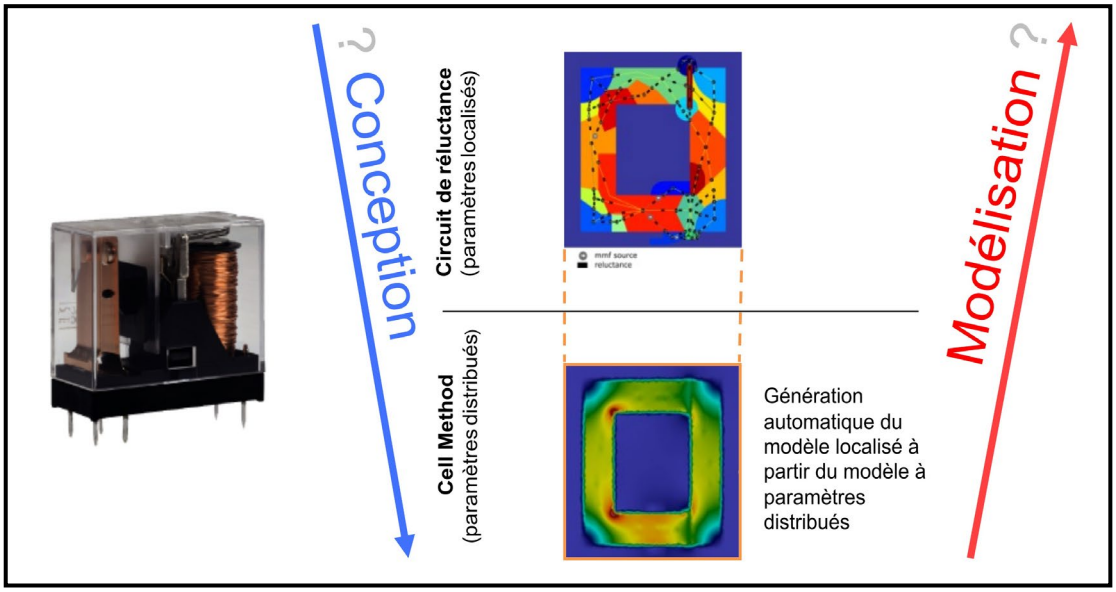
Geometric and energy-based methods for modeling and simulation of multi-physics systems in electrical engineering

Dimitri Tratkanov

Par **Dimitri TRATKANOV**

Méthodes géométriques et énergétiques pour la modélisation et la simulation des systèmes multi-physiques en génie électrique

Thèse présentée pour l'obtention du grade de Docteur de l'UTC



Soutenue le 18 mai 2020
Spécialité : Génie Électrique : Unité de recherche en Mécanique - Laboratoire Roberval (FRE UTC - CNRS 2012)
D2562

Université de Technologie de Compiègne

Thèse

En vue d'obtention du

Doctorat

de l'université de technologie de Compiègne

par **Dimitri Tratkanov**

Méthodes géométriques et énergétiques pour la modélisation et la simulation des systèmes multi-physiques en génie électrique

---

Spécialité : Génie Electrique

École doctorale : Sciences pour l'ingénieur ED71

Unité de Recherche : Laboratoire Roberval FRE UTC-CNRS 2012

Directeur de thèse : Arnaud Hubert

Co-encadrant de thèse : Luis-Alejandro Ospina-Vargas

Compiègne, 18 mai 2020

Geometric and energy-based methods for modeling and simulation of multi-physics systems in electrical engineering

Dimitri Tratkanov

18/05/2020

Contents

1 Introduction and context 15
1.1 Model-based systems engineering 15
1.2 Multi-level modeling 18
1.3 Thesis outline 21
2 Multilevel and multiphysic modelling 23
2.1 Properties and characteristics of models 24
2.1.1 Multiphysic models 24
2.1.2 Second level models 26
2.1.3 Third level models 37
2.2 Multiphysic topological models 40
2.2.1 A few words on the Cell-Method approach 40
2.2.2 Second level models 46
2.2.3 Third level models 50
2.3 Outline of the next chapters 55
3 Simulation Tools for field computation 57
3.1 Historical review of the evolution of topological methods in field computation 58
3.2 Mesh generation 59
3.3 Re-meshing techniques 63
3.4 PyDEC: library and working 70
3.4.1 Simplicial complex representation 70
3.4.2 Algorithm for discrete exterior derivative calculation . . . 72
3.4.3 Algorithm for Hodge star operator calculation 74
3.5 Implementation of cochain method 75
3.5.1 Scalar fields computation 80
3.5.2 Vector fields computation 84
4 From fields to lumped models 89
4.1 Review of the main model reduction methods 90
4.1.1 Proper Orthogonal Decomposition Method 91
4.1.2 Partial element equivalent circuit method 92
4.1.3 Topological PEEC approach. The limits of PEEC approach 97
4.1.4 Conclusion on model reduction methods. 99
4.2 Review of Artificial Intelligence methods 100
4.2.1 Bayesian approach 101
4.2.2 Neural Networks 103
4.2.3 Decision Tree Classifier 106
4.2.4 K-Means clustering 109
4.2.5 Support Vector Machine 111
4.2.6 Basic DBSCAN algorithm 113

4.2.7 Conclusion on AI-based methods relevant for MOR applications. 116
4.3 Automatic generation of lumped models 117
4.3.1 Step 1: Construction of the spatial mesh 119
4.3.2 Step 2: Solving physical equations 120
4.3.3 Step 3: Clustering of homogeneous field regions 123
4.3.4 Step 4: automated generation of a network model 128
4.3.5 Conclusion on the automatic generation of reduced LP model 137
5 Conclusion and perspectives 139
Appendix A Algebraic versus differential formulations 143
Appendix B Glossary of the Algebraic Topology 147
B.1 Boundaries, Coboundaries and the Incidence Matrices 151
B.2 Chains and Cochains Complexes, Boundary and Coboundary Processes 152

List of Figures

1.1 The V-model of the systems engineering process [Fritzson, 2011]. . 16
1.2 Simulation-Driven Product Development view by ARP-4754A [ARP, 2014]. 16
1.3 Stakeholders during systems engineering process [ARP, 2014]. . . 17
1.4 Different levels of modeling for MEMS and electronic devices [Kasper, 2011, Hubert, 2002]. 18
1.5 A coil with core and different models of its behaviour [Brisset, 2007]. 18
1.6 Pareto front of models used for electric motor design. [Brisset, 2007] 19
1.7 Coupling between multi-levels models : *design* and *modeling* views. 20

2.1 Sketch of a spring. Top view : spring at rest ($x = 0$ and $f = 0$). Bottom view : spring with a deformation (with a displacement x when submitted to a force f). 24
2.2 Computation of the elastic energy in the spring of Figure 2.1. . . 25
2.3 Description of a very simple electric circuit and its digraph. 27
2.4 Example of the illustrative simple electric circuit: (a) description of the digraph $\mathcal{G}$; (b) corresponding incidence matrix $\mathbf{A}_e$. 29
2.5 Exemple of the electric circuit: (a) description of the digraph $\mathcal{G}$; (b) Corresponding mesh matrix $\mathbf{M}_e$. 31
2.6 Description of the primal and dual graph of the illustrative electrical example. 34
2.7 Tonti diagram of electrical circuits: two ways to travel it, using $\mathbf{\Phi}$, potentials at nodes (Node analysis) or using $\mathbf{i}$, mesh currents (mesh analysis). 36
2.8 Tonti diagram of electrical circuits with current and voltage sources. 36
2.9 Description of the three main types of equations in the modeling of discrete/lumped physical system. 41
2.10 The general classification of physical variables by E. Tonti. 42
2.11 The four space elements: point, line, surface and volume. 43
2.12 Difference between global variables and densities. 43
2.13 The second classification of physical variables 45
2.14 Association between global variables and space elements of the primal and dual cell complexes, in different physical theories [Ferretti, 2014]. 46
2.15 Tonti's classification diagram of the physical variables in the general case [Ferretti, 2014]. 47
2.16 Tonti diagram for physical problem on discrete mesh (3-complex). 51

