 1999] C. Golovanov, Y. Maréchal and G. Meunier, "A New Technique for Stranded Coil Treatment in a 3D Edge Element Based Formulation", in *IEEE Trans. Mag.*, vol. 35, no. 3, May 1999
- [Badics et al 2007] Z. Badics et Z. J. Cendes, "Source Field Modeling by Mesh Incidence Matrices", in *IEEE Trans. Mag.*, vol. 43, no. 4, April 2007

Part VI

Appendixes

Appendix A

Reference documents

The documents used in the preparation of these Principles are:

- 1 - PhD thesis of Yvonnick Le Menach [[Le Menach 1999](#)];
- 2 - EDF R&D internal report by Natacha Bereux "code_Carmel - Note de Principe", 1^{er} February 2008;
- 3 - LAMEL internal report, "Étude de calculs de champs électromagnétiques - Qualification du code_Carmel", February 2011;
- 4 - PhD thesis of Thomas Henneron [[Henneron 2004](#)];
- 5 - Summary report with a view to obtaining Authorisation to Supervise Research by Yvonnick Le Menach [[Le Menach 2012](#)].
- 6 - K. Beddek. Propagation d'incertitudes dans les modèles éléments finis en électromagnétisme – Application au contrôle non destructif par courants de Foucault. PhD thesis, Université des Sciences et Technologie de Lille, 2012.
- 7 - R. Gaignaire. Contribution à la modélisation numérique en électromagnétisme statique stochastique. PhD thesis, École Nationale Supérieure d'Arts et Métiers, 2008.
- 8 - H. Mac. Résolution numérique en électromagnétisme statique de problèmes aux incertitudes géométriques par la méthode de transformation: Application aux machines électriques. PhD thesis, École Nationale Supérieure d'Arts et Métiers, 2012.
- 9 - R. Tittarelli Estimateurs d'erreur a posteriori pour les équations de Maxwell en formulation temporelle et potentielle, PhD thesis, Université des Sciences et Technologie de Lille, 2016.
- 10 - O. Moreau Note de principe de code_Carmel 3D, 2012
- 11 - B. Goursaud Note de principe de Code_Carmel3D version 2.5.0, 2015
- 12 - L. Montier, B. Goursaud Nouveaux développements dans code_Carmel effectués dans le cadre de la thèse sur la réduction de modèles, EDF R&D technical note 6125-1717-2017-02298-FR, July 2017
- 13 - A. Pierquin Imposition d'un courant uniforme par section dans un conducteur quelconque sous code_Carmel_3D, internship report, Master 2 Scientific Calculation - USTL, March 2011 - September 2011.

Appendix B

The quasi steady-state approximation (QSSA)

The quasi-steady-state approximation (QSSA) is a simplification of Maxwell's equations obtained when sources are slowly variable over time. The QSSA is also known as the eddy current model or the magnetodynamic model. To fully understand this approximation, two approaches are detailed below.

B.1 Analysis of time constants

A mathematical study of the transition from Maxwell's equations to the approximate model is presented in [Ammari et al 2000]. For this first approach, this section follows the presentation in [P  rez et al 1990] (pp. 272-282).

The QSSA consists in neglecting the propagation time of the electrical phenomena τ_{em} in the system studied compared with the characteristic time of variation of the source T_s . For a periodic current source, this characteristic time is the time period of the current.

Hence, in a vacuum, an electromagnetic wave spreads at the speed of light,

$$c = 3 \cdot 10^8 \text{ m.s}^{-1}.$$

The propagation time between two points 3 m apart is:

$$\tau_{em} = \frac{3}{3 \cdot 10^8} = 10^{-8} \text{ s}$$

The characteristic time for a current source at 50 Hz is:

$$T_s = \frac{1}{50} = 2 \cdot 10^{-2} \text{ s}$$

We have:

$$\tau_{em} \ll T_s$$

Appendix C

U.w gauge condition

Let \mathbf{V} be a vector field defined by $\text{div } \mathbf{V} = 0$. We thus have \mathbf{V} derived from a vector potential \mathbf{U} such that $\text{rot} \mathbf{U} = \mathbf{V}$. However, if we define \mathbf{U}_1 and \mathbf{U}_2 such that their curl is equal to \mathbf{V} , this gives:

$$\mathbf{U}_1 - \mathbf{U}_2 = \text{grad} \lambda \quad (\text{C.1})$$

This relation shows that \mathbf{U} is defined at a given gradient. To ensure a unique solution, it is thus necessary to set a scalar potential λ .

Now consider a vector field \mathbf{w} whose field lines do not close and are such that they connect all points in domain \mathcal{D} . Setting the condition:

$$\begin{aligned} \mathbf{U}_1 \cdot \mathbf{w} &= f(r) \\ \mathbf{U}_2 \cdot \mathbf{w} &= f(r) \end{aligned} \quad (\text{C.2})$$

These last two relations show that:

$$\text{grad} \lambda \cdot \mathbf{w} = 0 \quad (\text{C.3})$$

This condition comes down to setting λ . If we calculate the flow along the path $\Gamma_{P \rightarrow Q}$ which is written:

$$\Gamma_{P \rightarrow Q} = \int_P^Q \text{grad} \lambda \cdot d\mathbf{l} = \lambda_Q - \lambda_P \quad (\text{C.4})$$

since field \mathbf{w} can link all the points of the mesh, it is possible to choose flow $\Gamma_{P \rightarrow Q}$ along \mathbf{w} . However, equation C.3 requires that this flow is zero, hence $\Gamma_{P \rightarrow Q}$ is zero, which requires $\lambda_P = \lambda_Q$ and therefore sets λ and gauge \mathbf{U} .

Appendix D

Incorporation of Overlapping elements into code_Carmel

This chapter seeks to explain the incorporation of Overlapping Elements (or Overelements) into code_Carmel. The Overlapping method[Demenko et al 2006][Tsukerman 1992] allows motion to be considered for any rotation, and appears from this point of view as a generalisation of the blocked step. The method implemented here derives from the work done by Xiaodong Shi[Demenko et al 2006], and extends it to the edge functions to make it compatible with the formulation in **A**.

D.1 Presentation of the Overlapping element

The Overlapping element implemented in code_Carmel is an extension of the hexahedron. It allows motion to be considered in a non-mesh domain, without having to re-mesh during the motion.

D.1.1 Reference element

The Overlapping reference element is shown in Figure D.1. Unlike the hexahedron, the coordinate of its vertices is no longer 1 or -1 along x , but $-a$ (S_1, S_5), b (S_2, S_6), c (S_3, S_7) or finally d (S_4, S_8), with $a, b, c, d \geq 1$. The integration zone of the Overlapping element is identical to that of the hexahedron: $(x, y, z) \in [-1, 1]^3$, represented by the hexahedral surface in Figure D.1.

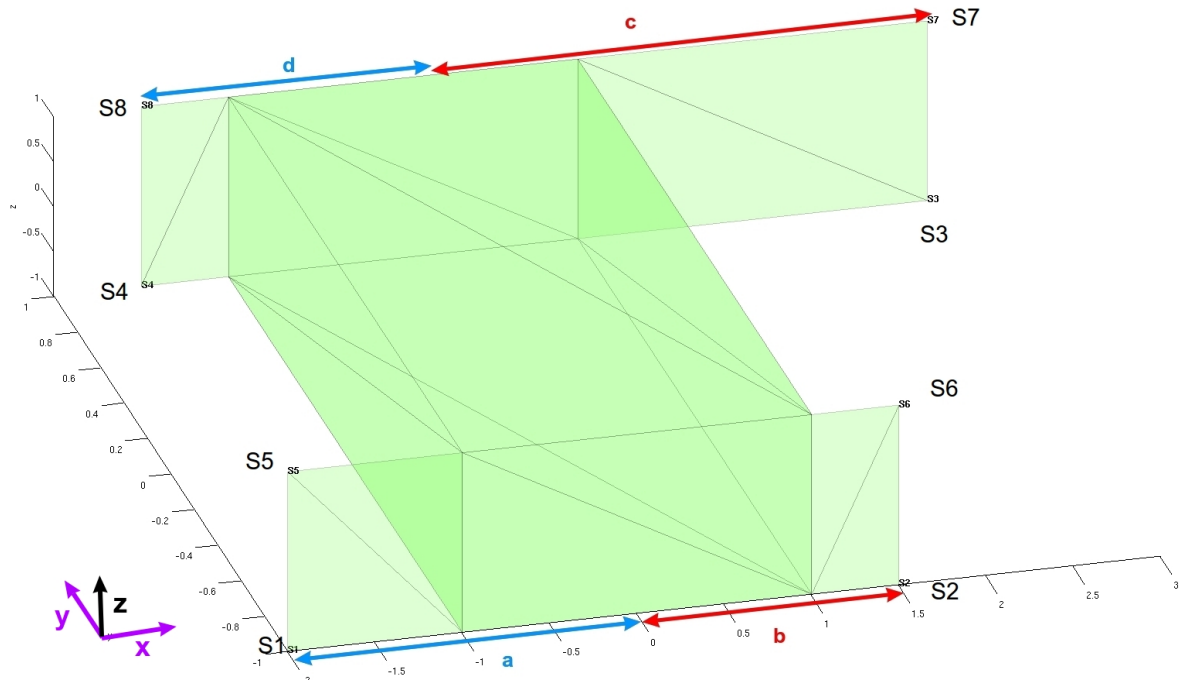


Figure D.1: Reference pyramid

In practice, however, the code uses two special cases of this general element, the left Overlapping element ($b = d = 1$) and the right Overlapping element ($a = c = 1$), as shown in Figure D.2.

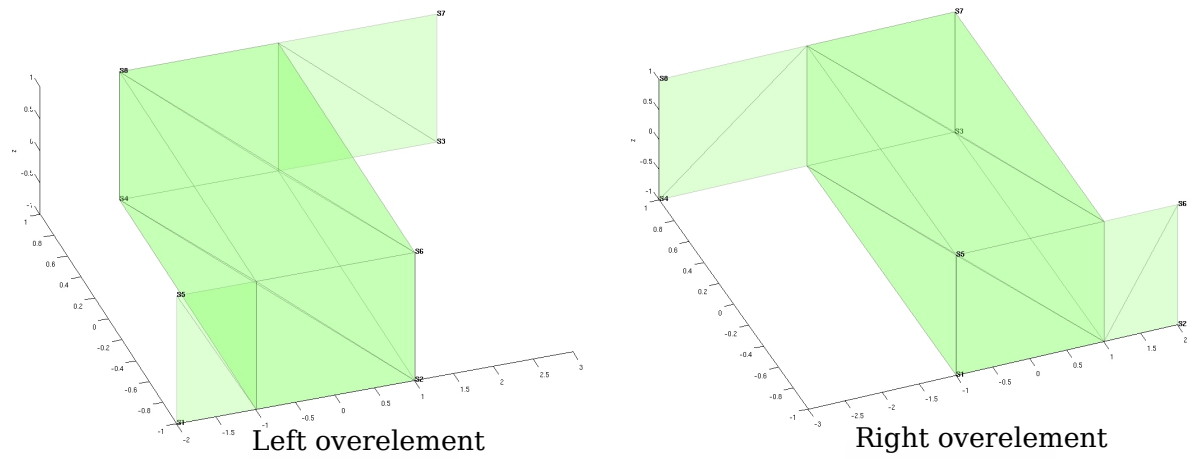


Figure D.2: Reference pyramid

The coordinates of the vertices are:

$$S_1 = \begin{pmatrix} -a \\ -1 \\ -1 \end{pmatrix}, \quad S_2 = \begin{pmatrix} b \\ -1 \\ -1 \end{pmatrix}, \quad S_3 = \begin{pmatrix} c \\ 1 \\ -1 \end{pmatrix}, \quad S_4 = \begin{pmatrix} -d \\ 1 \\ -1 \end{pmatrix}$$

$$S_5 = \begin{pmatrix} -a \\ -1 \\ 1 \end{pmatrix}, \quad S_6 = \begin{pmatrix} b \\ -1 \\ 1 \end{pmatrix}, \quad S_7 = \begin{pmatrix} c \\ 1 \\ 1 \end{pmatrix}, \quad S_8 = \begin{pmatrix} -d \\ 1 \\ 1 \end{pmatrix}$$

We will now present the shape functions used.

D.1.2 Nodal shape functions

The nodal functions are used to discretise elements belonging to $(H^1(\Omega))^3$. The nodal function associated with a node is 1 on that node, and 0 on all other nodes:

$$\int_{\{S_j\}} w_i^n \cdot \delta_{n_j} = \delta_i^j \quad (\text{D.1})$$

where δ_{n_j} is the Dirac distribution associated with node j , and δ_i^j , the Kronecker symbol. The 5 nodal functions are:

$$\begin{aligned} w_1^n(x, y, z) &= \frac{(b-x)(1-y)(1-z)}{4(a+b)} \\ w_2^n(x, y, z) &= \frac{(a+x)(1-y)(1-z)}{4(a+b)} \\ w_3^n(x, y, z) &= \frac{(d+x)(1+y)(1-z)}{4(c+d)} \\ w_4^n(x, y, z) &= \frac{(c-x)(1+y)(1-z)}{4(c+d)} \\ w_5^n(x, y, z) &= \frac{(b-x)(1-y)(1+z)}{4(a+b)} \\ w_6^n(x, y, z) &= \frac{(a+x)(1-y)(1+z)}{4(a+b)} \\ w_7^n(x, y, z) &= \frac{(d+x)(1+y)(1+z)}{4(c+d)} \\ w_8^n(x, y, z) &= \frac{(c-x)(1+y)(1+z)}{4(c+d)} \end{aligned}$$

It can be seen that they form a partition of the unit on the element.

D.1.3 Edge shape functions

The “edge” functions are used to discretise elements belonging to $H(\text{rot}, \Omega)$. They are referred to as edge functions because their circulation is equal to 1 on the edge with which they are associated, and 0 otherwise. They thus verify the following property:

$$\int_{e_j} \mathbf{w}_i^e \cdot d\mathbf{l} = \delta_{ij} \quad (\text{D.2})$$

They can be calculated using the reference formula [Geuzaine 2001]. However, with this approach, the nodal functions of the hexahedron or the Overlapping element should be used depending on whether the edge functions are calculated parallel to (Ox) , (Oy) or (Oz) . It is therefore easier to obtain them intuitively from the hexahedron.

The expressions for the functions associated with edge \mathbf{w}_{ij}^e , oriented from i to j are finally:

- for the edges along (Ox) :

$$\mathbf{w}_{12}^e = \begin{pmatrix} \frac{(1-y)(1-z)}{4(a+b)} \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{w}_{56}^e = \begin{pmatrix} \frac{(1-y)(1+z)}{4(a+b)} \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{w}_{43}^e = \begin{pmatrix} \frac{(1+y)(1-z)}{4(c+d)} \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{w}_{87}^e = \begin{pmatrix} \frac{(1+y)(1+z)}{4(c+d)} \\ 0 \\ 0 \end{pmatrix}$$

- for the edges along (Oy) :

$$\mathbf{w}_{14}^e = \begin{pmatrix} 0 \\ \frac{(1-x)(1-z)}{8} \\ 0 \end{pmatrix}, \quad \mathbf{w}_{23}^e = \begin{pmatrix} 0 \\ \frac{(1+x)(1-z)}{8} \\ 0 \end{pmatrix}, \quad \mathbf{w}_{58}^e = \begin{pmatrix} 0 \\ \frac{(1-x)(1+z)}{8} \\ 0 \end{pmatrix}, \quad \mathbf{w}_{67}^e = \begin{pmatrix} 0 \\ \frac{(1+x)(1+z)}{8} \\ 0 \end{pmatrix}$$

- for the edges along (Oz) :

$$\mathbf{w}_{15}^e = \begin{pmatrix} 0 \\ 0 \\ \frac{(1-y)(b-x)}{4(a+b)} \end{pmatrix}, \quad \mathbf{w}_{26}^e = \begin{pmatrix} 0 \\ 0 \\ \frac{(1-y)(a+x)}{4(a+b)} \end{pmatrix}, \quad \mathbf{w}_{37}^e = \begin{pmatrix} 0 \\ 0 \\ \frac{(1+y)(d+x)}{4(c+d)} \end{pmatrix}, \quad \mathbf{w}_{48}^e = \begin{pmatrix} 0 \\ 0 \\ \frac{(1+y)(c-x)}{4(c+d)} \end{pmatrix}$$

D.1.4 Gauss points

The Gauss points used are derived from those of the hexahedron in code_Aster, with 8 points. (Those in code_Carmel with 6 points appear to give very imprecise results...)

The 8 Gauss points used are:

$$\mathbf{p}_1 = \begin{pmatrix} a1 \\ a1 \\ a1 \end{pmatrix}, \quad \mathbf{p}_2 = \begin{pmatrix} a1 \\ a1 \\ -a1 \end{pmatrix}, \quad \mathbf{p}_3 = \begin{pmatrix} a1 \\ -a1 \\ a1 \end{pmatrix}, \quad \mathbf{p}_4 = \begin{pmatrix} a1 \\ -a1 \\ -a1 \end{pmatrix} \quad (\text{D.3})$$

(D.4)

$$\mathbf{p}_5 = \begin{pmatrix} -a1 \\ a1 \\ a1 \end{pmatrix}, \quad \mathbf{p}_6 = \begin{pmatrix} -a1 \\ a1 \\ -a1 \end{pmatrix}, \quad \mathbf{p}_7 = \begin{pmatrix} -a1 \\ -a1 \\ a1 \end{pmatrix}, \quad \mathbf{p}_8 = \begin{pmatrix} -a1 \\ -a1 \\ -a1 \end{pmatrix} \quad (\text{D.5})$$

with:

$$a_1 = \frac{1}{\sqrt{3}}$$

The weights used are identical and equal to:

$$w_1 = 1$$

We can verify that the sum of the 8 weights is indeed equal to 8, the area of the hexahedron on which the numerical integration is performed.

Appendix E

Taking account of non-linearity

To take account of non-linear materials with the finite element model, the constitutive relation as well as the explicit calculation of the Jacobian is explained.

E.1 Non-linear constitutive relation

To model the non-linear character of the material, a Marrocco-type law is used:

$$\nu(\|\mathbf{B}\|) = \frac{1}{\mu_0} \left(\epsilon_m + \frac{(c_m - \epsilon_m) \|\mathbf{B}\|^{2\alpha}}{\|\mathbf{B}\|^{2\alpha} + \tau_m} \right) \quad (\text{E.1})$$

where ϵ_m , τ_m and α are constants drawn from experience.

This modelling of non-linearity notably has three properties that allow the existence and uniqueness of the non-linear magnetostatic problem to be established:

$$\exists \nu_0 / \forall z, \nu_z \geq \nu_0 \quad (\text{E.2})$$

$$\exists \nu_\infty / \forall z, \frac{d\nu(z)}{dz} \leq \nu_\infty \quad (\text{E.3})$$

$$\exists M / \forall z, \frac{d\nu(z)}{dz} z + \nu(z) \leq M \quad (\text{E.4})$$

E.2 Calculation of the Jacobian

We recall the expression for the residual vector associated with the generic system of equations at the $k^{\text{ème}}$ time step:

$$\mathbf{R}(\mathbf{X}_j^k) = \left(\frac{\mathbf{K}}{\tau} + \mathbf{M}_\theta(\theta^k) + \mathbf{M}(\mathbf{X}_j^k) \right) \mathbf{X}_j^k - \mathbf{C} \mathbf{U}^k - \frac{\mathbf{K}}{\tau} \mathbf{X}^{k-1} \quad (\text{E.5})$$

The Jacobian matrix \mathbf{J} associated with the residual is thus written:

$$\mathbf{J} = \frac{\mathbf{K}}{\tau} + \mathbf{M}_\theta(\theta^k) + \bar{\mathbf{J}}(\mathbf{X}_j^k) \quad (\text{E.6})$$

with the non-linear part of the Jacobian defined by:

$$\bar{\mathbf{J}} = \frac{\partial (\mathbf{M}(\mathbf{X}_j^k) \mathbf{X}_j^k)}{\partial \mathbf{X}_j^k} \quad (\text{E.7})$$

This matrix $\mathbb{R}^{N \times N}$ represents the non-linear behaviour of ferromagnetic materials. In our model, the properties of these materials vary with $\|\mathbf{B}\|$. However, because $\mathbf{B} = \text{rot } \mathbf{A}$, only unknown \mathbf{A} generates a non-linearity (and thus the unknowns ϕ where the currents i_k , $k = 1, \dots, |\nu|$ are not *explicitement* responsible for this non-linear behaviour). Thus, and in a non-reductive manner, we present in this annex the calculation of the non-linear part of the Jacobian matrix for a magnetostatic problem without circuit coupling.