2.17 Tonti diagram for physical problem on 3D continuum space and de Rham theory [Tonti, 2013]. 55

3.1 Sequential steps for field computation. 57
3.2 Different possible location of the circumcenter. 60
3.3 Graphic representation of Delaunay condition: (a) this pair of triangles does not meet the Delaunay condition (the circumcircle contains more than three points)[Community of Wikipedia, 2018]; (b) flipping the common edge produces a valid Delaunay triangulation for the four points [Community of Wikipedia, 2018]; (c) the circumscribing circle of every Delaunay triangle is empty [Jonathan Richard Shewchuk, 2018]. 61
3.4 Voronoi diagram of plan sites [Lemaire, 1997]. 62
3.5 Graphic representation of Voronoi properties: (a) the circle $C(v)$ contains no other point of S [Prof. Roberto Tamassia, 1993]; (b) the graph constructed by connecting all vertices in a set S across the edges of their Voronoi polygons is a triangulation of S [Weisstein, Eric W., 1999]. And in the opposite, the dual graph for a Voronoi diagram (in the case of a Euclidean space with point sites) corresponds to the Delaunay triangulation for the same set of points so that the Delaunay triangulations are dual to the Voronoi diagram [Delaunay, 1934]; (c) orthogonality properties of circumcenter Ω of triangle $\triangle ABC$; (d) unbounded regions contain the points on the convex hull of the set S [Prof. Roberto Tamassia, 1993]. 63
3.6 Typical problems of Delaunay triangulation for a given set of points: (a) non convex shapes; internal boundaries; (b) discontinuities in interpolated functions [Jonathan Richard Shewchuk, 2018]. 63
3.7 Typical problems of Delaunay triangulation for a given set of points: (a) in both cases the radius-edge ratio is large; (b) slivers: the only case in which radius-edge ratio ρ is not large [Jonathan Richard Shewchuk, 2018]. 64
3.8 Delaunay-Voronoi tetrahedral mesh: (a) correct dual mesh; (b) non correct dual mesh: points consigned to triangles lying outside the tetrahedra. 65
3.9 (a) Delaunay mesh (b) Voronoi diagram with open (infinite) regions at the boundary (c) Voronoi mesh created by truncating Voronoi diagram by boundary of primal mesh [Garimella et al., 2014]. 66
3.10 An *intuitive* definition of isotropy [Ben-Chen, 2012]. 67
3.11 Example of Lloyd's algorithm [Dominik Moritz, 2018]. The plus signs denote the centroids of the Voronoi cells. 68
3.12 (a) Left: Ordinary Voronoi tessellation of a point set sampled from some density function (b) Right: Point set and its corresponding weighted centroidal Voronoi tessellation for the same density function. Each site coincides with the center of mass of its Voronoi cell. The sample set on the right was generated by Lloyd iterations applied to the sample set on the left. [Surazhsky et al., 2003]. 68

3.13 Primal/Dual Triangulations: (a) Centroidal Voronoi Tessellations (CVT) – barycentric dual; (b) CVT – circumcentric dual; (c) Optimal Delaunay Triangulation (ODT); (d) Hodge-Optimized Triangulation (HOT) – optimized dual mesh alone; (e) HOT – optimized both the primal and dual meshes. 69
3.14 All permutations of a triple (i, j,k) refer to the same triangle, and the sign of the permutation determines the orientation [Elcott and Schroder, 2006]. 71
3.15 Simplicial mesh with enumerated vertices and simplices [Tutorial PyDEC, 2012]. 72
3.16 Simplicial complex with oriented edges and triangles [Tutorial PyDEC, 2012]. 72
3.17 There is one dual polyhedron for every primal vertex, one dual polygon for every primal edge, one dual edge for every primal triangle, and one dual vertex for every primal tetrahedron [Elcott and Schroder, 2006]. 74
3.18 (a) The MEMS bridge condition after the release stage: electrode pad of size 20 μm $\times$ 160 μm is located underneath the suspended bridge. (b) The condition of the originally straight bridge structure before applying voltage [Latif et al., 2017]. 78
3.19 Geometry and boundary conditions of electrostatic MEMS: (a) Plot of a fixed-fixed beam separated from a fixed ground plane by dielectric spacers. (b) Plot showing the deformation profile of the beam due to an electrostatic force (constant drive voltage). . . 79
3.20 Schematic view of the thermal dilatation MEMS actuator. 80
3.21 Physics of the electrokinetic problem: an electric potential ϕ_1 is imposed at the top surface while an other electric potential is imposed at the bottom surface of the beam. Others faces (front, rear, left and right) are insulated (no current flow is possible). 80
3.22 Physics of the thermal problem: imposed ambient temperature T_0 on boundary conditions and internal volumetric heat sources, produces by Joule's effect. 80
3.23 Physics of the elastostatic problem: displacements clamped at the right and left ends of the beam (clamped condition) while the other surfaces of the beam are free of stress. A volume force is generated by the thermal expansion into the beam. 81
3.24 Tonti diagram of the current flow problem. 81
3.25 Tonti diagram of electrical circuits: two ways to travel it, using ϕ, potentials at nodes (Node analysis) or using $\mathbf{i}$, mesh currents (mesh analysis). 82
3.26 Comparison between our computational results and the results given by the FEM software ANSYS: Electrokinetic problem. Display of the voltage field 83
3.27 Tonti diagram of a thermal problem. 84
3.28 Comparison between our computational results and the results given by the FEM software ANSYS: Thermal problem. Display of the temperature field. 85

3.29 Position of an elastostatics problem according to [Tonti, 2013]: two particles $\mathcal{P}$ and $\mathcal{Q}$ in the initial configuration occupy the positions $\mathbf{P}$ and $\mathbf{Q}$. The same particles, at a later instant, occupy the positions $\mathbf{P}'$ and $\mathbf{Q}'$. The vectors in thin lines are position vectors, whereas those in thick lines are the displacements of the two particles. 86
3.30 Tonti diagram of the elastostatics problem [Tonti and Zarantonello, 2009]. 87
3.31 Comparison between our computational results and the results given by the FEM software ANSYS: Mechanical problem. Display of the displacement field. 87

4.1 The complete structure displaying all stages of our modeling approach: from geometrical models via Cochain Method to lumped models. 90
4.2 Partial element equivalent circuit method (PEEC) 93
4.3 (Lp)PEEC 95
4.4 (P)PEEC. 96
4.5 (R)PEEC 96
4.6 Four volume cells, separated by dashed lines, accounting for the current flowing in the direction of the arrows. 96
4.7 Dual-PEEC [Freschi et al., 2006]: (a) Unstructured discretization: consecutive triangular dual volumes. The couple primal edge (straight line)/dual face (dark gray) correspond to a two terminal circuit component. Light gray faces are related to the evaluation of mutual resistances; (b) Corresponding elementary equivalent circuit. 98
4.8 Tonti's diagram of electrokinetic for lumped and distributed parameters systems. 99
4.9 Example of one layer neural network. 104
4.10 Different sigmoid functions. 104
4.11 Example decision tree [Bishop, 2006]. 107
4.12 Example decision tree [Bishop, 2006]. 107
4.13 Fruit data divided by Color [Bishop, 2006]. 107
4.14 Fruit data divided by Diameter [Bishop, 2006]. 108
4.15 Root node of the fruit decision tree [Bishop, 2006]. 108
4.16 K-means clustering with two clusters [Bishop, 2006]. 110
4.17 Plot of basketball players and dividing lines [Bishop, 2006]. 111
4.18 Plot of basketball players for amateur team [Bishop, 2006]. 112
4.19 Basketball players in polynomial space [Bishop, 2006]. 112
4.20 Core points and border points [Ester et al., 1996]. 115
4.21 One step of DBSCAN [Ester et al., 1996]. 115
4.22 Comparing different clustering algorithms on toy datasets [Tutorial Sci-Kit, 2012]. 117
4.23 Lower part of software structure (see Figure 4.1). 117
4.24 2D Magnetostatic case study (relay): components and geometry. . 119
4.25 2D Magnetostatic case study (relay): Primal mesh obtained from GMSH (13692 triangles). 120
4.26 Upper part of the Tonti diagram of a magnetostatic problem [Alotto et al., 2013]. 122

4.27 Discrete form b (linear interpolation by ParaView) for the relay example. 122
4.28 Interpolation of $\vec{B}$ field with Whitney elements. 125
4.29 Value of the $\vec{B}$-field (z-plane is the amplitude of $\vec{B}$) in the xy-plane after Whitney interpolation in circumcenters of primal cell. 126
4.30 Result of DBSCAN - clustered cells: 184 classes (One specific color (and not one specific circle) corresponds to one cluster. The color range does no exactly match that of Figure 4.27 and 4.28). 126
4.31 Histogram of density levels of initial clustering of Figure 4.30 . . . 127
4.32 Reduction of density levels by Waterfall algorithm. 128
4.33 Illustration of the reduction of dofs with histogram: histogram of density levels of final clustering (corresponding for 25 clusters). . . 128
4.34 Final clustering with 25 clusters 129
4.35 Complete Tonti diagram of magnetostatics with no current sources ($j = 0$ here). Ψ is a well defined "single-valued" magnetic scalar potential only in region with no current source [Verite, 1987]. . . . 130
4.36 Complete Tonti diagram of magnetostatics with current sources. h_s is due to current sources such that $j = \tilde{C} \cdot h_s$ because $\tilde{C} \cdot (-\tilde{G} \cdot \Psi) = 0$. 131
4.37 One clustered region with its boundary sub-divided into tubes of magnetic flux (One reluctance by sub-boundary). 133
4.38 Clustered cells (25 classes) and the corresponding lumped network (see Figure 4.40 for details on the network). 133
4.39 Minimum spanning tree of digraph: branches of the tree are traced in yellow and links in grey. 134
4.40 Clusters and final LP networks. Only grey links contain mmf sources 135

B.1 Difference between convex and non-convex set. 147
B.2 Example of a convex hull (green shape). 148
B.3 Difference between valid and invalid representation of simplicial complex. 149
B.4 Basic topological operations: closure, star, link. 150
B.5 The example of graph [Perrin, 2017]. 151

List of Tables

3.1 Physical and geometric characteristics of the MEMS. 78
3.2 Comparison of the results between ANSYS and the cochain method calculation. 88

4.1 Summary of data set dimensions and DoF numbers. 136

Acknowledgement

During my PhD thesis, I met many nice and important people that contributed to finish this thesis. As I was in Roberval laboratory, M2EI team (Mechatronics, Energy, Electricity, Integration), situated in Compiègne, France, I would like to thanks them in French (even that this thesis is written in English).

J'exprime mes remerciements les plus vifs à M. Arnaud HUBERT, Professeur à l'Université de Technologie de Compiègne, pour m'avoir guidé, encouragé et accompagné tout au long de ce travail, qui je l'espère, sera un viatique pour l'avenir. La juste proposition des choses, le sens de ce que l'on peut croire et de ce dont il faut douter, le sens de ce qui est solide ou fragile, ce qui est important ou accessoire, marquèrent ces nombreuses interventions. Sans me manifester d'impatience devant les difficultés, il suggérait le plus souvent et incitant à travailler ensemble, il consacrait son temps précieux et son attention pour m'aider à avancer ma thèse. De cette attitude pédagogique trop rare et par sa bienveillance, je tiens à lui témoigner toute ma gratitude pour l'avoir eu comme directeur de thèse.

Ainsi, j'exprime mes remerciements les plus profonds à M. Alejandro-Luis OSPINA-VARGAS, Maître de conférences à l'Université de Technologie de Compiègne, pour avoir remplir avec disponibilité, compétence et gentillesse, la dure tâche de l'encadrant de ma thèse. Ses remarques pertinentes ainsi que son expertise très professionnelle ont participé à l'évolution et la bonne marche de nos travaux.

Je tiens à adresser mes remerciements à M. Gérard MEUNIER, Directeur de recherche au CNRS, pour avoir accepté d'être rapporteur de ma thèse et de participé au jury.

Je remercie également vivement M. Stéphane CLENET, Professeur des universités aux Arts et Métiers ParisTech, pour avoir accepté d'être rapporteur de ce mémoire et de participé au jury.

De même, je remercie Mme Dominique CHAMORET, Maître de conférences à l'Université de Technologie de Belfort-Montbéliard, et Mme Delphine BRANCHERIE, Professeur des universités à l'Université de Technologie de Compiènge, d'avoir accepté de participer au jury.

Mes remerciements vont également à M. Sulivan KÜTTLER, docteur en génie électrique de l'UTC, pour m'avoir encouragé de me lancer dans cette aventure.

J'exprime toute ma reconnaissance à mes collègues de laboratoire, M. Guy FRIEDRICH, M. Nicolas DAMAY, Mme Khadija EL KADRI BENKARA, M. Christophe FORGEZ, M. Vincent LANFRANCHI, M. Nicolas PATIN, M. Stéphane VIVIER, M. Loïc CHARBONNIER et M. Didier LEMOINE pour m'accueillir dans l'équipe et pour leur contribution à la création d'une ambiance amicale au sien du laboratoire. Je remercie toutes les personnes qui m'ont aidé à mener à bien ce travail.

Un clin d'œil amical à tous mes amis thésards avec qui nous avons su entretenir dans laboratoire une agréable ambiance de travail.

Je tiens à remercier également Dr. Jean-Michel CAHN-FILACHET, mon médecin, pour me remettre en forme et en bonne santé et les encouragements qu'ils m'ont apportés.

Je souhaite à tous mes collègues une bonne continuation avec beaucoup de courage, de persévérance et surtout de succès ...

Chapter 1

Introduction, context and objectives

Was man nicht weiß, das eben brauchte man,
Und was man weiß, kann man nicht brauchen.

Johann Wolfgang von Goethe,
Faust

1.1 Model-based systems engineering

The International Council on Systems Engineering (INCOSE) defines Model-Based Systems Engineering (MBSE) as "... an approach to engineering that uses models as an integral part of the technical baseline that includes the requirements, analysis, design, implementation, and verification of a capability, system, and/or product throughout the acquisition life cycle" [Beihoff et al., 2014]. In other words, MBSE is a systems engineering methodology that focuses on the creation and operation with numerical models as the primary means of communication between different engineers teams, rather than on the exchange of information based on documents (paper, electronic files or databases).

At present, paper documents and verbal exchanges constitute the main means of communication and information sharing between the various stakeholders of project development in all stages. This document-based engineering has shown its effectiveness but also its limitations. The increasing complexity of the systems can only convolved the contradiction between the difficulty of development of more and more sophisticated technological systems with increasingly large teams, distributed geographically over various cultural areas and continents.

That is why the INCOSE has developed a vision to 2025 of the systems engineering in which they consider that the future of systems engineering will be based on the MBSE approaches and that it will open up new engineering practices. Therefore, the INCOSE urges "... MBSE has grown in popularity as a way to deal with the limitations of document-based approaches, *but* is still in an early stage of maturity similar to the early days of Computer-Aided Design (CAD) / Computer-aided Engineering (CAE)" [Beihoff et al., 2014].

The challenge of MBSE is to meet together all teams and how they can apply collaborative methods to match the different stages of product design in a multi-disciplinary and multi-profession environment. Because engineers still face design failures that rising during the late phase of the physical integration, the modeling and numerical simulation in MBSE approach is intended to integrate the appropriate modeling and simulation techniques at each stage of the development life cycle (so called, "V-cycle" as displayed in Figure 1.1).

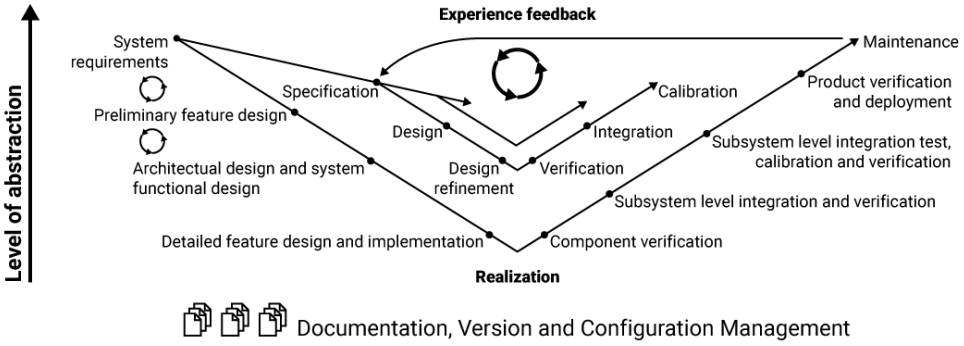


Figure 1.1: The V-model of the systems engineering process [Fritzson, 2011].