The finite element method leads to the N following equations E_i , $i = 1, \dots, N$:

$$E_i : \int_{\mathcal{D}} (\mathbf{H}(\mathbf{A}) \cdot \text{rot} \mathbf{w}_i^1) = \int_{\mathcal{D}} (\mathbf{J}_s \cdot \mathbf{w}_i^1) \quad (\text{E.8})$$

The Jacobian associated with equations E_i has the following coefficients:

$$(\bar{\mathbf{J}})_{i,j}(\mathbf{A}) = \int_{\mathcal{D}} \left(\frac{\partial \mathbf{H}(\mathbf{A})}{\partial A_j} \cdot \text{rot} \mathbf{w}_i^1 \right) d\mathcal{D} \quad (\text{E.9})$$

However, we have the following relations:

$$\mathbf{H}(\mathbf{A}) = \nu(\|\mathbf{B}\|) \mathbf{B} \quad (\text{E.10})$$

$$\mathbf{B} = \text{rot } \mathbf{A} \quad (\text{E.11})$$

Hence:

$$\frac{\partial \mathbf{H}(\mathbf{A})}{\partial A_j} = \frac{\partial \nu(\|\mathbf{B}\|)}{\partial A_j} \mathbf{B} + \nu(\|\mathbf{B}\|) \frac{\partial \mathbf{B}}{\partial A_j} \quad (\text{E.12})$$

In this sum of two terms, the second is simple to express. Knowing that $\mathbf{B} = \text{rot } \mathbf{A}$ and, using the breakdown of the finite elements $\mathbf{A} = \sum_l A_l \mathbf{w}_l^1$, we have:

$$\nu(\|\mathbf{B}\|) \frac{\partial \mathbf{B}}{\partial A_j} = \nu(\|\mathbf{B}\|) \text{rot } \mathbf{w}_j^1 \quad (\text{E.13})$$

The first term is expressed using the compound differentiation:

$$\mathbf{B} \frac{\partial \nu(\|\mathbf{B}\|)}{\partial A_j} = \mathbf{B} \frac{\partial \nu(\|\mathbf{B}\|)}{\partial \|\mathbf{B}\|} \cdot \frac{\partial \|\mathbf{B}\|}{\partial \|\mathbf{B}\|^2} \cdot \frac{\partial \|\mathbf{B}\|^2}{\partial A_j} \quad (\text{E.14})$$

Then:

$$\mathbf{B} \frac{\partial \nu(\|\mathbf{B}\|)}{\partial A_j} = \mathbf{B} \nu'(\|\mathbf{B}\|) \cdot \frac{1}{2\|\mathbf{B}\|} \cdot \left(\frac{\partial \|\mathbf{B}\|^2}{\partial B_x} \frac{\partial B_x}{\partial A_j} + \frac{\partial \|\mathbf{B}\|^2}{\partial B_y} \frac{\partial B_y}{\partial A_j} + \frac{\partial \|\mathbf{B}\|^2}{\partial B_z} \frac{\partial B_z}{\partial A_j} \right) \quad (\text{E.15})$$

And further:

$$\mathbf{B} \frac{\partial \nu(\|\mathbf{B}\|)}{\partial A_j} = \frac{\nu'(\|\mathbf{B}\|) \mathbf{B}}{2\|\mathbf{B}\|} \left(2B_x (\text{rot} \mathbf{w}_j^1)_x \mathbf{e}_x + 2B_y (\text{rot} \mathbf{w}_j^1)_y \mathbf{e}_y + 2B_z (\text{rot} \mathbf{w}_j^1)_z \mathbf{e}_z \right) \quad (\text{E.16})$$

And finally:

$$\mathbf{B} \frac{\partial \nu(\|\mathbf{B}\|)}{\partial A_j} = \frac{\nu'(\|\mathbf{B}\|)}{\|\mathbf{B}\|} (\mathbf{B} \otimes \mathbf{B}) \cdot \text{rot } \mathbf{w}_j^1 \quad (\text{E.17})$$

where the tensor product of \mathbf{B} by itself is:

$$\mathbf{B} \otimes \mathbf{B} = \begin{pmatrix} B_x B_x & B_x B_y & B_x B_z \\ B_y B_x & B_y B_y & B_y B_z \\ B_z B_x & B_z B_y & B_z B_z \end{pmatrix} \quad (\text{E.18})$$

Finally, defining the non-linear reluctivity matrix $\bar{\nu}$ by:

$$\bar{\nu} = \nu(\|\mathbf{B}\|) \mathbf{I}_3 + \frac{\nu'(\|\mathbf{B}\|)}{\|\mathbf{B}\|} \mathbf{B} \otimes \mathbf{B} \quad (\text{E.19})$$

The final expression of the Jacobian is:

$$(\bar{\mathbf{J}})_{i,j}(\mathbf{A}) = \int_{\mathcal{D}} (\bar{\nu} \text{rot} \mathbf{w}_j^1 \cdot \text{rot} \mathbf{w}_i^1) d\mathcal{D} \quad (\text{E.20})$$

E.3 Breakdown of operators into linear and non-linear parts

Although this section is trivial, it is worth recalling as it saves considerable time when assembling the full model.

Hence, it is often more efficient to separate the linear part of matrix $\mathbf{M}(\cdot)$ from the non-linear part. We thus break down $\mathbf{M}(\cdot)$ into:

$$\mathbf{M}(\cdot) = \mathbf{M}_{lin} + \mathbf{M}_{nl}(\cdot) \quad (\text{E.21})$$

where \mathbf{M}_{lin} and $\mathbf{M}_{nl}(\cdot)$ are two square matrices of $\mathbb{R}^{N \times N}$. \mathbf{M}_{lin} corresponds in particular to domains where the magnetic permeability is constant. Thus, the non-linear matrix $\mathbf{M}_{nl}(\cdot)$ is derived from the assembly of elements located in the non-linear ferromagnetic domains.

Similarly, the Jacobian can be broken down into a linear part \mathbf{J}_{lin} and a non-linear part \mathbf{J}_{nl} :

$$\mathbf{J}(\cdot) = \mathbf{J}_{lin} + \mathbf{J}_{nl}(\cdot) \quad (\text{E.22})$$

with the two matrices \mathbf{J}_{lin} and \mathbf{J}_{nl} defined by:

$$\mathbf{J}_{lin} = \frac{\mathbf{K}}{\tau} + \mathbf{M}_{\theta}(\theta^k) + \mathbf{M}_{lin} \quad (\text{E.23})$$

and:

$$\mathbf{J}_{nl}(\mathbf{X}_j^k) = \frac{\partial (\mathbf{M}_{nl}(\mathbf{X}_j^k) \mathbf{X}_j^k)}{\partial \mathbf{X}_j^k} \quad (\text{E.24})$$

$$= \frac{\partial (\mathbf{M}_{nl}(\mathbf{X}_j^k))}{\partial \mathbf{X}_j^k} \mathbf{X}_j^k + \mathbf{M}_{nl}(\mathbf{X}_j^k) \quad (\text{E.25})$$

Appendix F

Discrete model from incidence matrices

F.1 Discrete differential operators

Using the concept of incidence, “discrete” differential operators may be defined [Bossavit 1993], [Tonti 2000], [Clemens, Weiland 1987]. These are matrix operators whose construction is based on the connections between the different geometric entities that are the nodes, edges, facets and volumes. A pair of tetrahedra is then used to illustrate the developments concerning the discrete differential operators (see Figure F.1). The example shown has 5 nodes, 9 edges and 7 facets.

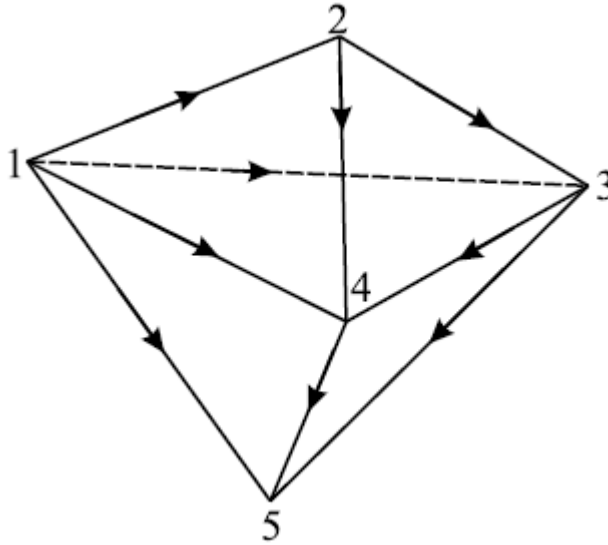


Figure F.1: Pair of tetrahedra used to illustrate the definition of incidence matrices

F.1.1 Node-edge incidence

Edges are geometric elements that are arbitrarily orientated. For example, we can choose an orientation from the node with the lowest index to the node with the highest index. The numbering of the edges as a function of the nodes, for the example in Figure F.1, is given in Table F.1.

By definition, the incidence g_{an} of a node n on an edge a is -1 if node n is the origin of edge a , 1 if n is the end of a or 0 if n does not belong to a . We thus define the incidence matrix \mathbf{G} of

edges	1	2	3	4	5	6	7	8	9
nodes	1,2	1,3	1,4	1,5	2,3	2,4	3,4	3,5	4,5

Table F.1: Edge numbering

size $n_a \times n_n$ with coefficients $(g_{an})_{(1 \leq a \leq n_a \text{ et } 1 \leq n \leq n_n)}$. For the example considered, we obtain the following matrix \mathbf{G} :

$$\mathbf{G} = \begin{pmatrix} -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 1 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix} \quad (\text{F.1})$$

Now consider two functions, one scalar, denoted u_n and the other vector, denoted \mathbf{u}_a , belonging respectively to W^0 and \mathbf{W}^1 , such that $\mathbf{u}_a = \mathbf{grad} u_n$.

we take:

$$u_n = \mathbf{U}_n^t \mathbf{W}_n = \sum u_n w_n \quad (\text{F.2})$$

and

$$\mathbf{u}_a = \mathbf{U}_a^t \mathbf{W}_a = \sum u_a \mathbf{w}_a \quad (\text{F.3})$$

Hence, we can show that:

$$\mathbf{U}_a = \mathbf{G} \mathbf{U}_n \quad (\text{F.4})$$

with $\mathbf{U}_a \in \mathcal{W}^1$ and $\mathbf{U}_n \in \mathcal{W}^0$.

Matrix \mathbf{G} can thus be considered as the discrete operator of the gradient.

F.1.2 Edge-facet incidence

Facets are also orientated geometric elements. The orientation of a facet may be given, by convention, by the direction of increasing nodes in the case of triangular facets. This convention only applies when the number of nodes per facet is less than or equal to 3. The numbering of the facets as a function of the nodes is given in Table F.2.

facets	1	2	3	4	5	6	7
nodes	1,2,3	1,2,4	1,3,4	1,3,5	1,4,5	2,3,4	3,4,5

Table F.2: Facet numbering

The incidence r_{fa} of an edge a relative to a facet f is 1 if, by traversing the boundary of the facet in the positive direction, edge a is traversed in its positive direction, -1 if the direction of a is opposite and 0 if a does not belong to f . A Using coefficients $(r_{fa})_{(1 \leq f \leq n_f \text{ et } 1 \leq a \leq n_a)}$ we define a matrix \mathbf{R} of size $n_f \times n_a$. For our example, this matrix is equal to:

$$\mathbf{R} = \begin{pmatrix} 1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 \end{pmatrix} \quad (\text{F.5})$$

Now consider a vector function, denoted \mathbf{u}_f , belonging to \mathbf{W}^2 such that $\mathbf{u}_f = \mathbf{rot} \mathbf{u}_a$. We take:

$$\mathbf{u}_f = \mathbf{U}_f^t \mathbf{W}_f = \sum u_f \mathbf{w}_f \quad (\text{F.6})$$

We can show that:

$$\mathbf{U}_f = \mathbf{R} \mathbf{U}_a \quad (\text{F.7})$$

with $\mathbf{U}_f \in \mathcal{W}^2$ and hence \mathbf{R} is the discrete operator of the curl.

F.1.3 Facet-element incidence

The numbering of the elements as a function of the for the example in Figure F.1 is given in Table F.3.

elements	1	2
nodes	1,2,3,4	1,3,4,5

Table F.3: Element numbering

The incidence d_{ef} of a facet f on an element e is 1 or -1 depending on the orientation of the normal to the facet or 0 if f does not belong to e . We can thus define matrix \mathbf{D} with coefficients $(d_{ef})_{(1 \leq e \leq n_e \text{ et } 1 \leq f \leq n_f)}$. For the example considered, the incidence matrix D of size $n_e \times n_f$ is thus equal to:

$$\mathbf{D} = \begin{pmatrix} 1 & -1 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 1 & -1 & 0 & 1 \end{pmatrix} \quad (\text{F.8})$$

For a scalar function \mathbf{u}_e belonging to \mathbf{W}^3 and defined such that $\mathbf{u}_e = \text{div} \mathbf{u}_f$. We can show that:

$$\mathbf{U}_e = \mathbf{D} \mathbf{U}_f \quad (\text{F.9})$$

with $\mathbf{U}_e \in \mathcal{W}^3$ and the discrete operator of the divergence.

F.1.4 Properties

Discrete operators have properties similar to those of differential operators in the continuous domain [Bossavit 1993]. In the case of a contractile domain, relations 5.4 remain valid on the spaces \mathcal{W}^i ($i \in \{0, 1, 2, 3\}$), and they are written:

$$\text{Ker}(\mathbf{R}(\mathcal{W}^1)) = \text{Im}(\mathbf{G}(\mathcal{W}^0)) \quad (\text{F.10})$$

$$\text{Ker}(\mathbf{D}(\mathcal{W}^2)) = \text{Im}(\mathbf{R}(\mathcal{W}^1)) \quad (\text{F.11})$$

We thus have $\mathbf{D}\mathbf{R} = \mathbf{0}$, this property remains true even if \mathbf{D} is not contractile. Conversely, if \mathbf{U}_f belonging to \mathcal{W}^2 is zero divergence, then there is a vector \mathbf{U}_a in \mathcal{W}^1 such that $\mathbf{U}_f = \mathbf{R}\mathbf{U}_a$.

Remark F.1.1 *From a practical point of view, a method based on a tree technique can be used to determine vector \mathbf{U}_a knowing vector \mathbf{U}_f (see annex G). Similarly, if the curl of \mathbf{U}_a is zero, there is a vector \mathbf{U}_n in \mathcal{W}^0 such that $\mathbf{U}_a = \mathbf{G}\mathbf{U}_n$ and we also have $\mathbf{R}\mathbf{G}\mathbf{U}_n = 0$.*

We can now propose a notation for Maxwell's equations in the discrete domain. Thus, according to the above, equations 1.3 and 1.4 can be written in the form:

$$\mathbf{R}\mathbf{E}_a = -\frac{\partial \mathbf{B}_f}{\partial t} \quad (\text{F.12})$$

$$\mathbf{D}\mathbf{B}_f = 0 \quad (\text{F.13})$$

with \mathbf{E}_a a function of $\mathcal{W}^1 \times [0, T]$ (the coefficients of vector \mathbf{E}_a are time-dependent scalar functions which represent the flows of the electric field on the edges of the mesh) and \mathbf{B}_f a function of $\mathcal{W}^2 \times [0, T]$ (the coefficients of vector \mathbf{B}_f are time-dependent scalar functions that represent the flux of the magnetic induction across the mesh facets).

F.2 Dual mesh

It is not easy to verify all Maxwell's equations on the same mesh simultaneously. Hence, it can be useful, as will be seen below, to introduce a second mesh called dual and denoted \tilde{M} that we construct from mesh M that we will refer to as primal [Bossavit, Kettunen 2000], [Tonti 2001].

Next, we will develop the construction of the dual mesh from the primal mesh. Then we will list some properties of this pair of meshes, in particular regarding the discrete operators introduced earlier.

F.2.1 Définitions

Each geometric entity in the primal mesh is matched by a geometric entity in the dual mesh: with a primal node n of M , we associate a dual element \tilde{e} of \tilde{M} , with a primal edge a a dual facet \tilde{f} , with a facet f an edge \tilde{a} and with an element e a node \tilde{n} .

Each edge a of M must traverse only one facet \tilde{f} of \tilde{M} and vice versa, and each node n of M is placed inside a element \tilde{e} of \tilde{M} and vice versa.

The orientation of each entity in \tilde{M} is deduced from the orientation of the primal entities. For example, application of the right hand rule allows deduction of the orientation of a facet \tilde{f} from the orientation of edge a . An illustration of these orientations is given in Figures F.2 and F.3.

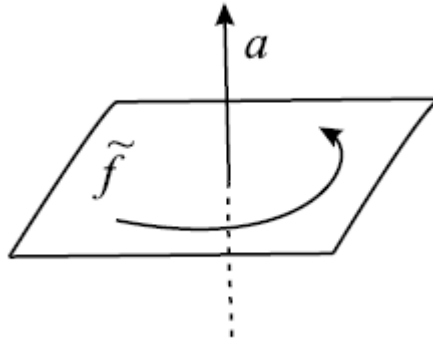
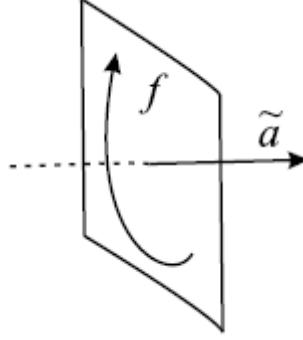


Figure F.2: Orientation of a facet \tilde{f} from the orientation of an edge a

The previous definition only gives the number of dual geometric entities and their connections. To fully define mesh \tilde{M} it is necessary to position the nodes, edges, facets and elements. Various techniques can be used. In the literature, we can find barycentric or Delaunay-Voronoi dual

Figure F.3: Orientation of an edge \tilde{a} from the orientation of a facet f

meshes. Barycentric dual meshes are based on the barycentre of each entity of the primal mesh, an edge \tilde{a} traverses a facet f at its barycentre and a dual node is located at the barycentre of a primal element, and vice versa. For Delaunay-Voronoi dual meshes, the dual edges traverse the primal facets perpendicular to their media. To illustrate these two types of dual mesh, a 2D example is given in Figures F.4 and F.5.

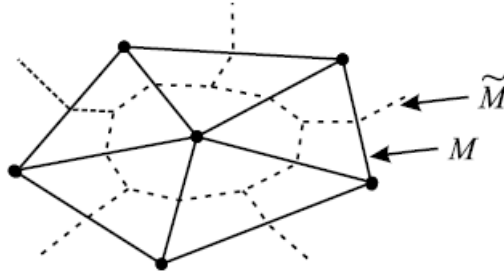


Figure F.4: Barycentric dual mesh

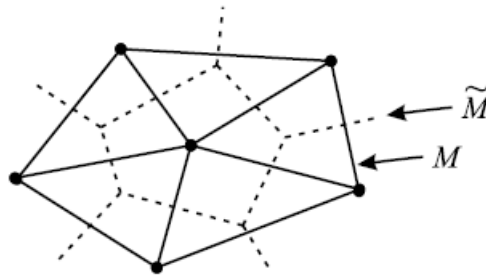


Figure F.5: Delaunay-Voronoi dual mesh

Remark F.2.1 *It should be noted that the elements generated for a dual mesh are polyhedra that can be complex shapes especially with a tetrahedral mesh. A special case is a mesh of regular hexahedra that leads to a dual mesh that is also hexahedral. This property has been put to good use in the finite integration method. In the case of the finite element method, the dual mesh is implicit and thus not constructed, and resembles a barycentric dual mesh. This aspect will be dealt with in more detail later.*

F.2.2 Properties

As with the primal mesh, an interpolation function is associated with each dual entity. Discrete spaces are also generated by these functions. We thus have \tilde{W}^0 , $\tilde{\mathbf{W}}^1$, $\tilde{\mathbf{W}}^2$ and \tilde{W}^3 the spaces generated respectively by the nodal, edge, facet and volume functions of the dual mesh and $\tilde{\mathcal{W}}^0$, $\tilde{\mathcal{W}}^1$, $\tilde{\mathcal{W}}^2$ and $\tilde{\mathcal{W}}^3$ the spaces generated by the degrees of freedom associated with the dual nodes, edges, facets and elements. These spaces have the same properties as those defined by the primal mesh.

Incidence matrices are also introduced by the various connections between dual entities. We denote $\tilde{\mathbf{G}}$, $\tilde{\mathbf{R}}$ and $\tilde{\mathbf{D}}$ the discrete differential operators of the gradient, curl and divergence respectively. Similarly, the properties of discrete operators F.10 and F.11 remain valid on the dual mesh.