This procedure allows the validation of the design decisions assigned to a each stage and makes possible to verify the solution at a given step before going to the next one. Moreover, the V-cycle proposed in Figure 1.2 for the field of aeronautics by ARP-4754A [ARP, 2014] stresses two levels in every V-cycle: i) the *system level* mainly concerned by architecture definitions and ii) the *items level* for the modeling and design of items and components. According to ARP-4754A, different teams need to collaborate together during the design of new products and they can be splitted into 3 distinct groups: *Systems Architects*, *Engineering Groups* and *Validation Groups*.

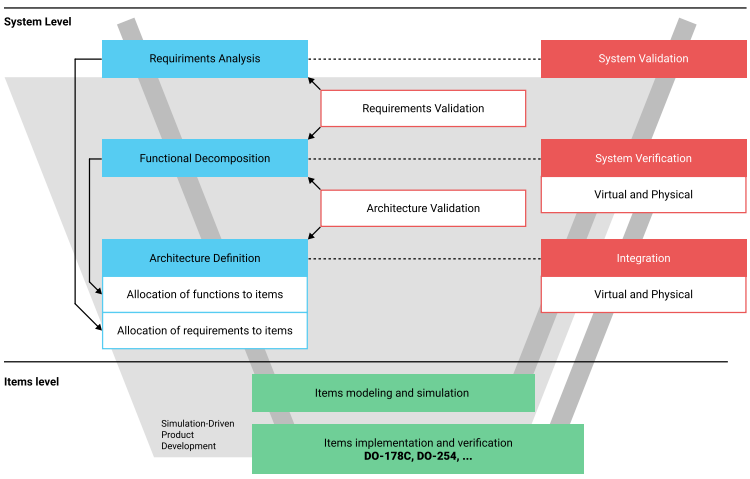


Figure 1.2: Simulation-Driven Product Development view by ARP-4754A [ARP, 2014].

As can be seen in Figure 1.3, all these three groups are using their own practices, approaches, methods and softwares. This thesis is focused on modeling and design aspects and its main contributions concern the *items level* and the proposal of new tools for *Engineering Groups* in the aim of simulation for (multi-)physical technological devices.

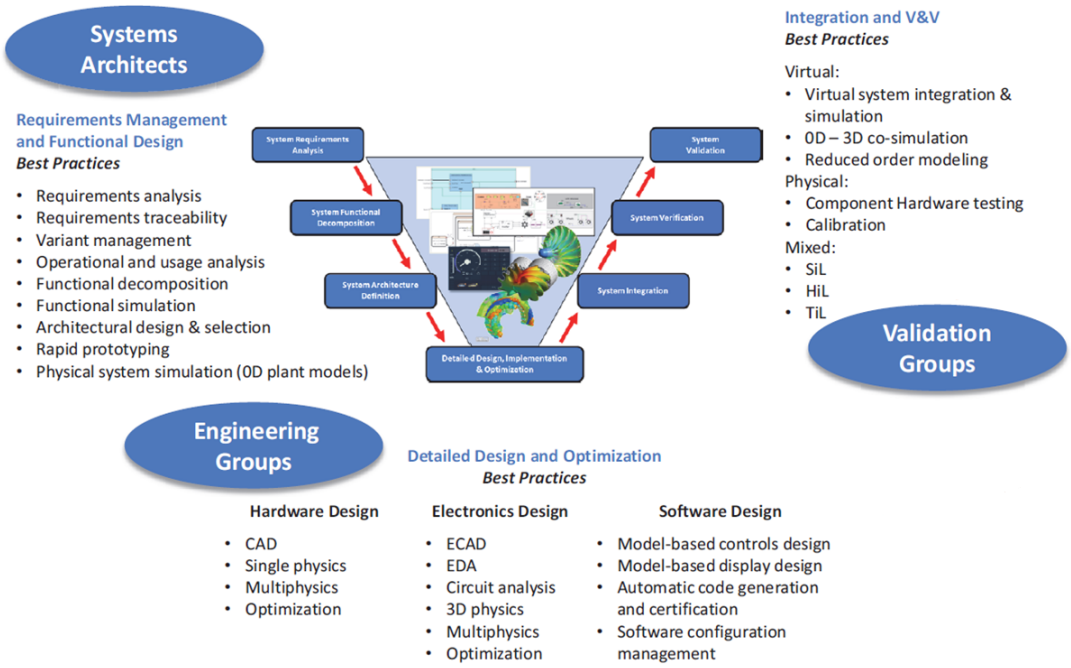


Figure 1.3: Stakeholders during systems engineering process [ARP, 2014].

For *Engineering Groups* at the *Items level*, Figure 1.3 presents different types of methods for different technological domains (Hardware, Electronics and Software). In such a way, the main developments proposed in this thesis are concerned with:

- CAD and geometric design of new devices,
- Single, multi and 3D physics,
- Circuit simulation and analysis,
- Model-based control and display design,
- Models for simulation and optimization purposes.

However, in order to propose a more significant advance in the field of MBSE, we also wish that these tools and models can be used by *Validation Groups*. The latter generally use simpler, often reduced-order models for co-simulation, real-time control, validation, monitoring and diagnostics during operation (see "Validation groups" in Figure 1.3). This work is part of a multi-level and multi-objective approach by developing bridges between the different groups operating at different levels of the product design V-cycle. One of the main challenges of this work is ultimately to ensure a certain compatibility between modelling and simulation tools and methods for different groups and levels.This is the main objective of this doctoral thesis.

1.2 Multi-level modeling

It is worth noting that, inside the *Engineering Groups*, there exist different levels of modeling for different purposes. For example, Figure 1.4 proposes several modeling levels for MEMS and electronic components design [Kasper, 2011, Hubert, 2002] and Figure 1.5 proposes different models for simulating a coil with core with different degrees of precision [Brisset, 2007].

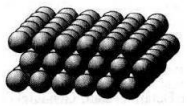
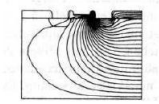
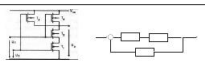
	Level	Simuling
	Solid state level (microscopic, quantum)	Crystallographic and band structure, cohesive energy, electronic properties...
	Geometric level (macroscopic, classical)	Fields (Temperature, stress, strain, magnetic, electric...)
	Component level(discrete elements, lumped parameters models)	resistance, inductance, capacitance, stiffness, mass, damping coefficients...
	System level (dynamic behaviour)	current, voltage, force, displacement...

Figure 1.4: Different levels of modeling for MEMS and electronic devices [Kasper, 2011, Hubert, 2002].

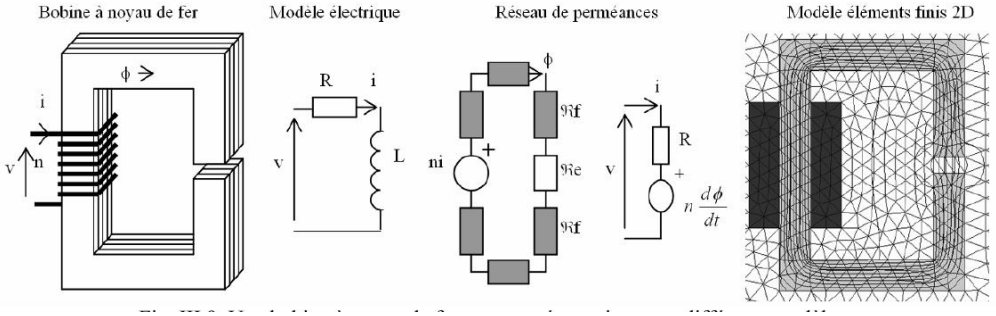


Figure 1.5: A coil with core and different models of its behaviour [Brisset, 2007].

It is common in *Engineering Groups* to split models into three different classes/levels according to their precision. The choice of one class among others during *Simulation-Driven Product Development* depends on their computation time. The more precise is a model, the less rapid is its evaluation. Therefore, because optimization procedures require a huge number of evaluations of one model, the computation time of precise models becomes a bottleneck for designing. In this aim, S. Brisset in [Brisset, 2007] proposes a two criteria Pareto Front for different classes of models : computation time versus modeling error. The *location* of corresponding models in this Pareto front is reported in Figure 1.6. As suggested previously, this front clearly shows that every model can be classified into three distinct classes:

1. Class of level 1 (or *analytical* models). They are the less precise but the more rapid models. They are mainly used for architecture and preliminary design. They concern quasi-static behavior and are usually focused on nominal (or maximal) working. At this level, time and space are not taken into account. Corresponding models are mainly based on a set of *algebraic equations*.