As the orientation of the dual entities is deduced from the orientation of the primal entities, properties between the discrete operators of M and \tilde{M} can be demonstrated, we thus have:

$$\mathbf{G} = -\tilde{\mathbf{D}}^t \quad (\text{F.14})$$

$$\mathbf{R} = \tilde{\mathbf{R}}^t \quad (\text{F.15})$$

$$\mathbf{D} = -\tilde{\mathbf{G}}^t \quad (\text{F.16})$$

F.3 Discrete Maxwell's equations

Given that \mathbf{E} and \mathbf{B} are discretised on the primal mesh, we now discretise the magnetic field \mathbf{H} and the current density \mathbf{J} on the dual mesh. This choice is arbitrary, especially since the dual mesh of \tilde{M} is the primal mesh M itself. Inversion is thus easily possible. We define \mathbf{B}_f the degrees of freedom associated with fluxes of \mathbf{B} across all facets of M , \mathbf{E}_a the degrees of freedom associated with flows of \mathbf{E} on all edges of M , $\tilde{\mathbf{H}}_a$ the degrees of freedom associated with flows of \mathbf{H} on all edges of \tilde{M} and $\tilde{\mathbf{J}}_f$ the degrees of freedom associated with flows of \mathbf{J} across all facets of \tilde{M} . Using the discrete operators of the two meshes, Maxwell's equations are written in the form:

$$\tilde{\mathbf{R}}\tilde{\mathbf{H}}_a = \tilde{\mathbf{J}}_a \quad (\text{F.17})$$

$$\mathbf{R}\mathbf{E}_a = -\frac{\partial \mathbf{B}_f}{\partial t} \quad (\text{F.18})$$

$$\mathbf{D}\mathbf{B}_f = \quad (\text{F.19})$$

$$\tilde{\mathbf{D}}\tilde{\mathbf{J}}_f = 0 \quad (\text{F.20})$$

Concerning the boundary conditions, they are imposed on the sequences of discrete spaces of the primal and dual meshes as a function of the discretised fields. In our case, the magnetic induction is projected onto the primal mesh, so the type Γ_B condition is associated with the sequence of spaces of M . We thus define the spaces \mathcal{W}_B^i by analogy with spaces W_B^i , \mathcal{W}_B^2 is then a sub-space of \mathcal{W}^2 grouping all vectors whose coefficients correspond to facets of Γ_B are zero. Any vector of \mathcal{W}_B^2 leads to a discrete field \mathbf{U}_f of \mathcal{W}_B^2 with zero flux through Γ_B . In the same way, the sequence of discrete spaces of the dual mesh is associated with the boundary condition of type Γ_H . By introducing the projection of fields and potentials in the discrete spaces, we obtain:

- for the primal mesh:

$$\begin{array}{ccccccc}
& & \mathbf{G} & & \mathbf{R} & & \mathbf{D} \\
\mathcal{W}_B^0 & \longrightarrow & \mathcal{W}_B^1 & \longrightarrow & \mathcal{W}_B^2 & \longrightarrow & \mathcal{W}_B^3 \\
& & \uparrow & & \uparrow & & \\
& & \mathbf{E}_a & & \mathbf{B}_f & &
\end{array}$$

- for the dual mesh:

$$\begin{array}{ccccccc}
& & \tilde{\mathbf{G}} & & \tilde{\mathbf{R}} & & \tilde{\mathbf{D}} \\
\tilde{\mathcal{W}}_H^0 & \longrightarrow & \tilde{\mathcal{W}}_H^1 & \longrightarrow & \tilde{\mathcal{W}}_H^2 & \longrightarrow & \tilde{\mathcal{W}}_H^3 \\
& & \uparrow & & \uparrow & & \\
& & \tilde{\mathbf{H}}_a & & \tilde{\mathbf{J}}_f & &
\end{array}$$

F.4 Discretisation of the constitutive relations

With the discretisation of Maxwell's equations established, we now have to discretise the constitutive relations. In the continuous domain, we have “local” relations. Indeed, if we know \mathbf{H} at a point on a soft ferromagnetic material, we can calculate \mathbf{B} at that point knowing the permeability μ . In the discrete domain, fields are not given locally but rather globally in terms of flow or flux along a finite number of edges and facets, and it is thus necessary to rewrite the constitutive relations known in the continuous domain in the discrete domain [Bossavit, Kettunen 2000], [Tonti 2001a], [Marrone 2004], [Alotto, Perugia 2004], [Tarhasaari et al 1999]. Thus, relations must be found that link the different discretised magnetic and electric values: $\tilde{\mathbf{H}}_a$ with \mathbf{B}_f and \mathbf{E}_a with $\tilde{\mathbf{J}}_f$.

Several methods in the literature can be used to obtain these relations. As an example, we will determine a “discrete” constitutive relation linking $\tilde{\mathbf{H}}_a$ and \mathbf{B}_f in the case of a linear magnetostatic problem, basing it on a calculation of magnetic energy. In the continuous domain, the magnetic energy W_{mag} stored in a material characterised by a linear constitutive relation $\mathbf{B} = \mu\mathbf{H}$ is deduced from the following relation:

$$W_{mag} = \frac{1}{2} \int_{\mathcal{D}} \mathbf{H} \mathbf{B}^t d\mathcal{D} = \frac{1}{2} \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B} \mathbf{B}^t d\mathcal{D} \quad (\text{F.21})$$

by replacing \mathbf{B} with its discrete 7.9, the expression becomes:

$$W_{mag}^1 = \frac{1}{2} \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_f^t \mathbf{W}_f (\mathbf{B}_f^t \mathbf{W}_f)^t d\mathcal{D} = \frac{1}{2} \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{B}_f^t \mathbf{W}_f \mathbf{W}_f^t \mathbf{B}_f d\mathcal{D} \quad (\text{F.22})$$

We thus introduce the mass matrix $\mathbf{M}_{ff}^{\mu^{-1}}$ of size $n_f \times n_f$ such that these coefficients $m_{ff}^{\mu^{-1}}$ are written:

$$m_{ff}^{\mu^{-1}} = \int_{\mathcal{D}} \frac{1}{\mu} \mathbf{w}_f \mathbf{w}_{f'} d\mathcal{D} \quad \text{avec} \quad 1 \leq f \leq n_f \quad \text{et} \quad 1 \leq f' \leq n_f \quad (\text{F.23})$$

The magnetic energy is then:

$$W_{mag}^1 = \frac{1}{2} \mathbf{B}_f^t \mathbf{M}_{ff}^{\mu^{-1}} \mathbf{B}_f \quad (\text{F.24})$$

Another way to introduce magnetic energy into the discrete domain is to take the view that this is given by:

$$W_{mag}^2 = \frac{1}{2} \mathbf{B}_f \tilde{\mathbf{H}}_a^t \quad (\text{F.25})$$

If we want W_{mag}^1 and W_{mag}^2 to be equal for any vector \mathbf{B}_f , then we have:

$$\tilde{\mathbf{H}}_a = \mathbf{M}_{ff}^{\mu^{-1}} \mathbf{B}_f \quad (\text{F.26})$$

In this way, a constitutive relation linking $\tilde{\mathbf{H}}_a$ and \mathbf{B}_f is established.

It is also possible to determine a discrete constitutive relation if the interpolation functions on the dual mesh are known. A similar approach to that presented above is used, the magnetic energy is thus expressed as a function of the magnetic field (equation F.21). This approach leads to:

$$\mathbf{B}_f = \mathbf{M}^{\mu_{\tilde{a}\tilde{a}}} \tilde{\mathbf{H}}_a \quad (\text{F.27})$$

$$m^{\mu_{\tilde{a}\tilde{a}}} = \int_{\mathcal{D}} \mu \tilde{\mathbf{w}}_a \tilde{\mathbf{w}}_{a'} d\mathcal{D} \quad \text{avec} \quad 1 \leq a \leq n_{\tilde{a}} \quad \text{et} \quad 1 \leq a' \leq n_{\tilde{a}} \quad (\text{F.28})$$

with $\mathbf{M}^{\mu_{\tilde{a}\tilde{a}}}$ of size $n_{\tilde{a}} \times n_{\tilde{a}}$ of \tilde{M} .

Similarly, based on a calculation of electric energy, the relation linking \mathbf{E}_a and $\tilde{\mathbf{J}}_f$ is given by:

$$\tilde{\mathbf{J}}_f = \mathbf{M}_{aa}^{\sigma} \mathbf{E}_a \quad (\text{F.29})$$

with \mathbf{M}_{aa}^{σ} , of size $n_a \times n_a$ of M , whose coefficients m_{aa}^{σ} are given by:

$$m_{aa}^{\sigma} = \int_{\mathcal{D}} \sigma \mathbf{w}_a \mathbf{w}_{a'} d\mathcal{D} \quad \text{avec} \quad 1 \leq a \leq n_a \quad \text{et} \quad 1 \leq a' \leq n_a \quad (\text{F.30})$$

If the interpolation functions are linearly independent, then these matrices, called mass matrices, are invertible. We can thus link \mathbf{B}_f to $\tilde{\mathbf{H}}_a$ (\mathbf{E}_a to $\tilde{\mathbf{J}}_f$ respectively) and vice versa.

F.5 Discrete formulations

Using discretised Maxwell's equations and mass matrices, a discrete Tonti diagram can be obtained (Figure F.6). This is the reproduction in the discrete domain of the Tonti diagram associated with the continuous domain.

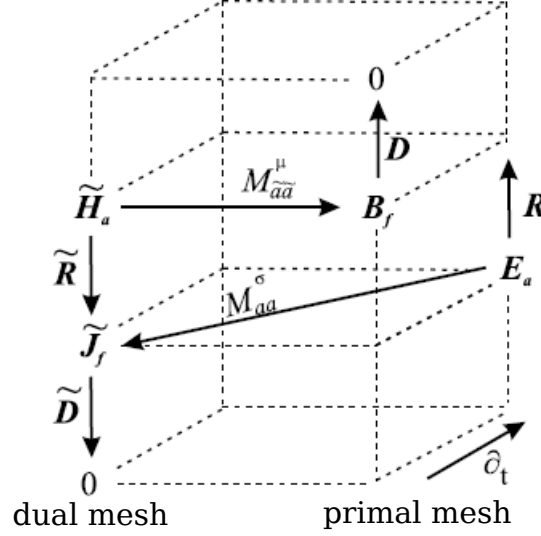


Figure F.6: Discrete Tonti diagram

On this diagram, we can note that the mass matrices “make the link” between the sequences of spaces defined on the primal mesh and the dual mesh.

Below, by using the incidence and mass matrices, we will develop the discrete formulations in potential to solve magnetodynamic, magnetostatic and electrokinetic problems. We apply the same conditions as in Chapter 2. We recall that a conductive domain \mathcal{D}_c , assumed to be contractible, is contained in domain \mathcal{D} and that the only source of fields consist of a wound inductor with its current density denoted \mathbf{J}_s . In these conditions, as in the continuous domain, the local form of Ampère’s circuital law is written:

$$\tilde{\mathbf{R}}\tilde{\mathbf{H}}_a = \tilde{\mathbf{J}}_{find} + \tilde{\mathbf{J}}_{fs} \quad (\text{F.31})$$

$$\text{with } \tilde{\mathbf{J}}_{find} = M_{aa}^\sigma \mathbf{E}_a, \tilde{\mathbf{R}}\tilde{\mathbf{H}}_{as} = \tilde{\mathbf{J}}_{fs} \text{ et } \tilde{\mathbf{J}}_{fs} = \mathbf{M}_{as}\tilde{\mathbf{J}}_{fs} \quad (\text{F.32})$$

with $\tilde{\mathbf{J}}_{find}$, all fluxes of the induced current density through the dual facets of \mathcal{D}_c ; $\tilde{\mathbf{J}}_{fs}$, the set of fluxes of \mathbf{J}_s through the facets of \tilde{M} ; and \mathbf{H}_{as} , the set of flows of the source field \mathbf{H}_s on the dual edges.

F.5.1 Current density discretisation

Depending on the need of the chosen formulation, the current density \mathbf{J}_s is either discretised on the primal mesh or on the dual mesh. If we want to have the current density on the dual mesh, the current density distribution \mathbf{J}_s is determined on the primal mesh and then projected onto the dual mesh using matrix \mathbf{M}_{af} .

In the literature, several methods can be used to calculate the current density \mathbf{J}_s for a given shape of inductor. This can be done directly, through a potential or through a source field \mathbf{H}_s such that $\text{rot}\mathbf{H}_s = \mathbf{J}_s$ [Nakata et al, 1988]. With this relation, the current density is implicitly conserved. In the case of a wound inductor of simple shape, the density \mathbf{J}_s may be determined analytically. For wound inductors, the Biot-Savart law can be used to calculate a source field. However, for volume inductors, this method proves inappropriate. Nevertheless, it allows inductors to be taken into account without explicitly meshing them [Mayergoyz 1983], [Biro, Preis 2000].

For other methods, a vector potential is calculated by minimising the quantity $(\text{rot}\mathbf{H}_s - \mathbf{J}_s)^2$ in a sub-domain of \mathcal{D} by a finite element calculation [Golovanov 1997], [Ren 1996b]. It must

of course contain the inductor while not containing a “hole”. In the case of wound inductors of complex shape, the automatic determination of this sub-domain may be difficult to construct, in which case the calculation of \mathbf{H}_s is performed throughout the domain. Other methods have also been suggested for solving an electrokinetic problem using a finite element calculation [Kawase et al 1998], [Le Floch 2002]. By considering a tensor electric conductivity, a density \mathbf{J}_s , with uniform distribution in the inductor, can be determined [Dular et al 1996].

Otherwise, there are methods based on tree techniques to directly calculate the current density \mathbf{J}_s and its associated source field \mathbf{H}_s [Le Menach 1999]. We thus introduce two vector fields such that:

$$\mathbf{J}_s = \mathbf{N} i \quad (\text{F.33})$$

$$\mathbf{H}_s = \mathbf{K} i \quad (\text{F.34})$$

$$\text{rot} \mathbf{K} = \mathbf{N} \quad (\text{F.35})$$

with i the current, \mathbf{N} defined in the inductor and discretised on \mathbf{W}^2 and \mathbf{K} defined in the whole domain and discretised on \mathbf{W}^1 . Such fields are obtained by facet and edge tree techniques respectively. The development of these trees is given in annex G. The geometry of the inductor is implicitly taken into account by \mathbf{N} such that:

$$\mathbf{N} = \frac{1}{S_{ind}} \cdot \mathbf{n} \quad (\text{F.36})$$

with S_{ind} the cross-section of the inductor and \mathbf{n} its normal. If we consider a domain \mathcal{D} including a wound inductor as shown in Figure F.7, the distribution of \mathbf{N} is given by Figure F.8.

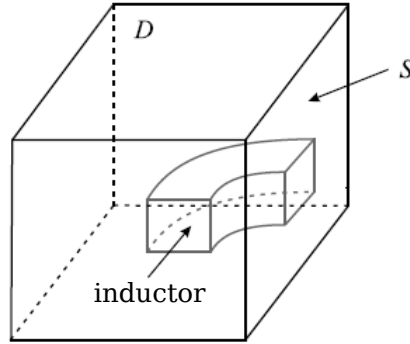
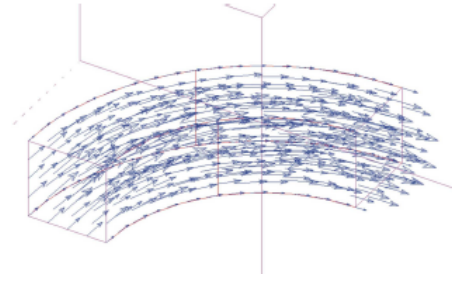


Figure F.7: Example of a wound inductor

It has been shown that the choice of the facet tree used for the calculation of \mathbf{N} induced much lower numerical errors than those due to discretisation [Le Menach 1999], [Le Menach et al 2000].

Figure F.8: Distribution of \mathbf{N}

Using an edge tree technique, an infinite number of fields \mathbf{K} can be calculated such that their curl is equal to \mathbf{N} . However, the distribution of the magnetic field does not depend on it. Figures F.9 and F.10 give two examples of field \mathbf{K} on an S surface of the domain as shown in Figure F.7.

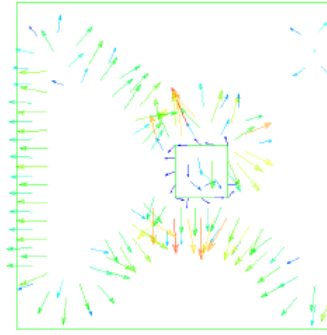


Figure F.9: Example 1

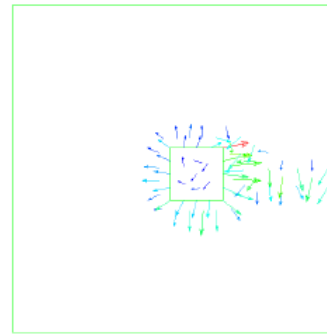


Figure F.10: Example 2

F.5.2 Magnetodynamic problem

As in the continuous domain, two formulations can be used to solve this type of problem: the electrical formulation $\mathbf{A} - \phi$ and the magnetic formulation $\mathbf{T} - \Omega$. Formulation $\mathbf{A} - \phi$ will be solved on the primal mesh and formulation $\mathbf{T} - \Omega$ on the dual mesh knowing, as mentioned earlier, that inversion is perfectly possible. In practice, the finite element method leads to solving both formulations on the primal mesh.

F.5.2.1 Electrical formulation $\mathbf{A} - \varphi$

The expression of the magnetic induction and the electric field, expressed as a function of potentials in the continuous domain, remain valid in the discrete domain. These are written:

$$\mathbf{B}_f = \mathbf{R}\mathbf{A}_a \quad \text{et} \quad \mathbf{E}_a = -\frac{\partial \mathbf{A}_a}{\partial t} - \mathbf{G}\varphi_n \quad (\text{F.37})$$

with \mathbf{A}_a the flows on the primal edges of \mathbf{A} and φ_n the scalar electric potential values φ at the primal nodes of \mathcal{D}_c . To ensure the uniqueness of \mathbf{A}_a , it is necessary to introduce a gauge. A gauge of type $\mathbf{A} \cdot \mathbf{W} = 0$ can be applied very simply if the vector potential \mathbf{A} is broken down in the space of the edge elements. This can be obtained by a tree technique, an edge tree not forming loops and connecting all the nodes of \mathcal{D}_c can be determined.

The set of edges of the tree are no longer considered as degrees of freedom, and the value of the flow of \mathbf{A} on these edges is cancelled out. As a result, the number of degrees of freedom of the potential \mathbf{A} is reduced. The system of equations associated with this formulation is given by the discretisation of the continuous domain equations. This is written:

$$\tilde{\mathbf{R}}\mathbf{M}_{ff}^{\mu^{-1}}\mathbf{R}\mathbf{A}_a + \mathbf{M}_{aa}^\sigma \left(\frac{\partial \mathbf{A}_a}{\partial t} + \mathbf{G}\varphi_n \right) = 0 \quad (\text{F.38})$$

$$\tilde{\mathbf{D}}\mathbf{M}_{aa}^\sigma \left(\frac{\partial \mathbf{A}_a}{\partial t} + \mathbf{G}\varphi_n \right) = 0 \quad (\text{F.39})$$

By using the properties of discrete operators, the system becomes:

$$\left| \begin{array}{cc} \tilde{\mathbf{R}}\mathbf{M}_{ff}^{\mu^{-1}}\mathbf{R} + \mathbf{M}_{aa}^\sigma \frac{\partial}{\partial t} & \mathbf{M}_{aa}^\sigma \mathbf{G} \\ \mathbf{G}^t \mathbf{M}_{aa}^\sigma \frac{\partial}{\partial t} & \mathbf{G}^t \mathbf{M}_{aa}^\sigma \mathbf{G} \end{array} \right| \left| \begin{array}{c} \mathbf{A}_a \\ \varphi_n \end{array} \right| = \left| \begin{array}{c} 0 \\ 0 \end{array} \right| \quad (\text{F.40})$$

The left term representing the stiffness matrix is not symmetric. By using an appropriate time discretisation method, this matrix can verify this property. The development of symmetrisation will be given later. The resulting stiffness matrix is not invertible if no gauge is used. To resolve this system, iterative methods are then used, such as the conjugated gradient method. We show that with this kind of iterative resolution process, the vector potential \mathbf{A} is implicitly gauged, hence the process is convergent [Kameari, Koganezawa 1997], [Johnson 1987], [Fujiwara et al 1993], [Ren et al 1990]. The preceding discrete formulation applies generally. Using the incidence matrices introduced above with the interpolation functions given earlier, we obtain the same formulation given by the finite element method and method of mean weighted residuals, as shown in annex H. We thus find here that the use of a dual mesh is implicit in the finite element method.

F.5.2.2 Electrical formulation $\mathbf{T} - \Omega$

The discrete expression of the magnetic field and the induced current density is given by:

$$\tilde{\mathbf{H}}_a = \tilde{\mathbf{H}}_{as} + \tilde{\mathbf{T}}_a - \tilde{\mathbf{G}}\tilde{\Omega}_n \quad (\text{F.41})$$

$$\tilde{\mathbf{J}}_f = \tilde{\mathbf{R}}(\tilde{\mathbf{T}}_a + \tilde{\mathbf{H}}_{as}) \quad (\text{F.42})$$

with $\tilde{\mathbf{H}}_{as}$ and $\tilde{\mathbf{T}}_a$ the respective flows of \mathbf{H}_s and \mathbf{T} on the dual edges and Ω_n the values of the potential Ω on the dual nodes.

The system of equations associated with this formulation is given by the discretisation of the continuous domain equations. This is written:

$$\mathbf{R}\mathbf{M}_{ff}^{\sigma^{-1}}\tilde{\mathbf{R}}\tilde{\mathbf{T}}_a + \frac{\partial}{\partial t}\mathbf{M}_{aa}^\mu \left(\tilde{\mathbf{T}}_a - \tilde{\mathbf{G}}\tilde{\Omega}_n \right) = -\mathbf{R}\mathbf{M}_{ff}^{\sigma^{-1}}\tilde{\mathbf{R}}\tilde{\mathbf{H}}_{as} - \frac{\partial}{\partial t}\mathbf{M}_{aa}^\mu \tilde{\mathbf{H}}_{as} \quad (\text{F.43})$$

$$\mathbf{D}\mathbf{M}_{aa}^\mu \left(\tilde{\mathbf{T}}_a - \tilde{\mathbf{G}}\tilde{\Omega}_n \right) = -\mathbf{D}\mathbf{M}_{aa}^\mu \tilde{\mathbf{H}}_{as} \quad (\text{F.44})$$

By using the properties of discrete operators, the system becomes:

$$\left| \begin{array}{cc} \tilde{\mathbf{R}}^t \mathbf{M}_{ff}^{\sigma^{-1}} \tilde{\mathbf{R}} + \frac{\partial}{\partial t} \mathbf{M}_{aa}^\mu & -\frac{\partial}{\partial t} \mathbf{M}_{aa}^\mu \tilde{\mathbf{G}} \\ \tilde{\mathbf{G}}^t \mathbf{M}_{aa}^\mu & -\tilde{\mathbf{G}}^t \mathbf{M}_{aa}^\mu \tilde{\mathbf{G}} \end{array} \right| \left| \begin{array}{c} \tilde{\mathbf{T}}_a \\ \tilde{\Omega}_n \end{array} \right| = \left| \begin{array}{c} -\mathbf{R}\mathbf{M}_{ff}^{\sigma^{-1}} \tilde{\mathbf{R}} \tilde{\mathbf{H}}_{as} - \frac{\partial}{\partial t} \mathbf{M}_{aa}^\mu \tilde{\mathbf{H}}_{as} \\ -\tilde{\mathbf{G}}^t \mathbf{M}_{aa}^\mu \tilde{\mathbf{H}}_{as} \end{array} \right| \quad (\text{F.45})$$

Only values defined on the dual mesh in the right term (current density \mathbf{J}_s) appear here, which is normal since the only field source is defined on the dual mesh.

As with the previous formulation, if the method of mean weighted residuals were applied, the resulting matrix system would be similar to that given earlier.

F.5.3 Magnetostatic problem

In magnetostatics, the discretisation of the system of equations described above is written in the form:

$$\tilde{\mathbf{R}}\tilde{\mathbf{H}}_a = \tilde{\mathbf{J}}_{fs} \quad (\text{F.46})$$

$$\mathbf{D}\mathbf{B}_f = 0 \quad (\text{F.47})$$

$$\tilde{\mathbf{H}}_a = \mathbf{M}_{ff}^{\mu^{-1}} \mathbf{B}_f \quad (\text{F.48})$$

In the case of formulation **A**, we get:

$$\mathbf{R}^t \mathbf{M}_{ff}^{\mu^{-1}} \mathbf{R} \mathbf{A}_{aa} = \tilde{\mathbf{J}}_{fs} \text{ avec } \mathbf{B}_f = \mathbf{R} \mathbf{A}_a \quad (\text{F.49})$$

In the right term there is a value linked to the dual mesh. However, we can obtain a system that involves only values linked to the primal mesh. Thus, the system is written:

$$\mathbf{R}^t \mathbf{M}_{ff}^{\mu^{-1}} \mathbf{R} \mathbf{A}_{aa} = \mathbf{M}_{af} \mathbf{J}_s \text{ avec } \mathbf{B}_f = \mathbf{R} \mathbf{A}_a \quad (\text{F.50})$$

In the case of formulation Ω , the system to be resolved is written:

$$\tilde{\mathbf{G}} \mathbf{M}_{aa}^\mu \tilde{\mathbf{G}} \tilde{\Omega}_n = \tilde{\mathbf{G}} \mathbf{M}_{aa}^\mu \tilde{H}_{as} \text{ avec } \tilde{H}_a = \tilde{H}_{as} - \tilde{\mathbf{G}} \tilde{\Omega}_n \quad (\text{F.51})$$

F.5.4 Electrokinetic problem

In electrokinetics, the system to be solved is given by the discretisation of the equations presented above.

$$\mathbf{R} \mathbf{E}_a = 0 \quad (\text{F.52})$$

$$\tilde{\mathbf{D}} \tilde{\mathbf{J}}_{find} = 0 \quad (\text{F.53})$$

$$\tilde{\mathbf{J}}_{find} = \mathbf{M}_{aa}^\sigma \mathbf{E}_a \quad (\text{F.54})$$

In the case of formulation **T**, the system of equations is written in the form:

$$\tilde{\mathbf{R}} \mathbf{M}_{ff}^{\sigma^{-1}} \tilde{\mathbf{R}} \tilde{\mathbf{T}}_a = 0 \text{ avec } \tilde{\mathbf{J}}_{find} = \tilde{\mathbf{R}} \tilde{\mathbf{T}}_a \quad (\text{F.55})$$

and in the case of formulation φ :

$$\mathbf{G}^t \mathbf{M}_{aa}^\sigma \mathbf{G} \varphi_n = 0 \text{ avec } \mathbf{E}_a = -\mathbf{G} \varphi_n \quad (\text{F.56})$$

Remark F.5.1 *In practice, the right term representing the source vector is not zero since the current or voltage sources are applied through the boundary conditions, which we have not considered here but are higher.*

F.6 Time discretisation

In the case of magnetodynamic formulations, in addition to space discretisation, time discretisation must also be introduced. It can be done using a backward Euler method. The interval of the study has a duration T , the time discretisation step Δt and the number of time steps N_T with $N_T = \frac{T}{\Delta t}$. In the general case, two functions are considered, a source or cause function $f(t)$ and a response or consequence function $u(t)$ such that:

$$\frac{du(t)}{dt} + au(t) = f(t) \quad (\text{F.57})$$

By applying a backward Euler method, the previous differential equation becomes:

$$\frac{u_{tn+1} - u_{tn}}{\Delta t} + au_{tn} = f_{tn} \quad (\text{F.58})$$

with $t_n = n\Delta t$ ($n \in [1, N_t]$) and A a constant.

Following time discretisation by the backward Euler method, the different stiffness matrices are made symmetric. As an example, formulation $\mathbf{A} - \varphi$ is developed. By multiplying the last line of the matrix by Δt , the matrix system becomes:

$$\left| \begin{array}{cc} \mathbf{R}^t \mathbf{M}_{ff}^{\mu-1} \mathbf{R} + \frac{M_{aa}^\sigma}{\Delta t} & M_{aa}^\sigma \mathbf{G} \\ \mathbf{G}^t M_{aa}^\sigma & \mathbf{G}^t M_{aa}^\sigma \mathbf{G} \Delta t \end{array} \right| \left| \begin{array}{c} \mathbf{A}_a \\ \varphi_n \end{array} \right|_{t_{n+1}} = \left| \begin{array}{cc} \frac{M_{aa}^\sigma}{\Delta t} & 0 \\ \mathbf{G}^t M_{aa}^\sigma & 0 \end{array} \right| \left| \begin{array}{c} \mathbf{A}_a \\ \varphi_n \end{array} \right|_{t_n} \quad (\text{F.59})$$

Appendix G

Determination of fields of given curl or divergence

Two fields are considered: \mathbf{X} (or \mathbf{X}_f) and \mathbf{Y} (or \mathbf{Y}_a) belonging respectively to \mathbf{W}^2 (or \mathcal{W}^2) and \mathbf{W}^1 (or \mathcal{W}^1). Field \mathbf{X} has conservative flux and \mathbf{Y} is such that its curl is equal to \mathbf{X} . We thus have:

$$\text{rot}\mathbf{Y} = \mathbf{X} \quad \mathbf{R}\mathbf{Y}_a = \mathbf{X}_f \quad (\text{G.1})$$

$$\text{div}\mathbf{X} = 0 \quad \mathbf{D}\mathbf{X}_f = 0 \quad (\text{G.2})$$

To obtain fields that verify the previous relations, tree techniques, based on graph theory, can be used. In this annex, we briefly recall the technique used at L2EP and developed by [\[Le Menach 1999\]](#).

G.1 Edge tree

Vector \mathbf{X}_f is known and \mathbf{Y}_a is sought, and it is must verify the relation [G.1](#). An edge tree is constructed by joining together a set of edges not forming loops and connecting all mesh nodes. The degrees of freedom (i.e. the components of \mathbf{Y}_a) associated with this tree are set to arbitrary values that may be zero, for example. The other degrees of freedom associated with the co-tree, i.e. the edges that do not belong to the tree, can then be calculated uniquely by verifying the following relation on each facet f of the mesh:

$$\int_f \mathbf{X} df = \oint_{\partial f} \mathbf{Y} d\partial f \quad (\text{G.3})$$

We then have for each facet f the sum of the flows of \mathbf{Y} along the edges of f which is equal to the flux of \mathbf{X} through f . If $(y_a)_{1 \leq a \leq n_a}$ and $(x_f)_{1 \leq f \leq n_f}$ designate the components of \mathbf{Y}_a and \mathbf{X}_f , we thus have:

$$x_f = \sum_{a=1}^{n_a} y_a \delta_a \quad (\text{G.4})$$

with $\delta_a = +1$ or -1 if a belongs to the boundary of f and 0 if a does not belong to the boundary of f . We further find the relation [G.1](#).

To illustrate this approach, we take the example of two tetrahedra shown in [Figure F.1](#). We consider a unit flux X_2 entering through facet 2 and a unit flux X_7 exiting through facet 7. The fluxes of the other external facets are set to zero so that \mathbf{X}_f has conservative flux. Given the

facet orientations, X_2 is equal to 1 and X_7 to -1. To calculate \mathbf{Y}_a , taking account of the approach presented previously, an edge tree consisting of edges 1, 2, 3 and 4 is constructed, the degrees of freedom associated with these edges are set to zero (Y_1, Y_2, Y_3 and Y_4). As a result, all that remains is to calculate the flows of \mathbf{Y} along the co-tree formed by edges 5 to 9. For information, Figures G.1 and G.2 illustrate the tree and co-tree used.

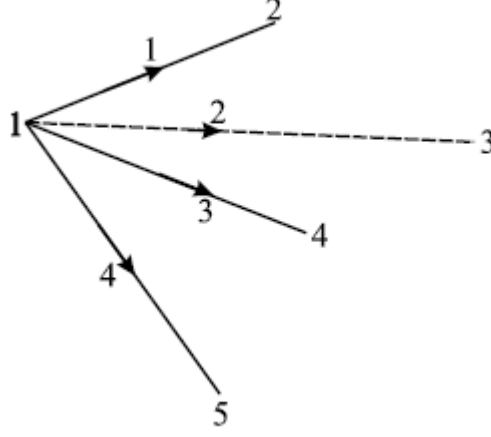


Figure G.1: Tree

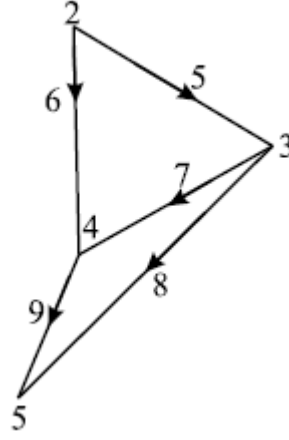


Figure G.2: Co-tree

For facet 1, we have:

$$X_1 = Y_1 + Y_5 - Y_2 = 0 \quad \text{then} \quad Y_5 = 0 \quad (\text{G.5})$$

For facet 2:

$$X_2 = Y_1 + Y_6 - Y_3 = 1 \quad \text{then} \quad Y_6 = 1 \quad (\text{G.6})$$

For facet 3:

$$X_3 = Y_2 + Y_7 - Y_3 = 0 \quad \text{then} \quad Y_7 = 1 \quad (\text{G.7})$$

and so on, for all the other facets.

By this technique, the whole vector \mathbf{Y}_a can be determined iteratively and very quickly. Note that this technique is only applicable with a field \mathbf{X} of zero divergence.

G.2 Facet tree

Because a facet connects two elements in the same way that an edge connects two nodes, by analogy, a facet tree can be determined. An element e_{ext} representing the exterior of the domain under study is added to account for the flux exiting the boundary. All facets making up the domain boundary are then connected to this external element. An example of this facet-element transposition to edge-node is given in Figure G.3 for the example shown in Figure F.1.

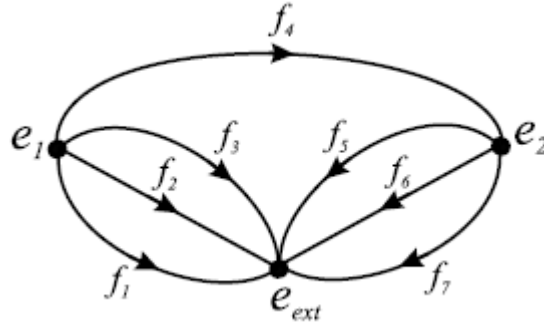


Figure G.3: Facet-element graph

Using the previous graph, a tree (representing a facet co-tree) can be calculated. We can then set the values of the flux on the facet tree. The other fluxes, through the co-co-tree facets, are determined by an iterative procedure verifying the relation G.2.

A facet tree can be used to obtain a vector \mathbf{X}_f with a given divergence. The method is illustrated using a 2D example. A zero divergence vector \mathbf{X}_f is sought in a sub-domain \mathcal{D}_X of the domain under study consisting of 6 elements as shown in Figure G.4. This sub-domain can represent a wound inductor or solid inductor.

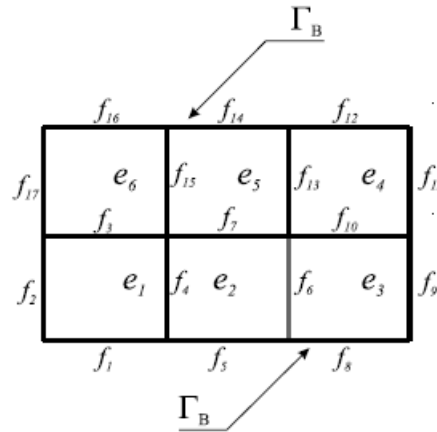


Figure G.4: Example of domain \mathcal{D}_X

The two edge surfaces of \mathcal{D}_X are in contact, with boundary conditions $\mathbf{B} \cdot \mathbf{n} = 0$.

As a first step, a facet tree is constructed, which must contain the facets outside of \mathcal{D}_X and those in contact with the boundary Γ_B , with the exception of one facet to avoid forming loops. Figure G.5 shows the facet tree and co-tree for the example studied. Figure G.6 shows the edge tree and co-tree resulting from the transposition of the facet-element relation to edge-node.

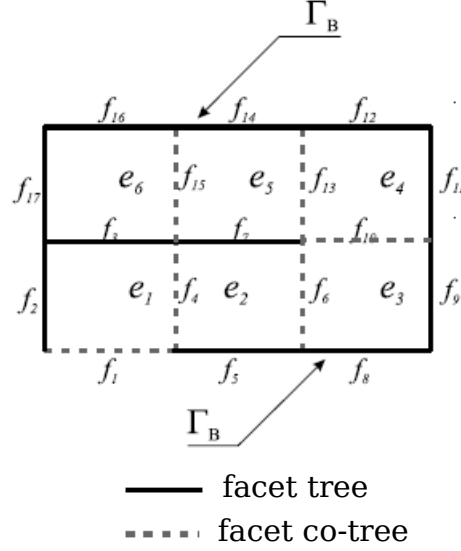


Figure G.5: Facet tree and co-tree

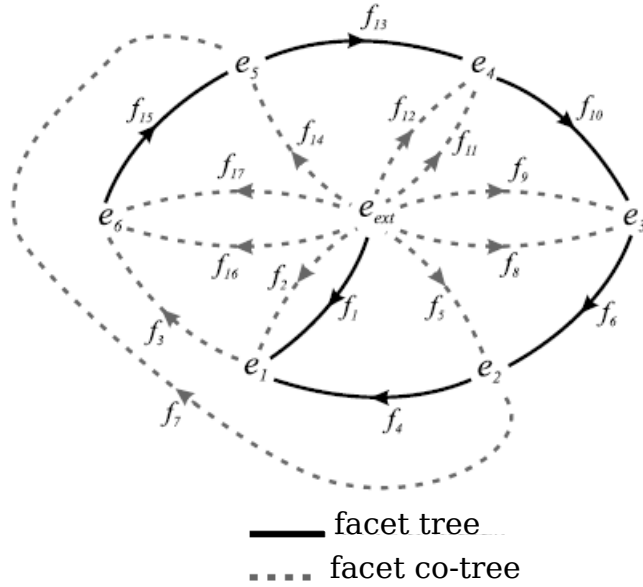


Figure G.6: Facet tree and co-tree resulting from transposition of facet-element to edge-node

The two preceding figures show that edge tree is equivalent to the facet co-tree.

In the second stage, a zero flux is imposed on facets outside domain \mathcal{D}_X to ensure zero divergence, and for our example we thus have:

$$X_2 = X_9 = X_{11} = X_{17} = 0 \quad (\text{G.8})$$

On the other tree facets, the flux is imposed, which is calculated by:

$$X_f = \int_{S_f} \mathbf{X} \cdot \mathbf{n}_f dS_f \quad (\text{G.9})$$

with S_f the area of facet f and \mathbf{n}_f its normal. In our case, the facets where we impose flux are f_5 , f_8 , f_{12} , f_{14} and f_{16} .

In the last step, it remains to calculate the flux of \mathbf{X} through the co-tree facets by verifying the relation [G.2](#). Hence, through element 6, as fluxes X_{16} and X_3 are known, we can deduce flux X_{15} . Knowing this flux, it is then possible to determine X_{13} in element 5, etc. The determination of fluxes on the facet co-tree uses an iterative method.

Appendix H

Formulation $\mathbf{A} - \varphi$

In the case of formulation $\mathbf{A} - \varphi$, the magnetic induction and electric field are expressed as a function of potentials:

$$\mathbf{B} = \text{rot} \mathbf{A} \quad \text{et} \quad \mathbf{E} = -\frac{\partial \mathbf{A}}{\partial t} - \text{grad} \varphi \quad (\text{H.1})$$

with \mathbf{A} , belonging to \mathbf{W}^1 , the vector magnetic potential defined throughout the domain and φ , belonging to \mathbf{W}^0 , the scalar electric potential defined in the conductive domain.

Using constitutive relations and replacing \mathbf{B} and \mathbf{E} by their expressions [H.1](#) in the Maxwell-Ampère and Maxwell-Faraday equations, we obtain the system of equations to be solved:

$$\text{rot} \left(\frac{1}{\mu} \text{rot} \mathbf{A} \right) + \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \text{grad} \varphi \right) = \mathbf{J}_s \quad (\text{H.2})$$

$$\text{div} \left(\frac{\partial \mathbf{A}}{\partial t} + \text{grad} \varphi \right) = 0 \quad (\text{H.3})$$

with \mathbf{J}_s , belonging to \mathbf{W}^2 , the current density, assumed to be known and uniform in a wound inductor. To resolve the previous system, the method of mean weighted residuals is used:

$$\int_{\mathcal{D}} \left[\text{rot} \left(\frac{1}{\mu} \text{rot} \mathbf{A} \right) + \sigma \left(\frac{\partial \mathbf{A}}{\partial t} + \text{grad} \varphi \right) - \mathbf{J}_s \right] \cdot \mathbf{u} \, d\mathcal{D} = 0 \quad (\text{H.4})$$

$$\int_{\mathcal{D}} \left[\text{div} \left(\frac{\partial \mathbf{A}}{\partial t} + \text{grad} \varphi \right) \right] v \, d\mathcal{D} = 0 \quad (\text{H.5})$$

with \mathbf{u} and v two test functions belonging to \mathbf{W}^1 and \mathbf{W}^0 respectively.