2. Class of level 2 (or *semi-analytical* models). This is the class of *lumped* models constituted with networks of discrete components. At this level, time is taken into account but not space. Corresponding models are mainly based on a set of *algebra-differential equations*.

3. Class of level 3 (or *Finite-Elements*-like models). This is the class of numeric models that approximate *distributed* models. They are based on Finite Elements or similar methods (Finite Difference, Finite Volume, FDTD, Cell-Method, etc... [Mattiussi, 2002]). The number of degrees of freedom of these models are usually quite big therefore these models are precise but time consuming. Nevertheless, their main advantage is that they are based on a precise description of the geometry of the device (usually a CAD model) contrary to the level 1 and 2 models. At this level, time and space are taken into account. Corresponding models are mainly based on a mesh-based discretization of *partial-differential equations*.

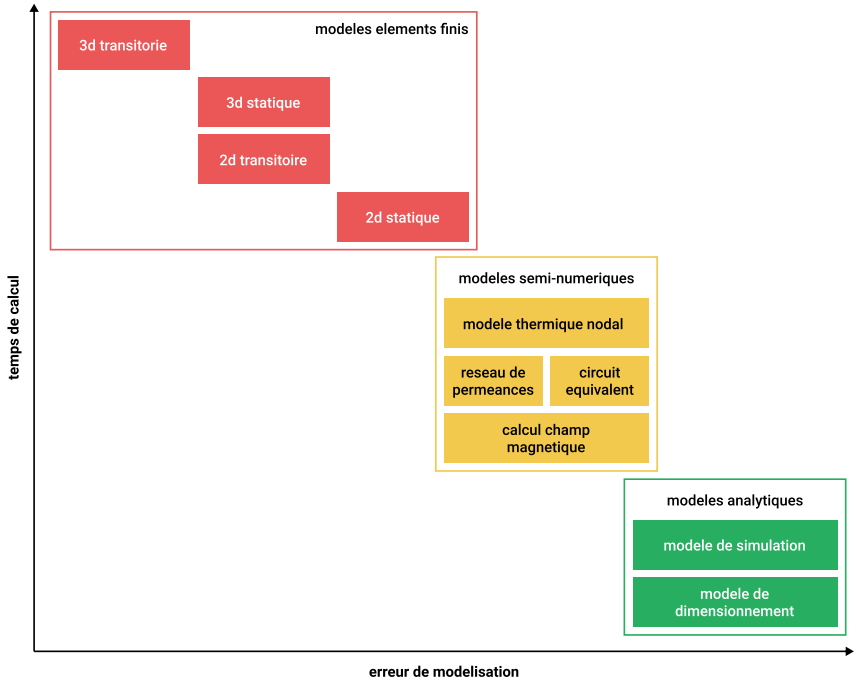


Figure 1.6: Pareto front of models used for electric motor design. [Brisset, 2007]

The most precise models, i.e. Finite Elements-like Models, lies on one extremity of the Pareto Front proposed in Figure 1.6. Nevertheless, because of their computation times, they are hardly used for optimization purpose even if some exceptions can be found in literature ([Biedinger and Lemoine, 1997, Sanogo et al., 2014, Kuci et al., 2017]). At the opposite side of the Pareto front, analytical models are currently used for architecture generation and preliminary design

steps but are hardly used for precise Items design because of their low precision. Actually, analytical models are mainly dedicated to *Systems Architects* at *System level* rather than *Engineering Groups* at *Items level*. Finally, models for optimization concern mainly the middle class, i.e. these of lumped models. As a matter of fact, they present the best compromise between precision and computation time for optimization purposes.

Nevertheless, one of the main drawbacks of lumped models is that they must be used by expert and experienced designers because they are not generated from geometric description (based on CAD models) but from *equivalent* models that require good *know-how* and *skills* of designers. The main objectives of this thesis is to mitigate this drawback by proposing automatic generation of lumped models from FEM-like models and CAD-based geometry description. This work proposes therefore a *multi-level* or *multi-scale* approach for modeling, simulation and design at *Items level* for *Engineering groups*.

To illustrate these objectives, Figure 1.7 presents a succession of models for designing electrical devices. This figure is, in fact, a simplification of the lower left part of a V-cycle. At top levels, *usefull* models are those that propose a general view of the system whereas, at lower levels, *usefull* models are those that propose deeper precision and details.

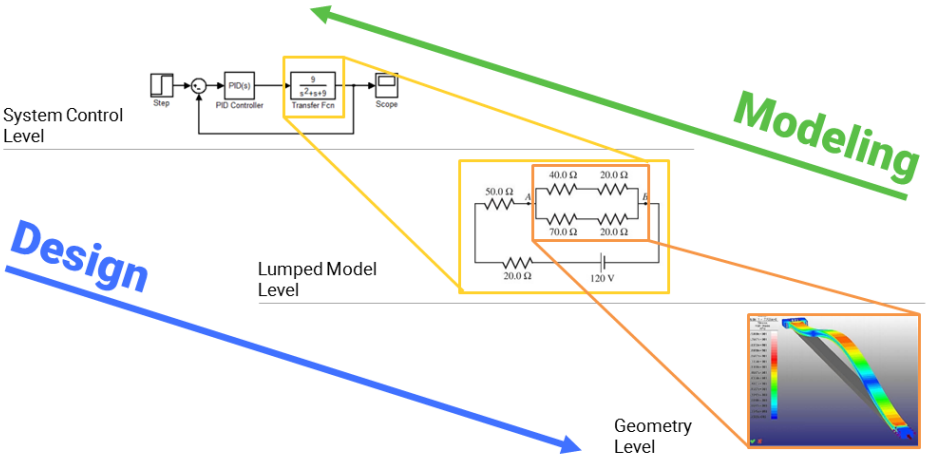


Figure 1.7: Coupling between multi-levels models : *design* and *modeling* views.

Additionally, in this figure, we stress two opposite *views* corresponding to *design* and *modeling* activities. The Top-Down view (Design) is used by engineers for design tasks according to the V-cycle procedure. Conversely, the Bottom-Up view (Modeling) is used by engineers to build new models or to reduce them. In order to use the MBSE approach throughout these two views, engineers have to operate with multi-level and multi-physical numeric models. Herein lies the main problem of implementation of MBSE. Even if MBSE has grown in popularity, it's not enough mature to deal with multi-level and multi-physics problems because of shortcomings of simulation software or use of different and incompatible softwares at differents levels. Even if some software can deal with multi-physics problems they hardly manipulate multi-level or multi-scale model as required by the opposite *views* of *design* and *modeling* activities. In this context, the main

contribution of our work is to propose a concept of automated lumped model generation from geometrical (3D) models. More precisely, the question to answer is: *How to get a* second level *model in Figure 1.7 from the* third level *model and, at the same time, preserving physical and topological properties of the initial* third level *geometric model ?* We believe that such kind of tool will greatly help the designers and engineers that manipulate simultaneously models from network and geometry levels. Currently, this task is based on a DIY (Do It Yourself) approach, based on expertise, intuition, experience and skills of designers. In the MBSE approach, it will be necessary to propose systematic procedures to automate these complex tasks.

1.3 Thesis outline

As we will see in more detail in the following chapters, this work will use modeling methods that are well known in mathematics and physics but not yet widely used in engineering. They will use both a geometric and topological approach (differential and algebraic topology), an energetic approach (duality and differential form theory) and advanced discretization and meshing techniques (Delaunay-Voroinoi meshing and computer graphics tools). The analogies so useful in multiphysics modeling will be presented using E. Tonti's perspective and diagrams [Tonti, 2013]. Finally, model reduction techniques will use recent machine learning algorithms to perform model clusturing and condensation. This work will be facilitated by the use of algebraic topology in the initial stages of modeling.

Chapter two will present the different theoretical tools necessary for this work. In particular, it will detail the choices that have been made and the reasons for these choices. Chapter 3 will present the practical development of these tools in the case of multiphysical modeling. The algorithms for meshing, modeling as well as simulation for the different physical fields will be presented. Examples of results will then be proposed for electrical, thermal and mechanical applications. Chapter 4 will focus on the problem of model reduction. This must be conducted while trying to keep the initial topological structure of the models, which implies restrictions on the techniques that can be used. After a brief state of the art of model reduction techniques and machine learning techniques, the tools finally selected will be presented in detail. An application example will be conducted on a magnetostatic example. Finally, the methodology for automatic lumped model generation will be presented. A final chapter of conclusion and perspective will complete this manuscript. It will highlight the main contributions of this thesis and propose some ways to continue this work.