In the discrete domain, the potential \mathbf{A} is broken down in the space of the edge elements and φ in the spaces of nodal elements:

$$\mathbf{A} = \sum_{a=1}^{n_a} A_a \mathbf{w}_a \quad \text{et} \quad \varphi = \sum_{n=1}^{n_n} \varphi_n w_n \quad (\text{H.6})$$

where \mathbf{w}_a is the interpolation function associated with edge a and w_n the nodal function associated with node n . The source current density \mathbf{J}_s is broken down in the space of the facet elements:

$$\mathbf{J}_s = \sum_{f=1}^{n_f} J_{fs} \mathbf{w}_f \quad (\text{H.7})$$

By using the interpolation function associated with each potential¹ as test functions and replacing **A**, φ and **J_s** with their discrete forms, we obtain:

$$\sum_{a'=1}^{n_a} \int_{\mathcal{D}} \left[\frac{1}{\mu} \mathbf{rot} \mathbf{w}_a \cdot \mathbf{rot} \mathbf{w}_{a'} A_{a'} + \sigma \mathbf{w}_a \frac{\partial \mathbf{w}_{a'} A_{a'}}{\partial t} \right] d\mathcal{D} + \sum_{n=1}^{n_n} \int_{\mathcal{D}} \sigma \mathbf{w}_a \mathbf{grad} w_n \varphi_n d\mathcal{D} = \int_{\mathcal{D}} \mathbf{w}_a \sum_{f=1}^{n_f} \mathbf{w}_f J_{fs} d\mathcal{D} \quad \forall a \in [1, n_a] \quad (\text{H.8})$$

$$\sum_{a=1}^{n_a} \int_{\mathcal{D}} \sigma \mathbf{grad} w_n \frac{\partial \mathbf{w}_a A_a}{\partial t} d\mathcal{D} + \sum_{n'=1}^{n_n} \int_{\mathcal{D}} \sigma \mathbf{grad} w_n \mathbf{grad} w_{n'} \varphi_{n'} d\mathcal{D} = 0 \quad \forall n \in [1, n_n] \quad (\text{H.9})$$

The integrals on boundary Γ are naturally cancelled out by the uniform boundary conditions imposed on the potentials and fields. At this level of development, it is possible to rewrite the previous equations using the concept of incidence matrices:

$$\mathbf{R}^t \mathbf{M}_{ff}^{\mu^{-1}} \mathbf{R} \mathbf{A}_a + \mathbf{M}_{aa}^{\sigma} \left(\frac{\partial \mathbf{A}_a}{\partial t} + \mathbf{G} \varphi_n \right) = \mathbf{M}_{af} \mathbf{J}_{fs} \quad (\text{H.10})$$

$$\mathbf{G}^t \mathbf{M}_{aa}^{\sigma} \frac{\partial \mathbf{A}_a}{\partial t} + \mathbf{G}^t \mathbf{M}_{aa}^{\sigma} \mathbf{G} \varphi_n = 0 \quad (\text{H.11})$$

With $\mathbf{M}_{ff}^{\mu^{-1}}$, \mathbf{M}_{aa}^{σ} and \mathbf{M}_{af} the mass matrices. If we consider matrix \mathbf{M}_{aa}^{σ} , we recall that it is of size $n_a \times n_a$ of M , for which the coefficients m_{aa}^{σ} are given by:

$$m_{aa}^{\sigma} = \int_{\mathcal{D}} \sigma \mathbf{w}_a \mathbf{w}_{a'} d\mathcal{D} \text{ avec } 1 \leq a \leq n_a \text{ et } 1 \leq a' \leq n_a \quad (\text{H.12})$$

with \mathbf{w}_a the interpolation function associated with edge a .

¹applying the Galerkin method, $u = \mathbf{w}_a$ and $v = w_n$

Appendix I

Finding the element containing a point in code_Carmel

The algorithm used in code_Carmel, to find an element that contains a point, is as follows. For a given point, we traverse all the mesh elements. For a given element, we calculate, face by face, the signed volume of the tetrahedron formed by three of the nodes of the face and the point sought. Calculating the normal vector on the face, orientated to the inside of the element, by the vector product of two of its edges, correctly chosen, having one of the three nodes in common. We then construct the vector formed by the common node of the face and the point sought. Then we perform the scalar product of the vector normal to the face and the vector containing the point sought¹. If this result is positive, the point can be inside the element. We repeat this operation on all faces of the element. If all results are positive², the point belongs to the element.

Below is a practical example for a circular coil the find the point (0, 0, -0.18) in the index elements 227 503 (found) and 198,462 (not found). The face closest to the point sought is made up of index nodes 36684, 33046 and 36685. The “signed” volume of the tetrahedron formed by this face and the point is small ($4 \times 10^{-8} \text{ m}^3$, *i.e.* one-thousandth of the volume of these elements) but calculable without errors in precision : this volume changes the sign of the element 198 462 to 227 503. The point belongs to element 227 503 because all “signed” volumes calculated between this point and each of the faces of this element are of the same sign (negative). This is not the case for element 198 462. The hypothesis of a possible numerical precision error is invalidated (6 significant digits on the coordinates of the points only, as displayed in the SMESH module of SALOME), because we verified that the value of the “signed” volume was only slightly modified (third digit) and did not in any case change the sign of the volume.

Details on finding the explorer point of coordinates: 0.00 0.00 -0.18 in element: 215786 (mesh index : 227503).

Node coordinates:

```
1 . (mesh index: 36684) : -5.3093636276877597E-002 -6.4094322793553996E-004 -0.23979082802635701
2 . (mesh index: 33046) : 2.5929286527864698E-003 6.5272437037976194E-002 -0.18818821712945100
3 . (mesh index: 36666) : 1.7567459136458598E-002 7.7384134953159295E-002 -0.26554294932959399
4 . (mesh index: 36685) : 8.4864373823978594E-003 -2.6694983467967399E-002 -0.16581176005213899
```

Calculation of signed volume for face 1 defined by points: 1 2 3

```
- First orientated edge of the face (P1P2 = nodes 1 to 2): 5.5686564929664069E-002 6.5913380265911731E-002 5.1602610896906015E-002
- Second orientated edge of the face (P1P3 = nodes 1 to 3): 7.0661095413336192E-002 7.8025078181094831E-002 -2.5752121303236980E-002
- Normal vector to the face, oriented inwards of element (P1P2 x P1P3): -5.7237071136938536E-003 5.0803441871928295E-003 -3.1256306971144763E-004
- Vector defined between node 1 and the point sought (P1P) 5.3093636276877597E-002 6.4094322793553996E-004 5.9790828026357018E-002
- Normal scalar product face and P1P: -3.1932461619598281E-004
```

Calculation of signed volume for face 2 defined by points: 1 4 2

```
- First orientated edge of the face (P1P2 = nodes 1 to 2): 6.1580073659275453E-002 -2.6054040240031860E-002 7.3979067974218021E-002
- Second orientated edge of the face (P1P3 = nodes 1 to 3): 5.5686564929664069E-002 6.5913380265911731E-002 5.1602610896906015E-002
- Normal vector to the face, oriented inwards of element (P1P2 x P1P3): -6.2206669399010603E-003 9.4194759213992157E-004 5.5098108154132920E-003
- Vector defined between node 1 and the point sought (P1P) 5.3093636276877597E-002 6.4094322793553996E-004 5.9790828026357018E-002
```

¹This result corresponds to 6 times the “signed” volume of the tetrahedron thus formed.

²In code_Carmel, this criterion is extended to: if all the results are of the same sign, either all positive or all negative at the machine accuracy (15 significant digits), the point belongs to the element. Because we find that the “signed” volumes are more often all negative than all positive while having the independent proof that the point belongs to the element in question.

326 APPENDIX I. FINDING THE ELEMENT CONTAINING A POINT IN CODE_CARMEL

- Normal scalar product face and P1P: -2.3794205431372593E-007

Calculation of signed volume for face 3 defined by points: 1 3 4

- First orientated edge of the face (P1P2 = nodes 1 to 2): 7.0661095413336192E-002 7.8025078181094831E-002 -2.5752121303236980E-002
- Second orientated edge of the face (P1P3 = nodes 1 to 3): 6.1580073659275453E-002 -2.6054040240031860E-002 7.3979067974218021E-002
- Normal vector to the face, oriented inwards of element (P1P2 x P1P3): 5.1012757577521724E-003 -6.8132595074518335E-003 -6.6457970849663371E-003
- Vector defined between node 1 and the point sought (P1P) 5.3093636276877597E-002 6.4094322793553996E-004 5.9790828026357018E-002
- Normal scalar product face and P1P: -1.3087934351660875E-004

Calculation of signed volume for face 4 defined by points: 2 4 3

- First orientated edge of the face (P1P2 = nodes 1 to 2): 5.8935087296113891E-003 -9.1967420505943587E-002 2.2376457077312006E-002
- Second orientated edge of the face (P1P3 = nodes 1 to 3): 1.4974530483672128E-002 1.2111697915183101E-002 -7.7354732200142995E-002
- Normal vector to the face, oriented inwards of element (P1P2 x P1P3): 6.8430982958427415E-003 7.9096772811908353E-004 1.4485493392644932E-003
- Vector defined between node 1 and the point sought (P1P) -2.5929286527864698E-003 -6.5272437037976194E-002 8.1882171294510031E-003
- Normal scalar product face and P1P: -5.7511020365228410E-005

The explorer point of coordinates: 0.00 0.00 -0.18 corresponds to element: 215786 (mesh index: 227503).

Details on finding the explorer point of coordinates: 0.00 0.00 -0.18 in element: 186745 (mesh index : 198462).

Node coordinates:

1 . (mesh index: 36684) : -5.3093636276877597E-002 -6.4094322793553996E-004 -0.23979082802635701
2 . (mesh index: 33045) : -4.2877107273884198E-002 3.3669352879108903E-002 -0.18074067450383599
3 . (mesh index: 33046) : 2.5929286527864698E-003 6.5272437037976194E-002 -0.18818821712945100
4 . (mesh index: 36685) : 8.4864373823978594E-003 -2.6694983467967399E-002 -0.16581176005213899

Calculation of signed volume for face 1 defined by points: 1 2 3

- First orientated edge of the face (P1P2 = nodes 1 to 2): 1.0216529002993399E-002 3.4310296107044447E-002 5.9050153522521021E-002
- Second orientated edge of the face (P1P3 = nodes 1 a 3) : 5.5686564929664069E-002 6.5913380265911731E-002 5.1602610896906015E-002
- Normal vector to the face, oriented inwards of element (P1P2 x P1P3): -2.1216943641209516E-003 2.7611006373800748E-003 -1.2372165707489110E-003
- Vector defined between node 1 and the point sought (P1P) 5.3093636276877597E-002 6.4094322793553996E-004 5.9790828026357018E-002
- Normal scalar product face and P1P: -1.8485296331716895E-004

Calculation of signed volume for face 2 defined by points: 1 4 2

- First orientated edge of the face (P1P2 = nodes 1 to 2): 6.1580073659275453E-002 -2.6054040240031860E-002 7.3979067974218021E-002
- Second orientated edge of the face (P1P3 = nodes 1 to 3): 1.0216529002993399E-002 3.4310296107044447E-002 5.9050153522521021E-002
- Normal vector to the face, oriented inwards of element (P1P2 x P1P3): -4.0767388039744112E-003 -2.8805035099353500E-003 2.3790124193007914E-003
- Vector defined between node 1 and the point sought (P1P) 5.3093636276877597E-002 6.4094322793553996E-004 5.9790828026357018E-002
- Normal scalar product face and P1P: -7.6052004036806842E-005

Calculation of signed volume for face 3 defined by points: 1 3 4

- First orientated edge of the face (P1P2 = nodes 1 to 2): 5.5686564929664069E-002 6.5913380265911731E-002 5.1602610896906015E-002
- Second orientated edge of the face (P1P3 = nodes 1 to 3): 6.1580073659275453E-002 -2.6054040240031860E-002 7.3979067974218021E-002
- Normal vector to the face, oriented inwards of element (P1P2 x P1P3): 6.2206669399010603E-003 -9.4194759213992157E-004 -5.5098108154132920E-003
- Vector defined between node 1 and the point sought (P1P) 5.3093636276877597E-002 6.4094322793553996E-004 5.9790828026357018E-002
- Normal scalar product face and P1P: 2.3794205431372593E-007

Calculation of signed volume for face 4 defined by points: 2 4 3

- First orientated edge of the face (P1P2 = nodes 1 to 2): 5.1363544656282054E-002 -6.0364336347076303E-002 1.4928914451697001E-002
- Second orientated edge of the face (P1P3 = nodes 1 to 3): 4.5470035926670670E-002 3.1603084158867291E-002 -7.4475426256150057E-003
- Normal vector to the face, oriented inwards of element (P1P2 x P1P3): -2.2233771805698483E-005 1.0613504646951959E-003 4.3680149668614120E-003
- Vector defined between node 1 and the point sought (P1P) 4.2877107273884198E-002 -3.3669352879108903E-002 7.4067450383599742E-004
- Normal scalar product face and P1P: -3.3453025824716477E-005

Appendix J

Libraies of linear algebra

J.1 Expression of needs

Given the profusion of offers and positive feedback, the question of whether to use a library or an external product is now unavoidable.

Why are we looking to use this type of scientific library to replace or complement our in-house solutions? Because this strategy can pay off immediately by reconciling several objectives:

- Economic objective: less technical, less invasive and much faster developments in the host code. Especially as these are generally not “core business” developments.
- Performance objective: it would be very difficult to do as well, because these products capitalise on decades of highly specialised expertise from international teams. They often combine efficiency, reliability, performance and portability. They allow to address, at a lower cost, a large scope of application while outsourcing many of the associated contingencies (problem typology, data representation and control, etc.).
- Sharing objective: we benefit from the feedback of a diverse user community.
- Standardisation objective: we share the risk with other users regarding the durability of the product over time, but in return we benefit from the visibility/recognition that this provides. And that’s not counting the “skills pool” aspects for our code development teams.

J.1.1 Management of loss of control

However, this loss of control of this often invisible but important link in the numerical simulation chain must be managed with foresight. In order to be profitable over time, this strategy must be accompanied by:

- *Maintaining “numerical computing” skills* in house to recommend the right product, optimise its use and integrate it into our codes. We also need to plan for regular efforts to upgrade/maintain/validate/document these functions, even if these are less extensive than for a purely in-house solution. These peripheral software projects also enable us to maintain a certain credibility and responsiveness with academic teams.
- If possible, *a partnership with the product development team* to maintain privileged channels of expertise (for some of our most pressing problems) and influence these future developments.

It is for all these reasons that EDF R&D has been engaged for 7 years in a very active partnership with the MUMPS development team. This collaboration allowed for a fruitful exchange of information (OPEX, bugs, usage tips, expertise) between EDF R&D and the MUMPS team. In

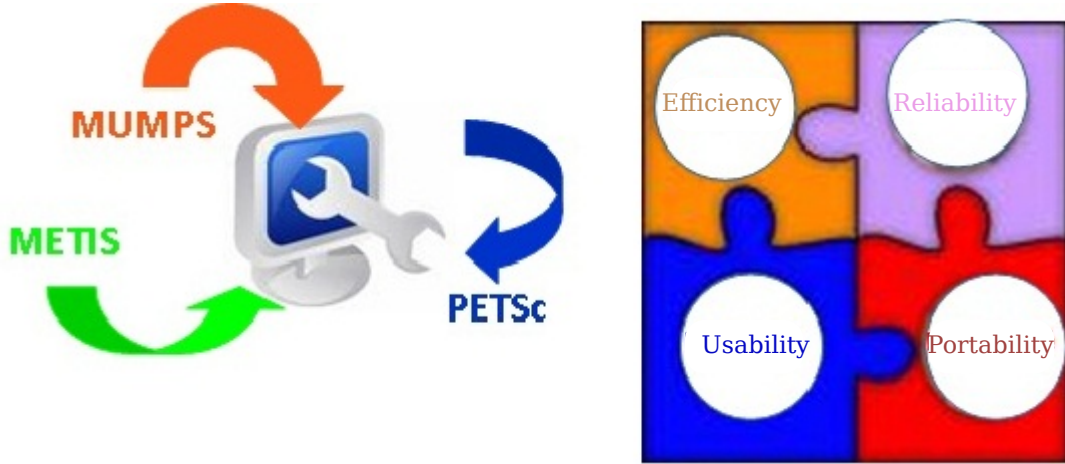


Figure J.1: Why use an external product to replace or complement our in-house solutions?

addition, several features of the product have been corrected or modified to take account of these exchanges.

Rapid optimisation of uses of MUMPS in code_Carmel, and its intensive use in Code_Aster, are the direct fruits of this pooling of our in-house resources and the close ties that unite us to the product team.



Figure J.2: Some “logos” of linear algebra libraries

J.1.2 A wide range of linear algebra libraries

Since the emergence in the 70s/80s of the first public libraries¹ and constructors².

Remark J.1.1 *To structure their use more efficiently and offer “black box” solutions to code teams, macro-libraries have emerged. They bring together a panel of these products to which they add “house” solutions: Numerical Platon (CEA-DEN), Arcane (CEA-DAM), etc.*

More specifically concerning *direct methods for solving linear systems, which are the core target of our study*, around thirty packages are available. A distinction is made between “stand-alone” products and those incorporated into a library, between public and commercial products, between those dealing with dense problems and others with sparse. Some work only in sequential mode, others support shared and/or distributed memory parallelism. Finally, some products are general

¹EISPACK(1974), LINPACK(1976), BLAS(1978) and then LAPACK(1992)...

²NAG(1971), IMSL/ESSL(IBM 1971), ASL/MathKeisan(NEC), SciLib(Cray), MKL(Intel/Bull), HSL(Harwell), CASI(ANSYS, ABAQUS...), etc.) and their communities of users, the offer has increased. The tendency is of course to offer high performance solutions (vector, parallelism with centralised and then distributed memory, multilevel parallelism via threads) as well as “tool kits” for handling linear algebra algorithms and associated data structures. To cite a non-exhaustive list: ScaLAPACK(Dongarra & Demmel 1997), SparseKIT(Saad 1988), PETSc(Argonne 1991), HyPre(LL 2000), TRILINOS(Sandia 2000), etc

(symmetric, non-symmetric, SPD, real/complex, etc.) while others are adapted to a specific need/scenario.

DIRECT SOLVERS	License	Support	Real	Complex	F77	C	Seq	Dist	SPD	Gen
DENSE										
FLAME	LGPL	Yes	X	X	X	X	X			
LAPACK	BSD	Yes	X	X	X	X	X			
LAPACK95	BSD	Yes	X	X	95		X			
NAPACK	BSD	Yes	X		X		X			
PLAPACK	?	Yes	X	X	X	X		M		
PRISM	?	No	X		X		X	M		
ScaLAPACK	BSD	Yes	X	X	X	X		M/P		
Trilinos/Pliris	LGPL	Yes	X	X		XandC++		M		
SPARSE										
DSCPACK	?	Yes	X			X	X	M	X	
HSL	?	Yes	X	X	X		X		X	X
MFACT	?	Yes	X			X	X	M	X	
MUMPS	PD	Yes	X	X	X	X	X	M	X	X
PSPASES	?	Yes	X		X	X		M	X	
SPARSE	?	?	X	X		X	X		X	X
SPOOLES	PD	?	X	X		X	X	M		X
SuperLU	Own	Yes	X	X	X	X	X	M		X
TAUCS	Own	Yes	X	X		X	X		X	X
Trilinos/Amesos	LGPL	Yes	X				X	M	X	X
UMFPACK	LGPL	Yes	X	X		X	X			X
Y12M	?	Yes	X		X		X		X	X

Figure J.3: Extract from Jack Dongarra's web page on free products implementing a direct method.

A fairly exhaustive list of all these products can be found on the website of one of the founding fathers of LAPACK/BLAS: Jack Dongarra³. The table below is a redacted version. This Internet resource also lists packages implementing iterative solvers, preconditioners, modal solvers and many support products (BLAS, LAPACK, ATLAS, etc.).

Remark J.1.2 A more detailed Internet resource focused on sparse direct solvers is maintained by another big name in the numerical world: T.A.Davis⁴, one of the contributors to Matlab.

J.2 Annex: Theoretical supplements

J.2.1 Krylov spaces

The action of the conjugate gradient (CG) can be summarised in one sentence: “It carries out orthogonal projections⁵ successive on the Krylov space $\kappa_i(\mathbf{K}, \mathbf{r}^0) := \text{vect}(\mathbf{r}^0, \mathbf{K} \mathbf{r}^0, \dots, \mathbf{K} \mathbf{r}^{i-1})$ where \mathbf{r}^0 is the initial residual”.

We thus resolve the linear system (P_1) by seeking an approximate solution \mathbf{u}^i in the refined sub-space (search space of dimension N):

$$\mathcal{A} = \mathbf{r}^0 + \kappa_i(\mathbf{K}, \mathbf{r}^0) \quad (\text{J.1})$$

while imposing the orthogonal constraint (constraint space of dimension N):

$$\mathbf{r}^i := \mathbf{f} - \mathbf{K} \mathbf{u}^i \perp \kappa_i(\mathbf{K}, \mathbf{r}^0) \quad (\text{J.2})$$

This Krylov space has the useful property of facilitating approximation of the solution, at the end of m iterations, in the form:

³<http://www.netlib.org/utk/people/JackDongarra/la-sw.html>.

⁴<http://www.cise.ufl.edu/research/sparse/codes/>.

⁵We thus construct a succession of iterations by projection on an approximate sub-space (called the search space) and perpendicular to another sub-space (called the constraint space). This general framework constitutes what we call the Petrov-Galerkin conditions. Here these two sub-spaces are confused and equal to a Krylov space.

$$\mathbf{K}^{-1} \mathbf{f} \approx \mathbf{u}^m = \mathbf{r}^0 + P_{m-1}(\mathbf{K}) \mathbf{f} \quad (\text{J.3})$$

where P_{m-1} is a certain matrix polynomial of order $m-1$. We show that the residuals and directions of descent generate this space:

$$\begin{aligned} \text{vec}(\mathbf{r}^0, \mathbf{r}^1, \dots, \mathbf{r}^{m-1}) &= \kappa_m(\mathbf{K}, \mathbf{r}^0) \\ \text{vec}(\mathbf{d}^0, \mathbf{d}^1, \dots, \mathbf{d}^{m-1}) &= \kappa_m(\mathbf{K}, \mathbf{r}^0) \end{aligned} \quad (\text{J.4})$$

while allowing the approximate solution, \mathbf{u}^m , to minimise the energy of the norm over the entire refined space \mathcal{A} :

$$\|\mathbf{u}^m\|_{\mathbf{K}} < \|\mathbf{u}\|_{\mathbf{K}} \quad \forall \mathbf{u} \in \mathcal{A} \quad (\text{J.5})$$

This joint result illustrates the optimality of CG: unlike descent methods, the minimum energy is not achieved successively for each descent direction \mathbf{d}^i , but jointly for all descent directions already obtained.

Remark J.2.1 *There are a wide variety of projection methods on Krylov-like spaces, more precisely called “Krylov methods”. To solve linear systems (GC, GMRES, FOM/IOM/DOM, GCR, ORTHODIR/MIN, etc.) and/or modal problems (Lanczos, Arnoldi, etc.). They differ by the choice of their constraint space and the preconditioning applied to the initial operator to form the working operator, knowing that different implementations lead to radically different algorithms (vector or block version, orthonormalisation tools, etc.).*

J.2.2 Orthogonality

As already noted, the descent directions are \mathbf{K} orthogonal with respect to each other. In addition, the choice of the optimal descent parameter (see section 15.2.1 or step (2) of algorithm 15.1) imposes, step by step, the orthogonalities:

$$\begin{aligned} \langle \mathbf{d}^i, \mathbf{r}^m \rangle &= 0 \quad \forall i < m \\ \langle \mathbf{r}^i, \mathbf{r}^m \rangle &= 0 \end{aligned} \quad (\text{J.6})$$

We thus note a slight inexactitude in the name “CG”, as the gradients are not conjugate and the conjugate directions do not only include gradients. But let’s not quibble, the designated ingredients are there all the same!

After N iterations, two possibilities appear:

- Either the residual is zero $\mathbf{r}^N = 0 \Rightarrow$ convergence.
- Or it is orthogonal to the N previous descent directions which constitute a basis of the finite approximation space \mathbb{R}^N (as they are linearly independent). Hence the necessity for $\mathbf{r}^N = 0 \Rightarrow$ convergence.

It would seem that CG is a direct method that converges in at most N iterations, at least this is what we thought before testing it on practical cases! Because what remains true in theory, in exact arithmetic, is undermined by the finite arithmetic of computers. *Progressively, notably due to rounding errors, the descent directions lose their beautiful conjugation properties and the minimisation leaves the required space.*

In other words, we solve an approximate problem that is no longer quite the wished-for projection of the initial problem. The (theoretically) direct method reveals its true nature! It is iterative and thus subject, in practice, to many uncertainties (matrix conditioning, starting point, stop tests, accuracy of the orthogonality, etc.).

To correct this, when constructing the new descent direction, we can impose a *re-orthogonalisation phase*. This widespread practice in modal analysis and domain decomposition can be found in several variants: total, partial, selective re-orthogonalisation, etc. via a whole range of orthogonalisation procedures (GS, GSM, IGSM, Householder, Givens, etc.). Other palliative solutions

may also consist in, periodically, *explicitly recalculating the residual* (step (4) of algorithm 15.1) or even *restarting the algorithm* with the last-found approximation as the initial solution. However, if in the end they do not always win in terms of computation time (they have an additional cost to be offset), they often increase the robustness of the process.

J.2.3 Convergence

Due to the particular structure of the approximating space (equation J.3) and the minimisation property on this space of the approximate solution \mathbf{u}^m (see equation J.5), we obtain an estimate of the convergence rate of the CG:

$$\|\mathbf{u} - \mathbf{u}^i\|_{\mathbf{K}}^2 = (\omega^i)^2 \|\mathbf{u}^0 - \mathbf{u}\|_{\mathbf{K}}^2 \text{ avec } \omega^i := \max_{1 \leq i \leq N} (1 - \lambda^i P_{m-1}(\lambda^i)) \quad (\text{J.7})$$

where we denote $(\lambda^i, \mathbf{v}^i)$ the eigenmodes of matrix \mathbf{K} and P_{m-1} any polynomial of degree $m-1$ at most. The famous Chebyshev polynomials, through their useful properties of increasing the polynomial space, improve the readability of this attenuation factor ω^i . At the end of i iterations, the descent is expressed in the form

$$\|\mathbf{u}^i - \mathbf{u}\|_{\mathbf{K}} \leq 2 \left(\frac{\sqrt{\eta(\mathbf{K})} - 1}{\sqrt{\eta(\mathbf{K})} + 1} \right)^2 \|\mathbf{u}^0 - \mathbf{u}\|_{\mathbf{K}} \quad (\text{J.8})$$

It ensures *superlinear convergence*, $\lim_{i \rightarrow \infty} \frac{J(\mathbf{u}^{i+1}) - J(\mathbf{u})}{J(\mathbf{u}^i) - J(\mathbf{u})} = 0$ i.e., of the process in a number of iterations proportional to the square root of the conditioning of the operator.

Thus, to obtain:

$$\frac{\|\mathbf{u}^i - \mathbf{u}\|_{\mathbf{K}}}{\|\mathbf{u}^0 - \mathbf{u}\|_{\mathbf{K}}} \leq \varepsilon \text{ (petit)} \quad (\text{J.9})$$

requires a number of iterations in the order of

$$i \approx \frac{\sqrt{\eta(\mathbf{K})}}{2} \ln \frac{2}{\varepsilon} \quad (\text{J.10})$$

For example, on the test case of Rubinacci's cube processed with Code_Carmel v1.7.6, we have the following number of iterations (depending on the preconditioner used and the precision ε desired):

Type of preconditioner	$\varepsilon = 10^{-3}$	$\varepsilon = 10^{-6}$	$\varepsilon = 10^{-9}$
Number of iterations in theory according to the formula J.10	Reference	X2	X3
Without (LinearSolverType=0)	236	567	965
Crout ILU(0) (LinearSolverType=1)	45	107	179
Relaxed single precision MUMPS LinearSolverType=3 + mumps_relax = 10^{-3}	2	4	6

Table J.1: Theoretical and actual convergence of the CG on the test case of Rubinacci's cube (Code_Carmel v1.7.6 on a 7-caliber station).

We note that while the number of iterations does not strictly follow the theoretical changes predicted by the formula, *the orders of magnitude are respected*. This compliance is further verified by the low number of iterations, i.e. the preconditioner proves to be effective. For example, the increases in the number of MUMPS iterations are closer to the expected numbers than those

of Crout (and still more so in the case without a preconditioner). This is no doubt due to the deleterious effect of loss of orthogonality.

Remark J.2.2 *In practice, taking advantage of special circumstances, the best starting point and/or advantageous spectral distribution, CG convergence can be much better than might be expected (J.10). As Krylov’s methods tend to uncover extreme eigenvalues as a matter of priority, the “effective conditioning” of the working operator is improved.*

J.2.4 Computation and memory costs

As with the Steepest Descent, *most of the computation cost* (excluding the preconditioner, see section 15.2.3) *of this algorithm lies* in step (1), the matrix-vector product. Its complexity is the order of $\mathcal{O}(k c N)$ where c is the average number of non-zero terms per line of \mathbf{K} and k the number of iterations required for convergence. To be much more effective than a simple Cholesky (of complexity $\mathcal{O}\left(\frac{N^3}{3}\right)$) thus requires:

- Taking full account of the sparse character of matrices resulting from finite element discretisations (storage MORSE, matrix-vector product optimised ad hoc, dedicated data representation format): $c \ll N$.
- Preconditioning the working operator: $c \ll N$.
- Optimising all steps, even the most basic ones (steps (3), (4) and (7)), because they will be repeated many times (calls on optimised BLAS functions, parallelism, etc.).

It has already been pointed out that, for an SPD operator, its theoretical convergence occurs in at most N iterations and proportionally to the square root of the conditioning (see J.10). In practice, for large systems that are poorly conditioned and mostly out of scope, it can be very slow to appear. In terms of memory usage, only the storage of the working matrix is possibly required ($\mathcal{O}(c N)$) plus some auxiliary working vectors ($\mathcal{O}(3 N)$). In practice, the introduction of sparse computer storage requires the management of additional integer vectors: for example for the MORSE storage used in Code_Carmel, vectors of the end-of-row indices and the column indices of the profile elements. Hence *effective memory complexity* of $\mathcal{O}((c + 3) N)$ real and $\mathcal{O}(c N + N)$ whole.

Remark J.2.3 *These considerations on memory usage do not take into account the storage problems of a possible preconditioner and the workspace temporarily occupied for its construction.*

Remark J.2.4 *The number of non-zero terms seems relatively small in cases processed using Code_Carmel (Rubinacci’s cube): $c \approx 10$. This very sparse character, combined with good matrix conditioning ($\eta(\mathbf{K}) \propto 10^6$), may explain the highly competitive performance of PCG. And this, even with “defective” preconditioning of the Jacobi or ILU(0) type. In the end, for as long as we don’t diverge... we accept iterating a lot, because these iterations come at a low cost.*

J.3 Annex: Non-linear resolution strategies

J.3.1 Construction of the preconditioner

This occurs through the parameter `reacprecond_methodeNL`.

In non-linear, we can also take great advantage of pooling, between several tens of iterations of the non-linear solver (often a Newton algorithm), of the construction of the preconditioner or the numerical factorisation of the direct solver. The non-linear process that operates on approximate data may then require more iterations, but in the end, as these are faster, the user often wins!

This strategy is especially beneficial for the most costly combination: PCG + MUMPS preconditioner (`LinearSolverType=3`). With a strictly positive value of this keyword (e.g. 30), the preconditioner is recalculated with the last code_Carmel matrix only if:

- For a given non-linear solver iteration, the PCG conjugate gradient has been through more than `reacprecond_methodeNL` iterations. To be consistent, we must verify that parameter `reacprecond_methodeNL` is *strictly less* than `nbIterationMax`, otherwise this criterion will never be enabled. A warning notifies the user when this condition is not met.
- That makes at least `reacprecond_methodeNL` iterations of the non-linear solver without this re-calculation.
- The residual of the non-linear solver increases rather than decreases.

It also works for other preconditioners (`LinearSolverType=1/2`, see section 15.2.2.3), but since they already have a low cost in time, the gains are often modest.

With *MUMPS direct solver* (`LinearSolverType=4`), we also pool the most time-consuming step, the numerical factorisation step (and the analysis phase that precedes it). It is recalculated with the last code `_Carmel` matrix only if:

- That makes at least `reacprecond_methodeNL` iterations of the non-linear solver without this re-calculation.
- The residual of the non-linear solver increases rather than decreases.

Appendix K

MUMPS copyright

This copyright must be attached to the theoretical documentation and/or the Code_Carmel user manual in order to remind the user of the authorship of the product and the conditions of its use.

```

COPYRIGHT (c) 1996-2003 P. R. Amestoy, I. S. Duff, J. Koster,
J.-Y. L'Excellent

CERFACS , Toulouse (France) (http://www.cerfacs.fr)
ENSEEIH-IRIT, Toulouse (France) (http://www.enseeiht.fr)
INRIA (France) (http://www.inria.fr)
PARALLAB , Bergen (Norway) (http://www.parallab.uib.no)
All rights reserved.

Your use or distribution of the package implies that you agree
with this License. Up-to-date copies of the MUMPS package can be
obtained from the Web page http://www.enseeiht.fr/apo/MUMPS/
This package is provided to you free of charge. It was
initially based on public domain software developed during
the European Esprit IV project PARASOL (1996-1999).
THIS MATERIAL IS PROVIDED AS IS, WITH ABSOLUTELY NO WARRANTY
EXPRESSED OR IMPLIED. ANY USE IS AT YOUR OWN RISK.

Permission is hereby granted to use or copy this
package provided that the Copyright and this License is
retained on all copies and that the package is used
under the same terms and conditions. User documentation
of any code that uses this software should include this
complete Copyright notice and this License.

You can modify this code but, at no time shall the right
or title to all or any part of this package pass to you.
All information relating to any alteration or addition
made to this package for the purposes of extending the
capabilities or enhancing the performance of this package
shall be made available free of charge to the authors for
any purpose.

You shall acknowledge (using references [1] and [2])
the contribution of this package in any publication
of material dependent upon the use of the package.
You shall use reasonable endeavours to notify
the authors of the package of this publication.

[1] P. R. Amestoy, I. S. Duff and J.-Y. L'Excellent (1998),
Multifrontal parallel distributed symmetric and unsymmetric solvers,
in Comput. Methods in Appl. Mech. Eng., 184, 501-520 (2000).
An early version appeared as a Technical Report ENSEEIH-IRIT (1998)
and is available at http://www.enseeiht.fr/apo/MUMPS/.
[2] P. R. Amestoy, I. S. Duff, J. Koster and J.-Y. L'Excellent,
A fully asynchronous multifrontal solver using distributed dynamic
scheduling, SIAM Journal of Matrix Analysis and Applications,
Vol 23, No 1, pp 15-41 (2001).
An early version appeared as a Technical Report ENSEEIH-IRIT,
RT/APO/99/2 (1999) and is available at
http://www.enseeiht.fr/apo/MUMPS/.

None of the text from the Copyright notice up to and
including this line shall be removed or altered in any way.

```

Figure K.1: MUMPS COPYRIGHT statement.

Appendix L

Moving from real element to reference element

L.1 Case of the tetrahedron

The linear tetrahedral element is defined by 4 nodes, 6 edges and 4 facets (see Figure L.1). Approximation functions are considered on the nodes (n_i , $i = 1, 4$) of the element for scalar unknowns and on the edges (a_i , $i = 1, 6$) for vector unknowns.

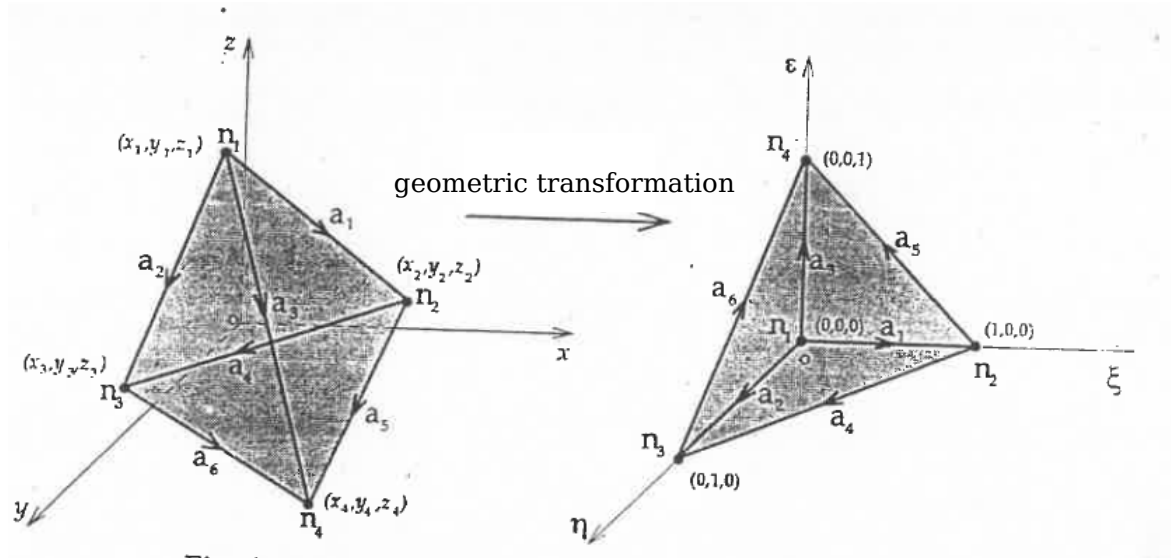


Figure L.1: Linear tetrahedral element and its reference element.

To more easily express the approximation functions, the real geometric element is reduced to a reference element. This is achieved by bijective transformation of the Oxyz coordinate system into an Oξηε reference system. Using this transformation allows the real element to be configured in the reference system.

L.2 Nodal approximation function

With configuration of the real element in the reference coordinate system, the approximation functions ($\tilde{\lambda}_i$, $i = 1, 4$) are given in Table L.1:

Node i	Approximation functions $\hat{\lambda}_i$
1	$1 - \xi - \eta - \varepsilon$
2	ξ
3	η
4	ε

Table L.1: Nodal approximation functions for a reference element

L.3 Edge approximation functions

For the reference element, the edge approximation functions (\hat{w}_k , $k = 1, 6$) are presented in Table L.2. An edge k is identified by two nodes i and j :

Edge k	Nodes i - j	Approximation functions \hat{w}_k		
		ξ	η	ε
1	1 - 2	$1 - \eta - \varepsilon$	ξ	ξ
2	1 - 3	η	$1 - \xi - \varepsilon$	η
3	1 - 4	ξ	ξ	$1 - \xi - \eta$
4	2 - 3	$-\eta$	ξ	0
5	2 - 4	$-\varepsilon$	0	ξ
6	3 - 4	0	$-\varepsilon$	η

Table L.2: Edge approximation functions for a reference element

L.4 Transformation of derivatives

The directional derivatives of a scalar function u defined in the real and reference coordinate system are connected by the following matrix expression:

$$\mathbf{grad}_{\xi\eta\varepsilon} u = \mathbf{J} \mathbf{grad}_{xyz} u \quad \mathbf{grad}_{xyz} u = \mathbf{J}^{-1} \mathbf{grad}_{\xi\eta\varepsilon} u \quad (\text{L.1})$$

Matrix \mathbf{J} is called the Jacobian matrix of the element and is defined as follows:

$$\mathbf{J} = \mathbf{grad} \hat{\lambda} [x, y, z] = \begin{bmatrix} -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_4 & y_4 & z_4 \end{bmatrix} \quad (\text{L.2})$$

It can also be shown that the curl of a vector function \mathbf{u} in the Oxyz coordinate system is connected to that defined in the $O\xi\eta\varepsilon$ coordinate system by:

$$\mathbf{rot}_{xyz} u = \frac{1}{\det \mathbf{J}} \mathbf{J}^T \mathbf{rot}_{\xi\eta\varepsilon} u \quad (\text{L.3})$$

L.5 Transformation of integrals

The volume integral of a function $f(x, y, z)$ on the real element E is reduced to an integral on the reference element \hat{E} by the following relation:

$$\int_E f(x, y, z) dx dy dz = \int_{\hat{E}} f(\xi, \eta, \varepsilon) |\det \mathbf{J}| d\xi d\eta d\varepsilon \quad (\text{L.4})$$

with $\det \mathbf{J}$ the matrix determinant \mathbf{J} .

Appendix M

Add-ins for force and torque calculation

M.1 Maxwell stress tensor

M.1.1 General case

The discretisation of the Maxwell formula (16.2) for force calculation is obtained by replacing the surface integral with a finite sum on the “ N_e ” surface elements. The magnetic field \mathbf{H} is given by its approximation on the elements concerned $\mathbf{H}_e (h_x^e, h_y^e, h_z^e)$ and the outgoing normal for each element is expressed by $\mathbf{n}_e (n_x^e, n_y^e, n_z^e)$. By denoting Γ'^e the surface of the element, we obtain the following relations:

$$\mathbf{F} = \mu_0 \sum_{e=1}^{N_e} \Gamma'^e \left((\mathbf{H}_e \cdot \mathbf{n}_e) \mathbf{H}_e - \frac{1}{2} |\mathbf{H}_e|^2 \mathbf{n}_e \right) \quad (\text{M.1})$$

From this expression we can deduce the components of force $\mathbf{F}^T (F_x F_y F_z)$:

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \mu_0 \sum_{e=1}^{N_e} \Gamma'^e \left((H_x^e n_x^e + H_y^e n_y^e + H_z^e n_z^e) \begin{bmatrix} H_x^e \\ H_y^e \\ H_z^e \end{bmatrix} - \frac{1}{2} (H_x^{e2} + H_y^{e2} + H_z^{e2}) \begin{bmatrix} n_x^e \\ n_y^e \\ n_z^e \end{bmatrix} \right) \quad (\text{M.2})$$

This equation can be rewritten as:

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \mu_0 \sum_{e=1}^{N_e} \Gamma'^e \begin{bmatrix} \frac{1}{2} (H_x^{e2} - H_y^{e2} - H_z^{e2}) n_x^e + H_x^e H_y^e n_y^e + H_x^e H_z^e n_z^e \\ \frac{1}{2} (H_y^{e2} - H_x^{e2} - H_z^{e2}) n_y^e + H_x^e H_y^e n_x^e + H_y^e H_z^e n_z^e \\ \frac{1}{2} (H_z^{e2} - H_x^{e2} - H_y^{e2}) n_z^e + H_x^e H_z^e n_x^e + H_y^e H_z^e n_y^e \end{bmatrix} \quad (\text{M.3})$$

By applying to each component of the force, it is possible to reduce it to matrix form. As an example, we give the calculations to express the component F_x :

$$F_x = \mu_0 \sum_{e=1}^{N_e} \Gamma'^e \mathbf{H}^{eT} \frac{1}{2} \begin{bmatrix} n_x^e & 0 & 0 \\ 0 & -n_x^e & 0 \\ 0 & 0 & -n_x^e \end{bmatrix} \mathbf{H}^e + \mathbf{H}^{eT} \begin{bmatrix} 0 & 0 & 0 \\ n_y^e & 0 & 0 \\ n_z^e & 0 & 0 \end{bmatrix} \mathbf{H}^e \quad (\text{M.4})$$

Finally, we obtain the relation that corresponds to (16.4):

$$F_x = \mu_0 \sum_{e=1}^{N_e} \Gamma'^e \mathbf{H}^{eT} \frac{1}{2} \begin{bmatrix} n_x^e & 0 & 0 \\ n_y^e & -n_x^e & 0 \\ n_z^e & 0 & -n_x^e \end{bmatrix} \mathbf{H}^e = \mu_0 \sum_{e=1}^{N_e} \Gamma'^e \mathbf{H}^{eT} \mathbf{M}_x \mathbf{H}^e \quad (\text{M.5})$$

M.1.2 Two-dimensional case

M.2 Virtual work method

M.2.1 Derivative of the magnetic energy (vector potential formulation)

The value of the force or torque is obtained by differentiating, depending on the direction of displacement s , the magnetic energy at constant flux. Discretisation is obtained by a sum on the deformed elements with the use of the flows of the vector potential c_a^e :

$$F_s = -\frac{1}{2} \sum_{e=1}^{N_e} c_a^{eT} \partial_s \mathbf{S}_a^e c_a^e \quad (\text{M.6})$$

In the reference frame, matrix \mathbf{S}_a^e is written (see annex L):

$$\mathbf{S}_a^e = \int_{\hat{\mathcal{D}}^e} \frac{1}{\mu_0} \left(\frac{1}{\det \mathbf{J}} \mathbf{rot} \hat{w}^T \mathbf{J} \right) \cdot \left(\mathbf{J}^T \mathbf{rot} \hat{w} \frac{1}{\det \mathbf{J}} \right) |\det \mathbf{J}| d\hat{v} \quad (\text{M.7})$$

with \hat{w} the edge approximation functions in the reference frame.