Chapter 2

Multilevel and multiphysic modelling: features and requirements

To meet the requirements of the MBSE approach, the previous chapter showed the importance of building digital tools to automatically link or generate models at different levels of the V-cycle. In this thesis, we have decided to focus mainly on two levels: the level 2, i.e. the level of *Network* and *lumped parameters models* and the level 3, i.e. the level of *Geometry* and *distributed parameters models* (fields distributed in space). As our main goal is the automatic coupling of multiphysic models between these two levels, some properties and characteristics must be shared by these two kinds of models. In this aim, this chapter presents firstly the properties and main characteristics of models at level 2 and 3. Then, the rest of the chapter presents a multilevel approach for sharing these various features. The practical development and implementation of these characteristics in a numerical environment will be detailed in the following chapters.

A brief analysis of the requirements of targeted objectives reveals two needs that will guide our approach throughout this research work. These needs can be expressed in practice through the following two points:

1. Attempt to use a *system approach* that manipulates *energy quantities* to facilitate interconnection between multiphysic sub-systems.

2. Develop tools that preserves the *topological structure* of lumped parameter models to facilitate model reduction from level 3 to level 2.

The first requirement is obvious: for complex and interconnected systems, it is important to guaranteed a systematic connection between several models or systems, regardless of the type of physics which is considered. The second requirement is a necessity for reducing the number of degree of freedom (DOF) from *field* description towards *Network* description while keeping topological properties. As a matter of fact, imagine the case where we have generated a lumped model with the same DOF as the initial discretized distributed model. In this case, the practical utility of using lumped model instead of discretized distributed model is very limited.

2.1 Properties and characteristics of models

The aim of this section is to resume the main features of each type of models, such that the main characteristics can be shared in the modeling tools proposed in the next sections and chapters.

2.1.1 Multiphysic models and energetic duality

One of the purpose of this thesis is the modelling of multiphysic systems. In this case, it is required to adopt a generic point of view regardless of the type of physics. Because thermodynamics and variational methods adopts naturally this genericity, we decide to built our models in their framework.

A first illustrative example

Before proposing more abstract ideas, let us start with a very simple mechanical example. Let x be the displacement of a linear elastic spring submitted to a force f (see Figure 2.1. x is supposed null at rest). The position x describe the *configuration* of the system, f is the *source* of motion and/or deformation. In this way, we can talk about *configuration* and *source* variables. The *scalar/inner/dual* product between the position (configuration variable) and force (source variable) is the energy, a third kind of variables called energy variables. Thus, f is the energetic dual of x. As the dual variable of x, f is written x^* by mathematician. A constitutive law is a constraint – a mapping – between these two dual variables. For elastic behaviour, this relationship is a *bijection*, i.e. a one-to-one map and the system is said *conservative*. The most simple constitutive law of a spring is described by its stiffness k such that for a linear elastic spring:

$$f = k \cdot x \tag{2.1}$$

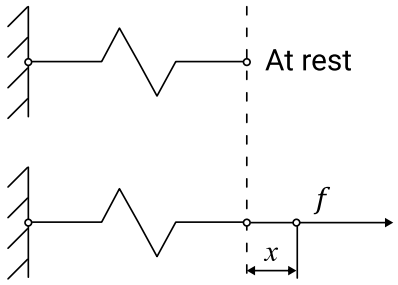


Figure 2.1: Sketch of a spring. Top view : spring at rest ($x = 0$ and $f = 0$). Bottom view : spring with a deformation (with a displacement x when submitted to a force f).

For this conservative system, the work $\mathcal{W}$ done by the applied force during this change of configuration is stored into the spring as an elastic energy $\mathcal{E}$. The computation of this elastic energy $\mathcal{E}$ is illustrated on Figure 2.2 for a linear

constitutive law: this can be computed using integral or variational calculus if f is considered as a function of x as implied by the constitutive law:

$$\mathcal{E} = \int_0^x f\,dx = \int_0^x k \cdot x\,dx = \frac{1}{2}k \cdot x^2 \tag{2.2}$$

Because energy is a *state function*, f can be computed from this energy as:

$$f = \frac{\partial \mathcal{E}}{\partial x} = \frac{\partial \left(\frac{1}{2}k \cdot x^2\right)}{\partial x} = k \cdot x \tag{2.3}$$

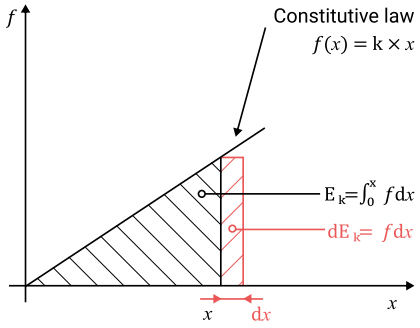


Figure 2.2: Computation of the elastic energy in the spring of Figure 2.1.

This is the basics of all variational methods for modeling multiphysic problems. Good examples can be found in [Lanczos, 1986] for mechanical problems, in [Hammond, 1981] for electromagnetic problems and in [White and Woodson, 1959, Crandall et al., 1968] for electromechanical problems.

Extension to a broader field

Mathematically, this energy framework is rigorously generalized through duality theory so that two duals x and x^* define a product of duality $< , >_{\mathcal{E}}$, i.e. an energetic scalar product. If we take (2.2), it can be seen that the integrand is a duality product,

$$f\,dx =< x^*, dx >_{\mathcal{E}} \in \mathbb{R} \tag{2.4}$$

This integrand is a *linear differential form* ω. Then a linear differential 1-form ω – or a linear differential form of degree 1 – can be computed from an infinitesimal change of configuration dx. In this way,

$$\int_0^x f\,dx = \int_0^x x^*\,dx = \int_0^x < x^*, dx >_{\mathcal{E}} = \int_0^x \omega = \mathcal{W} \tag{2.5}$$

Then, this linear differential form ω can be integrated for a *finite* change of configuration $0 \to x$ to compute the work $\mathcal{W}$ done on the system. For a conservative system, this work is stored as an internal energy $\mathcal{E}$ with $d\mathcal{E}$ an *exact* differential form [Arnol'd, 1997]. In such a way, the dual variable x^* can be computed as:

$$x^* = \frac{\partial \mathcal{E}}{\partial x} \tag{2.6}$$

In classical thermodynamics, x and x^* are classified as *extensive* and *intensive* variables [Callen, 1960] and two energetic dual variables are linked mathematically by a *metrics* that corresponds physically to a phenomenological constitutive law. Based on a classification proposed firstly by [Penfield and Haus, 1967], E. Tonti proposed in the 1970's to call these variables *configuration* and *source* variables, according to topologic properties enlightened in the next sections. In this work, we retain the names proposed by Tonti. Then, like previously used, x is a configuration variable whereas f is a source variable.

In this energetic framework, two dual variables (from configuration and source dual spaces) are linked by a physical constitutive laws, i.e. a metrics. In the field of differential geometry, constitutive laws are described using *Hodge star* operators [Bamberg and Sternberg, 1991] as it will be used in the next chapters.

As mentioned at the opening of this chapter, the main purpose of this work is to model interconnected subsystems such that the whole device is constituted by the interconnection of several *energy storage devices*. Interconnection between sub-systems gives rise to additional equations between configuration and sources variables. Corresponding *topologic* equations are fundamentaly different from metric constitutive laws seen before. Metric equations links dual quantities (source vs configuration quantities) whereas topologic equations links quantities of the same type (source with source quantities *or* configuration with configuration quantities). Then, interconnection equations define a geometric structure of the whole device according to its *topology*. They are *metric free* and their mathematical topological framework is the purpose of the next section.

2.1.2 Topology of network for second level models

One of the main characteristics of models at the second level is that they are focused on topology and interconnection of components. In this aim, the focus is not, as previously, on the metric constitutive laws but on topological interconnections laws. Algebraic topology is not a recent topics and it goes back to Euler work and its famous Kœnigsberg bridges problem. At present, this is always a hot topics of research for computation in physics and for telecommunication or computer networks design. In engineering, corresponding methods are extensively used for numeric simulations of electric circuits using nodal, mesh, loop or cut-set methods [Desoer and Kuh, 1969a]. For instance, modern computer-aided analysis of electronic devices are nearly all based on such methods. [Chua and Lin, 1975] presents the basics of these electronic circuits computation methods. In engineering, this approach was also extended for multiphysic and mechatronic devices. [Rowell and Wormley, 1997] presents this generalization in a very educational way.

As proposed in the previous section, physical quantities can be divided into 2 dual groups. In lumped systems engineering literature [Cannon, 2012], they are called *across* and *through* variables[1]. Across variables are most of the time

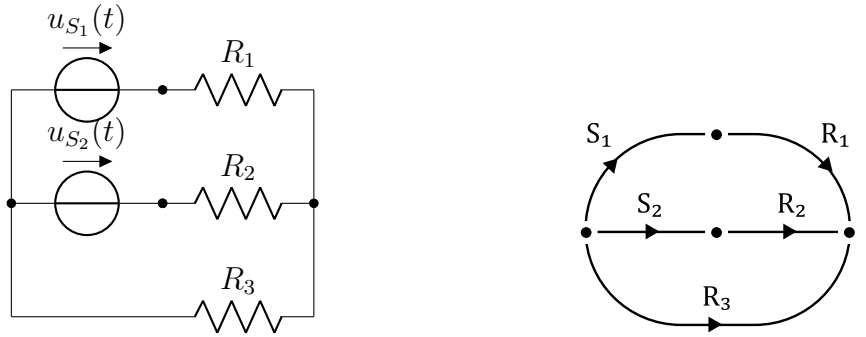
[1] *Across* because they are defined across two points. *Through* because they are defined through a cross-section. In engineering literature, these two types of variables are sometimes

configuration variables therefore through variables are sources variables.

A network is geometricaly described by an oriented – or directed – linear graph, i.e. a *digraph*. Each edge of the graph represents a dipole, i.e. an elementary component with two terminals of interconnection. A vertex represents a connection node between several terminals of components. Oriented surfaces can also be defined on digraph. They correspond to surfaces bounded by several edges forming an oriented closed loop. In circuit theory, a vertex is called a node, an edge a branch and an oriented surface a mesh.

A first illustrative example

To illustrate network modeling and present the corresponding mathematical properties and definitions, let us illustrate this section with an example of a very simple electrical circuit such as the one shown in Figure 2.3. This electrical circuit will be represented as a Kirchhoff network, itself represented by an oriented linear graph. In this example, the digraph $\mathcal{G}$ has $n_E = 5$ edges (corresponding to the 5 branches of the network containing only one lumped component) and $n_V = 4$ vertices (corresponding to the 4 nodes of the network).



(a) Description of the circuit. (b) Oriented linear graph.

Figure 2.3: Description of a very simple electric circuit and its digraph.

Definition 2.1.1 (Oriented linear graph)**.** *An oriented – or directed – linear graph is a schematic representation allowing to graphically describe the interconnections between nodes and branches of a Kirchhoff network, a node becoming a vertex and a branch, an edge of this graph. A graph is oriented when each of its edges has an orientation. We also speak of oriented arcs to designate oriented edges.*

At each vertex indexed by an integer $1 \leq l \leq n_V$, we can associate a corresponding potential ϕ_l and at each edge indexed by an integer $1 \leq k \leq n_E$, we can associate a current – through – variable i_k circulating between the two terminals of this lumped component. The voltage v_k of a subsystem k is defined as the difference between the potentials at its two terminals. The orientation of the branches of a Kirchhoff network is generally chosen as that of the current variables circulating in each branch.

called *two-points* and *one-point* variables because they are measured across two points or at one point [Cannon, 2012]. This classification is based on the *mobility analogy* for discrete/lumped systems [Firestone, 1932].

Properties 2.1.2 (Subgraph, degeneration and connectivity)**.**
An oriented linear graph can be decomposed into subgraph*, which contains only a subset of edges and vertices of its original graph. A degenerate sub-graph is a sub-graph that contains only one vertex. A connected graph is a linear oriented graph where there is at least one path to go from each vertex to all the others. The number of separate parts of a graph is then the maximum number of its connected subgraphs. A graph is therefore said to be connected when, from any vertex, all the other vertices are reachable by a path passing through its edges. When a graph is not connected, it can be considered as representing several separate networks.*

An oriented linear graph can be described algebraically using an incidence matrix. This will make it possible to fully describe the topological structure of the graph, i.e. how the different subsystems of the underlying network are interconnected at the nodes of its terminals.

Definition 2.1.3 (Incidence)**.** *An edge of a graph (or a branch of a network) is said to be* incidental *at a vertex (or at a node) if that edge leaves or enters that vertex.*

The interconnection topology of an algebraically oriented linear graph can be fully described by indicating, for each edge, which vertex it leaves and into which vertex it enters. This information can be grouped in a table called the *vertex to edge incidence matrix*, or more simply *incidence matrix* of the graph.

Definition 2.1.4 (Extended incidence matrix $\mathbf{A}_e$)**.** *The "extended" incidence matrix $\mathbf{A}_e$ of a graph $\mathcal{G}$ is a matrix with n_V rows associated with the Vi vertices of the graph ($1 \leq i \leq n_V$) and n_E columns associated with the Ej edges of this graph ($1 \leq j \leq n_E$). Its a_{ij} components are defined as follows:*

- $\star$ $a_{ij} = +1$ *if the Vi vertex is the starting vertex of the Ej edge,*
- $\star$ $a_{ij} = -1$ *if the Vi vertex is the ending vertex of the Ei edge,*
- $\star$ $a_{ij} = 0$ *if the Vi vertex does not belong to the Ej edge.*

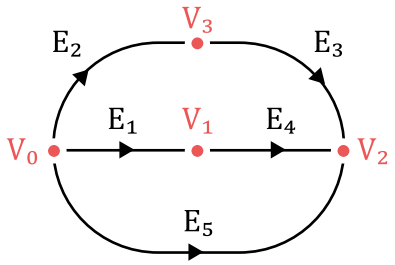
This matrix is of rank $\rho = n_s - 1$ whereas ρ is called the rank of the graph $\mathcal{G}$.

For the illustration circuit, ρ =3 and its incidence matrix is displayed in FIG. 2.4.

Definition 2.1.5 (Reduced incidence matrix $\mathbf{A}$)**.** *Matrix $\mathbf{A}_e$ being of rank $n_s - 1$ for n_s lines, one of these lines is a linear combination of others and can be skipped without loss of topologic information. The deleted line is most of the time the line ossociated with the vertex of the reference potential. The $\mathbf{A}$ matrix obtained after a one line deletion is called the reduced incidence matrix of the graph.*

Incidence matrices (reduced or not) allow a systematic definition of generalized Kirchhoff laws of the network. For this purpose, Gauss surface and Cut-set have to be carefully defined.

Definition 2.1.6 (Gauss Surface and Cut-set)**.** *A Gauss surface is an oriented surface defined on a graph whose boundary cuts several edges of the graph. The full set of the cut edges define a Cut-set of the graph. In a similar way, a Cut-set can be seen as the set of edges that must be cut to transform a connected graph into a non-connected graph.*



$$\mathbf{A}_e = \begin{array}{c} \\ V0 \\ V1 \\ V2 \\ V3 \end{array} \begin{array}{c} \begin{array}{ccccc} \text{E1} & \text{E2} & \text{E3} & \text{E4} & \text{E5} \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & -1 & -1 \\ 0 & -1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

(a) Description of the digraph. (b) Incidence matrix $\mathbf{A}_e$.

Figure 2.4: Example of the illustrative simple electric circuit: (a) description of the digraph $\mathcal{G}$; (b) corresponding incidence matrix $\mathbf{A}_e$.

Let $\mathcal{V}_j$ and $\mathcal{V}_e$ be the vector spaces of currents $\mathbf{j}$ and voltages $\mathbf{e}$ of the n_E edges of a graph with n_V vertices. This graph represents a Kirchhoff network with n_E interconnected components with n_V nodes.