By differentiating this matrix with respect to s , we obtain:

$$\begin{aligned} \partial_s S_a^e = & \frac{\text{sign}(\det \mathbf{J})}{\mu_0} \int_{\hat{\mathcal{D}}^e} (\mathbf{rot} \hat{w}^T \partial_s \mathbf{J}) \cdot (\mathbf{J}^T \mathbf{rot} \hat{w}) \frac{1}{\det \mathbf{J}} d\hat{v} \\ & + \frac{\text{sign}(\det \mathbf{J})}{\mu_0} \int_{\hat{\mathcal{D}}^e} (\mathbf{rot} \hat{w}^T \mathbf{J}) \cdot (\partial_s \mathbf{J}^T \mathbf{rot} \hat{w}) \frac{1}{\det \mathbf{J}} d\hat{v} \\ & + \frac{\text{sign}(\det \mathbf{J})}{\mu_0} \int_{\hat{\mathcal{D}}^e} (\mathbf{rot} \hat{w}^T \mathbf{J}) \cdot (\mathbf{J}^T \mathbf{rot} \hat{w}) \partial_s \left(\frac{1}{\det \mathbf{J}} \right) d\hat{v} \quad (\text{M.8}) \end{aligned}$$

This expression can also be written:

$$\begin{aligned} \partial_s S_a^e = & \frac{\text{sign}(\det \mathbf{J})}{\mu_0} \int_{\hat{\mathcal{D}}^e} (\mathbf{rot} \hat{w}^T \mathbf{J}) \left(\frac{1}{\det \mathbf{J}} \mathbf{J}^{-1} \partial_s \mathbf{J} + \partial_s \mathbf{J}^T \mathbf{J}^{-1} \frac{1}{\det \mathbf{J}} \right) (\mathbf{J}^T \mathbf{rot} \hat{w}) d\hat{v} \\ & + \frac{\text{sign}(\det \mathbf{J})}{\mu_0} \int_{\hat{\mathcal{D}}^e} (\mathbf{rot} \hat{w}^T \partial_s \mathbf{J}) \cdot (\mathbf{J}^T \mathbf{rot} \hat{w}) \frac{-1 \det \mathbf{J}}{(\det \mathbf{J})^2} d\hat{v} \quad (\text{M.9}) \end{aligned}$$

The derivative of the determinant of \mathbf{J} is easily obtained by taking:

$$\mathbf{J}^{-1} = \frac{1}{\det \mathbf{J}} \mathbf{J}' \quad (\text{M.10})$$

with \mathbf{J}' the transposed matrix of cofactors of \mathbf{J} .

We can thus write:

$$\det \mathbf{J} \mathbf{I} = \mathbf{J}' \mathbf{J} \quad (\text{M.11})$$

with \mathbf{I} the identity matrix.

By using these relations in expression M.9, we obtain:

$$\partial_s S_a^e = \frac{\text{sign}(\det \mathbf{J})}{\mu_0} \int_{\hat{\mathcal{D}}^e} \frac{1}{(\det \mathbf{J})^2} (\mathbf{rot} \hat{w}^T \mathbf{J}) (\mathbf{J}' \partial_s \mathbf{J} + \partial_s \mathbf{J}^T \mathbf{J}'^T - \partial_s \mathbf{J}' \mathbf{J}^T - \mathbf{J}' \partial_s \mathbf{J}^T) (\mathbf{J}^T \mathbf{rot} \hat{w}) d\hat{v} \quad (\text{M.12})$$

After some simplifications, we get the final expression (see relation 16.17) of the derivative of S_a^e defined by the approximation of edge w in the real coordinate system and the derivative of the Jacobian matrix:

$$\partial_s S_a^e = \frac{\text{sign}(\det \mathbf{J})}{\mu_0} \int_{\hat{\mathcal{D}}_e} \mathbf{rot} w^T [(\partial_s \mathbf{J}^T) \mathbf{J}'^T - (\partial_s \mathbf{J}') \mathbf{J}] \mathbf{rot} w d\hat{v} \quad (\text{M.13})$$

M.2.2 Derivative of the magnetic co-energy (scalar potential formulation)

In this case, the force or torque is obtained by differentiation of the magnetic co-energy at constant current. Using the values of the scalar potential at the deformed element level, we obtain the discrete form for the value of the force following displacement s :

$$F_s = \frac{1}{2} \sum_{e=1}^{N_e} \Omega^{eT} \partial_s S_{\Omega}^e \Omega^e \quad (\text{M.14})$$

Matrix S_{Ω}^e is expressed using the nodal approximation functions $\hat{\lambda}$ in the reference frame as follows (see annex L):

$$S_{\Omega}^e = \int_{\hat{\mathcal{D}}_e} \mu_0 \left(\mathbf{grad} \hat{\lambda}^T \mathbf{J}^{-1T} \right) \cdot \left(\mathbf{J}^{-1} \mathbf{grad} \hat{\lambda} \right) |\det \mathbf{J}| d\hat{v} \quad (\text{M.15})$$

By introducing the matrix of transposed cofactors \mathbf{J}' , differentiation with respect to s of the previous expression gives:

$$\begin{aligned} \partial_s S_{\Omega}^e &= \text{sign}(\det \mathbf{J}) \mu_0 \int_{\hat{\mathcal{D}}_e} \left(\mathbf{grad} \hat{\lambda}^T \partial_s \mathbf{J}'^T \right) \cdot \left(\mathbf{J}' \mathbf{grad} \hat{\lambda} \right) \frac{1}{\det \mathbf{J}} d\hat{v} \\ &\quad + \text{sign}(\det \mathbf{J}) \mu_0 \int_{\hat{\mathcal{D}}_e} \left(\mathbf{grad} \hat{\lambda}^T \mathbf{J}'^T \right) \cdot \left(\partial_s \mathbf{J}' \mathbf{grad} \hat{\lambda} \right) \frac{1}{\det \mathbf{J}} d\hat{v} \\ &\quad + \text{sign}(\det \mathbf{J}) \mu_0 \int_{\hat{\mathcal{D}}_e} \left(\mathbf{grad} \hat{\lambda}^T \mathbf{J}'^T \right) \cdot \left(\mathbf{J}' \mathbf{grad} \hat{\lambda} \right) \partial_s \frac{1}{\det \mathbf{J}} d\hat{v} \end{aligned} \quad (\text{M.16})$$

We can rewrite this relationship as:

$$\begin{aligned} \partial_s S_{\Omega}^e &= \\ &\text{sign}(\det \mathbf{J}) \mu_0 \int_{\hat{\mathcal{D}}_e} \left(\mathbf{grad} \hat{\lambda}^T \mathbf{J}'^T \right) \cdot \left(\frac{1}{\det \mathbf{J}} \mathbf{J}'^{-1T} \partial_s \mathbf{J}'^T + \partial_s \mathbf{J}' \mathbf{J}'^{-1} \frac{1}{\det \mathbf{J}} \right) \cdot \left(\mathbf{J}' \mathbf{grad} \hat{\lambda} \right) d\hat{v} \\ &\quad + \text{sign}(\det \mathbf{J}) \mu_0 \int_{\hat{\mathcal{D}}_e} \left(\mathbf{grad} \hat{\lambda}^T \mathbf{J}'^T \right) \cdot \left(\mathbf{J}' \mathbf{grad} \hat{\lambda} \right) \frac{-\partial_s (\det \mathbf{J})}{(\det \mathbf{J})^2} d\hat{v} \end{aligned} \quad (\text{M.17})$$

By replacing the derivative of the determinant with its matrix expression, we obtain (WARNING ERROR IN THE FORMULA):

$$\begin{aligned} \partial_s S_{\Omega}^e &= \\ &\text{sign}(\det \mathbf{J}) \mu_0 \int_{\hat{\mathcal{D}}_e} \left(\mathbf{grad} \hat{\lambda}^T \mathbf{J}^{-1T} \right) \cdot \left(\mathbf{J}^T \partial_s \mathbf{J}'^T + \partial_s \mathbf{J}' \mathbf{J} - \partial_s \mathbf{J}' \mathbf{J} - \mathbf{J}' \partial_s \mathbf{J} \right) \left(\mathbf{J}^{-1} \mathbf{grad} \hat{\lambda} \right) d\hat{v} \end{aligned} \quad (\text{M.18})$$

Finally, using the nodal approximation functions λ defined in the real coordinate system, we return to the following expression (see relation 16.22):

$$\partial_s S_\Omega^e = \text{sign}(\det \mathbf{J}) \mu_0 \int_{\hat{\mathcal{D}}^e} (\mathbf{grad} \hat{\lambda}^T) (\mathbf{J}^T \partial_s \mathbf{J}'^T - \mathbf{J}' \partial_s \mathbf{J}) (\mathbf{grad} \hat{\lambda}) d\hat{v} \quad (\text{M.19})$$

For the two cases presented, the calculation of force or torque depends on the evaluation of the same bracketed expression in relations [M.13](#) and [M.19](#), in other words:

$$(\mathbf{J}^T \partial_s \mathbf{J}'^T - \mathbf{J}' \partial_s \mathbf{J})$$

In Chapter [16](#) on the calculation of forces and torque, we explained the differentiation of the Jacobian matrix $\partial_s \mathbf{J}$. In the next section, we will give the procedure for calculating the derivative of matrix \mathbf{J}' .

M.2.3 Calculation of the derivative of matrix \mathbf{J}'

The Jacobian matrix is given as a function of the coordinates of the element¹, in the Cartesian coordinate system $((x_i, y_i, z_i), i = 1, 4)$, by the following relation:

$$\mathbf{J} = \mathbf{grad} \hat{\lambda} \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_4 & y_4 & z_4 \end{bmatrix} = \begin{bmatrix} x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \\ x_4 - x_1 & y_4 - y_1 & z_4 - z_1 \end{bmatrix} \quad (\text{M.20})$$

By defining vectors Δx , Δy and Δz as follows:

$$\Delta x = \begin{bmatrix} x_2 - x_1 \\ x_3 - x_1 \\ x_4 - x_1 \end{bmatrix} \quad \Delta y = \begin{bmatrix} y_2 - y_1 \\ y_3 - y_1 \\ y_4 - y_1 \end{bmatrix} \quad \Delta z = \begin{bmatrix} z_2 - z_1 \\ z_3 - z_1 \\ z_4 - z_1 \end{bmatrix} \quad (\text{M.21})$$

We can write matrix \mathbf{J} and its derivative $\partial_s \mathbf{J}$ in a compact form:

$$\mathbf{J} = |\Delta x \ \Delta y \ \Delta z| \quad \partial_s \mathbf{J} = |\partial_s \Delta x \ \partial_s \Delta y \ \partial_s \Delta z| \quad (\text{M.22})$$

Using this notation, matrix $\mathbf{J}' = (\det \mathbf{J}) \mathbf{J}'$ and its derivative \mathbf{J}' are written:

$$\mathbf{J}' = |\Delta y \times \Delta z \ \Delta z \times \Delta x \ \Delta x \times \Delta y| \quad (\text{M.23})$$

$$\partial_s \mathbf{J}' = |\partial_s \Delta y \times \Delta z + \Delta y \times \partial_s \Delta z \ \partial_s \Delta z \times \Delta x + \Delta z \times \partial_s \Delta x \ \partial_s \Delta x \times \Delta y + \Delta x \times \partial_s \Delta y| \quad (\text{M.24})$$

From the relation [M.24](#) we can deduce the derivatives in the directions $s = (x, y, z)$:

$$\partial_s \mathbf{J}' = \begin{cases} |0 & \Delta z \times \partial_x \Delta x & \partial_x \Delta x \times \Delta y| & s = x \\ |\partial_y \Delta y \times \Delta z & 0 & \Delta x \times \partial_y \Delta y| & s = y \\ |\Delta y \times \partial_z \Delta z & \partial_z \Delta z \times \Delta x & 0| & s = z \end{cases} \quad (\text{M.25})$$

for the calculation of the torque, we take $s = \theta$, which gives:

$$\partial_\theta \mathbf{J}' = |\partial_\theta \Delta y \times \Delta z \ \Delta z \times \partial_\theta \Delta x \ \partial_\theta \Delta x \times \Delta y + \Delta x \times \partial_\theta \Delta y| \quad (\text{M.26})$$

The value of the derivatives $\partial_x \Delta x$, $\partial_y \Delta y$, $\partial_z \Delta z$, $\partial_\theta \Delta x$ and $\partial_\theta \Delta y$ defined in the last two equations, are obtained using Table [16.1](#).

M.2.4 Two-dimensional case

¹case of a tetrahedron

Appendix N

Development using orthogonal polynomials

N.1 Généralités

Let $f(x) : \mathbb{R}^m \rightarrow \mathbb{R}^n$ be a regular function (more precisely, we assume that $f(x)$ is $\mathcal{C}^\infty[-1, 1]$) that can be represented as a linear combination of functional $\{\psi_j\}_j$ of \mathbb{R}^n , as follows:

$$f(x) \approx \sum_j u_j \psi_j(x) \quad , \quad u_j \in \mathbb{R} \quad (\text{N.1})$$

Spectral methods are based on this notation by considering as functionals ψ_j polynomials orthogonal with respect to the weight function w , in other words, the ψ_j verify:

$$\int \psi_i(x) \psi_j(x) w(x) = c_i \delta_{ij} \quad (\text{N.2})$$

where $c_i \in \mathbb{R}$ and δ_{ij} is the Kronecker symbol.

The choice of a family of orthogonal polynomials among all existing ones is an essential issue in spectral approaches. For example, it is well known that Fourier bases are suitable for the development of periodic functions. They offer an optimal (exponential) convergence rate when f and its derivatives are periodic. If f or its derivatives are not periodic then it is more appropriate to use non-periodic bases such as orthogonal polynomials. More generally, the choice of the polynomial basis can be justified by the Sturm-Liouville problem. The Sturm-Liouville problem has the following form:

$$-\partial_x(p(x)\partial_x v) + q(x)v = \lambda w(x)v, \quad \forall x \in]a, b[\quad (\text{N.3})$$

This implies that the eigenvalues λ are real and the associated eigenfunctions $\{v(x)_j\}_j = 0^\infty$ are orthogonal. More particularly, the v_j form a basis of $L_w^2(a, b)$ and we can thus represent any square-integrable function (for the weight function w). In the special case where the interval $[a, b]$ is equal to $[-1, 1]$, the eigenfunctions of the Sturm-Liouville problem define the Jacobi polynomials $P_k^{\alpha, \beta}$ given by the following recurrence:

$$P^{(\alpha, \beta)} = \dots \quad (\text{N.4})$$

The Jacobi polynomials $P^{\alpha, \beta}$ thus define a basis of $L_w^2(-1, 1)$. Thus any square-integrable function f can be expanded as:

$$f(x) = \sum_{k=0}^{\infty} \tilde{f}_k P_k^{\alpha, \beta}(x) \quad (\text{N.5})$$

where the coefficients \tilde{f} are given by (consequence of the orthogonality of the polynomials):

$$\tilde{f}_k = \frac{\int f(x) P_k^{\alpha, \beta}(x) dx}{\|P_k^{\alpha, \beta}(x)\|^2} \quad (\text{N.6})$$

The representation of the derivative of f is a little more delicate because the derivatives of the Jacobi polynomials are not specific functions of the Sturm-Liouville problem (note that the problem does not arise with the Fourier basis because the basis of the differentiation operator is the same as the Fourier basis). In the following, we will look at two families of orthogonal polynomials which are Legendre polynomials, obtained from Jacobi polynomials by taking $\alpha = \beta = 0$, and Chebyshev polynomials obtained in the case where $\alpha = \beta = -1/2$. For these two families of polynomials, we will present the formulas that allow us to calculate them easily, as well as the formulas for the polynomial development of derivatives.

N.2 Legendre polynomials

Legendre polynomials are orthogonal with respect to the weight function $w = 1$. They verify the following recurrence relation:

$$L_{k+1}(x) = x \frac{2k+1}{k+1} L_k(x) - \frac{k}{k+1} L_{k-1}(x), \quad \text{avec } L_0(x) = 1 \text{ et } L_1(x) = x \quad (\text{N.7})$$

the squared norm of polynomial $L_k(x)$ is $\frac{2}{2k+1}$. The derivatives of Legendre polynomials verify the following recurrence:

$$L_k(x) = \frac{L_{k+1}^{(1)}(x)}{(2k+1)} - \frac{L_{k-1}^{(1)}(x)}{(2k+1)}, \quad \text{avec } L_0^{(1)} = 0 \text{ et } L_{-1}^{(1)} = 0 \quad (\text{N.8})$$

Now, we seek to identify the development coefficients in the Legendre basis of the derivative of f in the form:

$$f^{(1)}(x) = \sum_{k=0}^{\infty} \tilde{f}_k^{(1)} L_k(x) \quad (\text{N.9})$$

We start by substituting the relation (N.8) in (N.9), and then by considering the derivative of relation (N.9) we show the following equality:

$$\tilde{f}_k = \frac{\tilde{f}_{k-1}^{(1)}}{2k-1} - \frac{\tilde{f}_{k+1}^{(1)}}{2k+3}, \quad \forall k \geq 1 \quad (\text{N.10})$$

And by recurrence, we have:

$$\tilde{f}_k^{(1)} = (2k+1) \left(\tilde{f}_{k+1} + \frac{\tilde{f}_{k+2}^{(1)}}{2k+5} \right), \quad \text{avec } \tilde{f}_N^{(1)} = \tilde{f}_{N+1}^{(1)} = 0 \quad (\text{N.11})$$

If we develop this recurrence, we show the following relationship:

$$\tilde{f}_k^{(1)} = (2k+1) \left(\tilde{f}_{k+1} + \tilde{f}_{k+3} + \tilde{f}_{k+5} + \dots \right) \quad (\text{N.12})$$

$$= (2k+1) D_k^t \cdot \tilde{F} \quad (\text{N.13})$$

More generally, we write:

$$\tilde{F}^{(1)} = \mathbf{D} \tilde{F} \quad (\text{N.14})$$

with $\tilde{F}^{(1)}$ the vector containing the development coefficients of $f^{(1)}$, \tilde{F} the vector containing the development coefficient of f and \mathbf{D} the coupling matrix.

Example:

We assume that $N=5$, i.e. that:

$$f(x) = \tilde{f}_0 L_0(x) + \tilde{f}_1 L_1(x) + \tilde{f}_2 L_2(x) + \tilde{f}_3 L_3(x) + \tilde{f}_4 L_4(x) + \tilde{f}_5 L_5(x)$$

We seek its derivative in the form:

$$f(x)^{(1)} = \tilde{f}_0^{(1)} L_0(x) + \tilde{f}_1^{(1)} L_1(x) + \tilde{f}_2^{(1)} L_2(x) + \tilde{f}_3^{(1)} L_3(x) + \tilde{f}_4^{(1)} L_4(x)$$

Then passing (by matrix \mathbf{D}) from coefficients \tilde{f}_k to coefficients $\tilde{f}_k^{(1)}$ is thus written:

$$\begin{pmatrix} \tilde{f}_0^{(1)} \\ \tilde{f}_1^{(1)} \\ \tilde{f}_2^{(1)} \\ \tilde{f}_3^{(1)} \\ \tilde{f}_4^{(1)} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 3 & 0 & 3 & 0 \\ 0 & 0 & 0 & 5 & 0 & 5 \\ 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 9 \end{pmatrix} \begin{pmatrix} \tilde{f}_0 \\ \tilde{f}_1 \\ \tilde{f}_2 \\ \tilde{f}_3 \\ \tilde{f}_4 \\ \tilde{f}_5 \end{pmatrix} \quad (\text{N.15})$$

and the reverse (by matrix $\hat{\mathbf{D}}$) (from the coefficients of the derivative to those of f) is written:

$$\begin{pmatrix} \tilde{f}_0 \\ \tilde{f}_1 \\ \tilde{f}_2 \\ \tilde{f}_3 \\ \tilde{f}_4 \\ \tilde{f}_5 \end{pmatrix} = \begin{pmatrix} 0 & \frac{-1}{3} & 0 & 0 & 0 & 0 \\ 1 & 0 & \frac{-1}{5} & 0 & 0 & 0 \\ 0 & \frac{1}{3} & 0 & \frac{-1}{7} & 0 & 0 \\ 0 & 0 & \frac{1}{5} & 0 & \frac{-1}{9} & 0 \\ 0 & 0 & 0 & \frac{-1}{7} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{9} \end{pmatrix} \begin{pmatrix} \tilde{f}_0^{(1)} \\ \tilde{f}_1^{(1)} \\ \tilde{f}_2^{(1)} \\ \tilde{f}_3^{(1)} \\ \tilde{f}_4^{(1)} \\ \tilde{f}_5^{(1)} \end{pmatrix} \quad (\text{N.16})$$

We note here that:

$$\mathbf{D}\hat{\mathbf{D}} = \mathbf{I} \quad (\text{N.17})$$

N.3 Chebyshev polynomials

The other family of polynomials studied here is that of Chebyshev. They are obtained from Jacobi polynomials by considering $\alpha = \beta = -1/2$. These polynomials have a direct relationship with the Fourier transformation as they are given by:

$$T_k(x) = \cos(k \cos^{-1}(x)) \quad (\text{N.18})$$

Chebyshev polynomials can also be obtained by the following recurrence:

$$T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x), \text{ avec } T_0 = 1 \text{ et } T_1 = x \quad (\text{N.19})$$

The derivatives $T_k^{(1)}$ of Chebyshev polynomials verify the following recurrence:

$$2T_k(x) = \frac{T_{k+1}^{(1)}(x)}{k+1} - \frac{T_{k-1}^{(1)}(x)}{k-1} \quad (\text{N.20})$$

The norm L^2 of these polynomials is:

$$\|T_k\|_w^2 = c_k \frac{\pi}{2} \quad \text{où } c_0 = 2 \text{ et } c_k = 1 \text{ si } k \geq 1$$

The relation between the development coefficients in the Chebyshev polynomial basis of function f and its derivative $f^{(1)}$ is written:

$$c_k \tilde{f}_k^{(1)} = \tilde{f}_{k+2}^{(1)} + 2(k+1) \tilde{f}_{k+1}, \quad \forall k \geq 0 \quad (\text{N.21})$$

In the same way as for Legendre polynomials, by developing the recurrence relation we show that:

$$\tilde{f}_k^{(1)} = \frac{2}{c_k} \sum_{p=k+1, p+k \text{ impair}}^N p \tilde{f}_p \quad (\text{N.22})$$

$$D = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 & \dots & \dots \\ 0 & 0 & 3 & 0 & 3 & 0 & \dots & \dots \\ \vdots & \ddots & 0 & 5 & 0 & 5 & & \\ \vdots & \dots & \ddots & 0 & 7 & 0 & 7 & \end{pmatrix} \quad (\text{N.23})$$

Chebyshev polynomials are closely related to Fourier decomposition. When considering the change in variable $x = \cos(\theta)$, Chebyshev polynomials are written $T_n(x) = \cos(n\theta)$. Thus, the spectral coefficients of the series:

$$f(x) = \sum_{n=0}^{\infty} a_n T_n(x)$$

are identical to those of the series:

$$f(\cos(\theta)) = \sum_{n=0}^{\infty} a_n \cos(n\theta)$$

This observation is highly significant. It indicates that the Chebyshev series for not only periodic functions converges at the same rate as the Fourier series for periodic functions (exponential convergence).

N.4 Development using the Fourier basis

Any periodic function of period T (of pulse $\omega = 2\pi/T$) can be decomposed as the sum of trigonometric polynomials $\Psi_n(t) = e^{j\omega n t}$ such as:

$$f(t) = \sum_{n=-\infty}^{n=+\infty} c_n \Psi_n(t) \quad (\text{N.24})$$

As trigonometric polynomials $\{\Psi_n(t)\}_n$ form an orthonormal basis of \mathbb{C} on the interval $[0, T]$, hence the Fourier coefficients c_n are given by:

$$c_n = \langle f(t), \Psi_n(t) \rangle = \frac{1}{T} \int_0^T f(t) \bar{\Psi}_n(t) \quad (\text{N.25})$$

For real functions, Fourier series development is reduced to a development in a trigonometric series as follows:

$$f(t) = \frac{a_0}{2} + \sum_{n \in \mathbb{N}^*} a_n \cos(n\omega t) + b_n \sin(n\omega t) \quad (\text{N.26})$$

Coefficients a_n and b_n are directly related to coefficients c_n by the following relations:

$$\begin{cases} a_0 = 2c_0 \\ a_n = c_n + c_{-n}, & \forall n \in \mathbb{N}^* \\ b_n = j(c_n - c_{-n}), & \forall n \in \mathbb{N}^* \end{cases} \quad (\text{N.27})$$

Below, we will work with the trigonometric form with which we will associate the Hilbert basis $\{1, \cos(n\omega t), \sin(n\omega t)\}$ of $L^2([0, T])$. In this case, we write:

$$f(t) = \sum_{k=0}^{2n_h} \tilde{f}_k \Psi_k(t) \quad (\text{N.28})$$

with $\Psi_k(t) \in \{1, \cos(\omega t), \sin(\omega t), \cos(2\omega t), \sin(2\omega t), \dots\}$ and $\tilde{f}_k = \frac{2}{T} \int_0^T f(t) \Psi_k(t) dt$

We will now look at the relation between the Fourier series development of $f(t)$ and that of the derivative of $f(t)$ which we will assume is written in the following form:

$$f^{(1)}(t) = \sum_{k=0}^{2n_h} \bar{f}_k \Psi_k^{(1)}(t) \quad (\text{N.29})$$

By term-by-term identification, we show the following relation:

$$\begin{cases} \bar{f}_0 = 0 \\ \bar{f}_k = -[\frac{k+1}{2}] \omega \tilde{f}_{k+1} & \text{Si } k \text{ est impair} \\ \bar{f}_k = [\frac{k}{2}] \omega \tilde{f}_{k-1} & \text{Si } k \text{ est pair} \end{cases} \quad (\text{N.30})$$

By analogy with orthogonal polynomials, we define the differentiation matrix \mathbf{D} allowing us to link the coefficients of $f^{(1)}(t)$ directly to those of $f(t)$. Denoting \mathbf{F} the vector containing the development coefficients in series of $f(t)$ and $\bar{\mathbf{F}}$ those of $f^{(1)}(t)$, we thus write:

$$\bar{\mathbf{F}} = \mathbf{D} \mathbf{F} \quad (\text{N.31})$$

We will also denote $\hat{\mathbf{D}}$ the matrix such that (\mathbf{I} is the identity matrix):

$$\mathbf{D} \hat{\mathbf{D}} = \mathbf{I} \quad (\text{N.32})$$

Exemple:

Either $f(x)$ in the form:

$$f(x) = \tilde{f}_0 + \tilde{f}_1 \cos(\omega t) + \tilde{f}_2 \sin(\omega t) + \tilde{f}_3 \cos(2\omega t) + \tilde{f}_4 \sin(2\omega t) + \tilde{f}_5 \cos(3\omega t) + \tilde{f}_6 \sin(3\omega t)$$

so by differentiation we have $f^{(1)}(t)$ which is written:

$$f^{(1)}(x) = -\omega \tilde{f}_1 \sin(\omega t) + \omega \tilde{f}_2 \cos(\omega t) - 2\omega \tilde{f}_3 \sin(2\omega t) + 2\omega \tilde{f}_4 \cos(2\omega t) - 3\omega \tilde{f}_5 \sin(3\omega t) + 3\omega \tilde{f}_6 \cos(3\omega t)$$

By identification with (N.29), we show that the relation between coefficients \bar{f}_k and \tilde{f}_k :

$$\begin{pmatrix} \bar{f}_0 \\ \bar{f}_1 \\ \bar{f}_2 \\ \bar{f}_3 \\ \bar{f}_4 \\ \bar{f}_5 \\ \bar{f}_6 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \omega & 0 & 0 & 0 & 0 \\ 0 & -\omega & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2\omega & 0 & 0 \\ 0 & 0 & 0 & -2\omega & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3\omega \\ 0 & 0 & 0 & 0 & 0 & 3\omega & 0 \end{pmatrix} \begin{pmatrix} \tilde{f}_0 \\ \tilde{f}_1 \\ \tilde{f}_2 \\ \tilde{f}_3 \\ \tilde{f}_4 \\ \tilde{f}_5 \\ \tilde{f}_6 \end{pmatrix} \quad (\text{N.33})$$

In this case, matrix $\hat{\mathbf{D}}$ is written:

$$\hat{\mathbf{D}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/\omega & 0 & 0 & 0 & 0 \\ 0 & -1/\omega & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2\omega & 0 & 0 \\ 0 & 0 & 0 & -1/2\omega & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/3\omega \\ 0 & 0 & 0 & 0 & 0 & 1/3\omega & 0 \end{pmatrix} \quad (\text{N.34})$$

Appendix O

Kronecker product

The tensor product of a matrix, denoted \otimes , of size $n_1 \times m_1$ with a matrix of size $n_2 \times m_2$ is a matrix of size $n_1 n_2 \times m_1 m_2$. Example:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \otimes \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11} * \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} & a_{12} * \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \\ a_{21} * \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} & a_{22} * \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \end{pmatrix} \quad (\text{O.1})$$

The Kronecker product has the following properties:

- 1– $\mathbf{A} \otimes (\mathbf{B} + \lambda \mathbf{C}) = (\mathbf{A} \otimes \mathbf{B}) + \lambda (\mathbf{A} \otimes \mathbf{C})$
- 2– $\mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}) = (\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C}$
- 3– $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD})$
- 4– $(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}$
- 5– $(\mathbf{A} \otimes \mathbf{B})^t = \mathbf{A}^t \otimes \mathbf{B}^t$
- 6– $Tr(\mathbf{A} \otimes \mathbf{B}) = Tr(\mathbf{A})Tr(\mathbf{B})$
- 7– $(\mathbf{A} \otimes \mathbf{B})\mathbf{Y} = vec(\mathbf{BCB}^t)$ (see [\[Beddek 2012\]](#) for the definition of the operator *vec* and the matrix \mathbf{C}).

Appendix P

Hadamard product

The Hadamard product of two matrices, denoted \circ , is written:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \circ \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} & a_{12}b_{12} \\ a_{21}b_{21} & a_{22}b_{22} \end{pmatrix} \quad (\text{P.1})$$

The Hadamard product has the following properties:

- 1– Commutative: $\mathbf{A} \circ \mathbf{B} = \mathbf{B} \circ \mathbf{A}$
- 2– Associative: $\mathbf{A} \circ (\mathbf{B} \circ \mathbf{C}) = (\mathbf{A} \circ \mathbf{B}) \circ \mathbf{C}$
- 3– Distributive on “+”: $\mathbf{A} \circ (\mathbf{B} + \mathbf{C}) = (\mathbf{A} \circ \mathbf{B}) + (\mathbf{A} \circ \mathbf{C})$

We define the Hadamard product per block by:

$$\begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 \\ \mathbf{A}_3 & \mathbf{A}_4 \end{pmatrix} \circ \begin{pmatrix} \mathbf{B}_1 & \mathbf{B}_2 \\ \mathbf{B}_3 & \mathbf{B}_4 \end{pmatrix} = \begin{pmatrix} \mathbf{A}_1 \cdot \mathbf{B}_1 & \mathbf{A}_2 \cdot \mathbf{B}_2 \\ \mathbf{A}_3 \cdot \mathbf{B}_3 & \mathbf{A}_4 \cdot \mathbf{B}_4 \end{pmatrix} \quad (\text{P.2})$$

with \cdot the usual matrix product.

Appendix Q

Modified Gauss quadrature

Gauss quadrature methods are based on the principle of an approximation of the integral of $f(x)$ by:

$$I_f = \int_a^b f(x)w(x)dx \approx \sum_{i=1}^n w_i f(x_i) \quad (\text{Q.1})$$

The points $x_i \in [a, b]$ and weights w_i are chosen such that the approximation error $E_n = \|I_f - \sum w_i f(x_i)\|$ is minimal. This notation assumes that the function $f(x)$ can be evaluated at each point x_i . However, for sampled functions or those with singularity, this evaluation may not be possible. In these cases, the quadrature is modified by taking points \tilde{x}_i sufficiently close to x_i and where the function $f(x)$ can be evaluated. To reduce the error in this approximation, weights w_i must be re-evaluated such that E_n is minimised. In practice, to estimate the new weights \tilde{w}_i , it is assumed that the initial quadrature method at n points is accurate for polynomials $\{\phi_i\}$ of order $2n - 1$ at most. This results in the resolution of the following minimisation problem (taking $r = 2n - 1$):

$$\min \left[\begin{pmatrix} \int \phi_1(x)w(x) \\ \int \phi_2(x)w(x) \\ \vdots \\ \int \phi_r(x)w(x) \end{pmatrix} - \begin{pmatrix} \phi_1(\tilde{x}_1) & \phi_1(\tilde{x}_2) & \dots & \phi_1(\tilde{x}_n) \\ \phi_2(\tilde{x}_1) & \phi_2(\tilde{x}_2) & \dots & \phi_2(\tilde{x}_n) \\ \vdots & \vdots & & \vdots \\ \phi_r(\tilde{x}_1) & \phi_r(\tilde{x}_2) & \dots & \phi_r(\tilde{x}_n) \end{pmatrix} \begin{pmatrix} \tilde{w}_1 \\ \tilde{w}_2 \\ \vdots \\ \tilde{w}_n \end{pmatrix} \right] \quad (\text{Q.2})$$

Assuming that $\{\phi_i\}$ are the Legendre polynomials (orthonormal with respect to $w(x) = 1$, and $L_1(x) = 1$), the minimisation is rewritten as:

$$\min \left[\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} - \begin{pmatrix} L_1(\tilde{x}_1) & L_1(\tilde{x}_2) & \dots & L_1(\tilde{x}_n) \\ L_2(\tilde{x}_1) & L_2(\tilde{x}_2) & \dots & L_2(\tilde{x}_n) \\ \vdots & \vdots & & \vdots \\ L_r(\tilde{x}_1) & L_r(\tilde{x}_2) & \dots & L_r(\tilde{x}_n) \end{pmatrix} \begin{pmatrix} \tilde{w}_1 \\ \tilde{w}_2 \\ \vdots \\ \tilde{w}_n \end{pmatrix} \right] \quad (\text{Q.3})$$

The new quadrature formula is obviously less accurate than the original. Note that there are more mathematically robust methods for constructing quadrature formulas where quadrature points are imposed. However, these methods are difficult to implement.

Index

Symbols			
α		26	Curie temperature
β		26	current
			current density
			5, 98, 99
A			D
additional losses	258		diamagnetic materials
anisotropy field	253		dimensional analysis
anomalous losses	255, 258		dipole
antiferromagnetic order	251		direct linear solver
ARQS	6, 9		direct methods
attached elements method	151		discrete curl
			discrete differential operators
			discrete divergence
			discrete gradient
			divergence matrix
			domain under study
			dual mesh
B			E
Barkhausen	256		edge
barycentric coordinate method	265		edge function
barycentric coordinates	263, 266		edge tree
barycentric coordinates method	267, 272		electric charge conservation
Betti number	65		electric charge density
Biot-Savart	248		electric field
Bloch wall	252, 254		electric induction
blocked step	151, 157		electric wall
blocked step method	150, 151		electrical anisotropy
boundary	3, 4		electrical conductivity
boundary conditions	4, 6, 14, 106		electrokinetics
branch	57		electroquasistatic model
			Euler's method
			Euler-Poincaré formula
			Eulerian description
			exchange energy
			exchange integral
			exploratory point
C			F
characteristic length	6		facet
circuit equation	56, 194		facet function
circuit graph	59		facet tree
circulation	105		factorisation
co-tree	91		Faraday's law
co-tree matrix	96		ferromagnetic materials
coil density	55, 193		
Compressed Sparse Row	200		
conditioning	226		
conductive domain	3, 8, 11		
conjugate gradient	202, 205		
connectivity table	153		
constitutive relations	11		
continuity conditions	5		
conventional losses	255		
Coulomb gauge	43		
crossing conditions	13		
Crout	213		

ferromagnetic order 251
 first magnetisation curve 254
 fixed point 163, 169
 force 237, 238
 Fröhlich's equation 12
 function space 41, 42

G

gauge 20, 21, 23, 75
 gauge condition 291
 Gauss 142
 geometric transformation 264
 guideline 88

H

hard materials 251
 Harmonic Balance Method 113
 hexahedron 134, 146
 hole 37
 hysteresis losses 255

I

imposed current 27, 29, 33, 34
 imposed flux 29, 31, 33, 34
 imposed magnetic potential difference 30, 31
 imposed magnetomotive force 33, 34
 imposed voltage 28, 32, 34
 incidence matrices 68, 303
 induced current losses 256
 inductance 56
 inductor 3
 inductor current 26
 inductor volume 56, 194
 integral method 150
 integration on a hexahedron 146
 integration on a prism 146
 integration on a rectangle 143
 integration on a tetrahedron 144
 integration on a triangle 143
 interpolation method 151
 iron losses 249
 iterative linear solver 201
 iterative method 202

J

Jacobi 212
 Jacobian matrix 263, 264, 299
 Jacobian matrix method 267, 271

K

K 26, 80
 Kirchhoff's voltage law 57
 Krylov spaces 329

L

Lagrange operators 151
 Lagrangian description 150
 large axis 260

linear algebra 327
 linear system 199
 local flux 245
 loop 57
 low frequency 6

M

macro-element 150
 magnet 3
 magnetic anisotropy 12
 magnetic co-energy 241
 magnetic domain 252
 magnetic energy 240
 magnetic field 4
 magnetic flux 56, 245
 magnetic flux density 4
 magnetic forces 237
 Magnetic losses 255
 magnetic material 249
 magnetic moment 249
 magnetic permeability 12
 magnetic polarisation 250
 magnetic susceptibility 250, 251
 magnetic wall 15
 magnetisation 249
 magnetocrystalline anisotropy energy 252
 magnetodynamics 9, 22, 32, 179, 193
 magnetoquasistatic model 8
 magnetostatic energy 252
 magnetostatics 10, 20, 29, 39, 178
 Marrocco 299
 Marrocco's equation 12
 mass matrix 98
 Maxwell stress tensor 237
 Maxwell's equations 5
 Mesh current 57
 method of mean weighted residuals 44
 minimisation matrix 93
 minimisation method 92
 mortar 151
 motion 149
 motion strip 150
 MUMPS 214, 221

N

N 26, 80
 Newton-Raphson 165, 170
 nodal function 65, 104
 nodal function method 267, 269
 nodal interpolation functions 263
 node 65, 103
 non-conductive domain 3, 8, 11
 non-connectedness 37
 non-linearity 299
 normal trace 106
 number of coils 55
 numerotation 177

- O**
- orbital moment 249
 - orientation 71
 - overlapping 151, 293
 - overlapping method 150, 153
 - overlapping reference elements 154
- P**
- parallelism 199, 201, 202
 - paramagnetic materials 251
 - perfect electrical conductor 15
 - perfect magnetic conductor 14
 - permanent magnets 13
 - pivoting 224
 - preconditioned conjugate gradient 209
 - preconditioning 208
 - prism 131, 269
 - propagation phenomena 6
 - pyramid 137
- Q**
- quasistatic state 9
 - quasistatic states 6
- R**
- rectangle 143
 - relative permeability 250
 - renumerotation 219
 - resistance 56
 - retarded potential solutions 6
 - right member 98
 - rotational field losses 259
 - rotor 150
- S**
- scalar electric potential 18, 23
 - scalar function 106
 - scalar magnetic potential 21, 23
 - scalar potential 43
 - singular matrices 229
 - slipping surface 153
 - small axis 260
 - soft materials 251
- source current density 55, 193
 - source magnetic field 19, 21, 22
 - source scalar potential 18
 - sources 4
 - space harmonics 257
 - spanning tree 59, 60
 - spatial integration 142
 - spin 249, 251
 - stator 150
 - steepest descent 204
 - sub-domain 65
 - surface integrals 46
- T**
- tangential trace 105
 - tension imposée 33
 - tensor product 183
 - test function 44
 - tetrahedron 128, 144, 266
 - time discretisation 123
 - time harmonics 257
 - torque 238, 239
 - transport term 150
 - tree 75
 - tree of the electrical circuit 59
 - triangle 143
- U**
- uniform current density 26
- V**
- vector electric potential 19, 23
 - vector function 104, 105
 - vector magnetic potential 20, 22
 - vector potential 43
 - virtual work 240, 242
 - voltage source 56
 - volume element 65
 - volume function 67, 106
- W**
- Weiss domain 251, 252
 - wound inductor 3, 55