Definition 2.1.7 (Generalized Kirchhoff Current Law (GKCL))**.** *For any network described by a graph, the algebraic sum of the variables through (current for an electrical network) that belong to every cut-set of this graph is always equal to zero. Algebraically, with* $\mathbf{j}$ *the vectors of the through variables and* $\mathbf{A}_e$ *the incidence matrix of the graph this condition is written:*

$$\mathbf{A}_e \cdot \mathbf{j} = 0 \tag{2.7}$$

As previously noted, a connected graph with n_V vertices has only $\rho = n_V - 1$ independant GKCL, therefore (2.7) can be replaced without loss of information by the *minimal* system:

$$\mathbf{A} \cdot \mathbf{j} = 0 \tag{2.8}$$

For the illustrative electric circuit, if we delete the first line of incidence matric (vertex V0), we find a system of 3 GKCL corresponding to 3 Gauss surface surrounding the 3 vertices V1, V2 and V3 :

$$\begin{pmatrix} -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & -1 & -1 \\ 0 & -1 & 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} j_1 \\ j_2 \\ j_3 \\ j_4 \\ j_5 \end{pmatrix} = 0 \Rightarrow \begin{cases} -j_1 + j_4 = 0 \\ -j_3 - j_4 - j_5 = 0 \\ -j_2 + j_3 = 0 \end{cases}$$

It is fairly easy to show that the definition of the incidence matrix in terms of *vertex to edge incidence* can also be used to fully describe the relationships between potential variables and voltage variables. Indeed, the voltages, defined along the edges, being defined as differences between the potentials at the vertices, their description involves the *dual* operator of the incidence operator. In terms of matrix, the dual operator is none other than the transposed matrix, so that by grouping

the potentials at nodes into a vector of potentials $\mathbf{\Phi}_e = (\phi_{V0}\ \phi_{V1} \cdots \phi_{S_{n_V}-1})^T$ and branch voltages in a voltage vector $\mathbf{e} = (e_1\ e_2 \cdots e_{n_E})^T$, we obtain :

$$\mathbf{e} = \mathbf{A}_e^T \cdot \mathbf{\Phi}_e \tag{2.9}$$

If we set one of the potentials to zero, generally the reference potential associated with the vertex deleted in the reduced incidence matrix $\mathbf{A}$ and note $\mathbf{\Phi}$ the vector of potentials without the reference potential, we get the relationship :

$$\mathbf{e} = \mathbf{A}^T \cdot \mathbf{\Phi} \tag{2.10}$$

For the example of the electrical circuit, we obtain the following relationships, which are easily verified on the graph:

$$\begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & 0 & -1 \\ 0 & 0 & -1 & 1 \\ 1 & 0 & -1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \phi_0 \\ \phi_1 \\ \phi_2 \\ \phi_3 \end{pmatrix} \Rightarrow \begin{cases} e_1 = \phi_0 - \phi_1 \\ e_2 = \phi_0 - \phi_3 \\ e_3 = -\phi_2 + \phi_3 \\ e_4 = \phi_1 - \phi_2 \\ e_5 = \phi_0 - \phi_2 \end{cases}$$

The incidence matrix therefore makes it possible to express the complete set of interconnection equations of a lumped components network described by digraph. The equilibrium equations, or GKCL, are expressed by the relationships (2.7) or (2.8), in a form called *Kernel*. The compatibility equations, or Generalized Kirchoff Voltage Law (GKVL), are expressed by the relationships (2.9) or (2.10), in a form called *image*.

Alternatively there is a second way of describing algebraically interconnections on a graph. This second method uses the notion of mesh.

Definition 2.1.8 (Surface and Mesh in a graph)**.** *A mesh – or a loop – is a connected subgraph of a graph $\mathcal{G}$ that has exactly two incident edges at each vertex. A mesh is oriented. An elementary mesh is a mesh whose surface defined by its edges does not contain any inner or outer edges. Elementary meshes are boundaries of surfaces of the graph.*

The interconnection topology of an algebraically oriented linear graph can be fully described by indicating for each elementary mesh, which edges are or are not part of this mesh and whether or not the orientation of these edges are coincident with the orientation of the mesh. This information can be grouped in a table called the matrix of the incidence *edges to meshes*, or more simply *mesh matrix.*

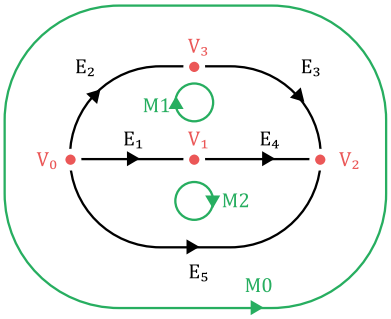
Definition 2.1.9 (Extended mesh matrix $\mathbf{M}_e$)**.** *An "extended" Mesh matrix $\mathbf{M}_e$ of a digraph $\mathcal{G}$ is a matrix with n_M lines associated to meshes Mi ($0 \leq i \leq n_M - 1$) and n_E columns associated to edges Ej ($1 \leq j \leq n_E$) such that its m_{ij} components are :*

- $\star$ *$m_{ij} = +1$ if the Ej edge belongs to and has the same orientation as the Mi elementary mesh,*
- $\star$ *$m_{ij} = -1$ if the Ej edge belongs to and has the inverse orientation of the Mi elementary mesh,*

⋆ $m_{ij} = 0$ *if the Ei edge does not belongs to the Mi elementary mesh.*

The rank of this matrix is $\nu = n_M - 1 = n_E - \rho = n_E - (n_V - 1)$ *whereas* ν *is called the nullity of the graph.*

For the illustration electrical circuit, $\nu = 2$ and its mesh matrix is reported on the FIG. 2.5.



$$\mathbf{M}_e = \begin{array}{c} \\ M0 \\ M1 \\ M2 \end{array} \begin{array}{c} \begin{array}{ccccc} E1 & E2 & E3 & E4 & E5 \end{array} \\ \begin{pmatrix} 0 & -1 & -1 & 0 & 1 \\ -1 & 1 & 1 & -1 & 0 \\ 1 & 0 & 0 & 1 & -1 \end{pmatrix} \end{array}$$

(a) Description of the digraph. (b) Mesh matrix $\mathbf{M}_e$.

Figure 2.5: Exemple of the electric circuit: (a) description of the digraph $\mathcal{G}$; (b) Corresponding mesh matrix $\mathbf{M}_e$.

Definition 2.1.10 (Reduced Mesh Matrix $\mathbf{M}$)**.** *Matrice* $\mathbf{M}_e$ *being of rank* ν *for* $n_{cs} = \nu + 1$ *lines, one of these lines is a linear combination of others and can be skipped without information loss. The deleted line generally corresponds to the one associated with the external mesh. The resulting matrix* $\mathbf{M}$ *is called the reduced mesh matrix of the graph.*

Mesh matrices (reduced or not) allow a systematic definition of generalized Kirchhof laws of the network.

Definition 2.1.11 (Generalized Kirchhoff Voltage Law (GKVL))**.** *For any network described by a graph, the algebraic sum of the across variables (voltage for an electrical network) along any mesh of this graph is always equal to zero. Algebraically, with* $\mathbf{e}$ *the vectors of the across variables and* $\mathbf{M}_e$ *the mesh matrix of the graph this condition is written:*

$$\mathbf{M}_e \cdot \mathbf{e} = 0 \tag{2.11}$$

As previously noted, a connected graph with n_M elementary mesh has only $\nu = n_M - 1$ independant GKVL, therefore (2.11) can be replaced without loss of information by the *minimal* system:

$$\mathbf{M} \cdot \mathbf{e} = 0 \tag{2.12}$$

For the illustration electric circuit, by removing the first line of the mesh matrix (mesh M0), we find a set of 2 KVL corresponding to the two inner meshes M1

and M2 :

$$\begin{pmatrix} -1 & 1 & 1 & -1 & 0 \\ 1 & 0 & 0 & 1 & -1 \end{pmatrix} \cdot \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \end{pmatrix} = 0 \Rightarrow \begin{cases} -e_1 + e_2 + e_3 - e_4 = 0 \\ e_1 + e_4 - e_5 = 0 \end{cases}$$

It is quite easy to show that the definition of the mesh matrix can also be used to fully describe the relationships between edge currents and mesh currents. Indeed, since mesh currents are the currents flowing in the edges constituting each subgraph based on the definition of an elementary mesh, their description involves the *dual* operator of the mesh operator. In terms of matrix, the dual operator is none other than the transposed matrix, so that by grouping the mesh currents in a vector $\mathbf{i}_e = (i_{M0}\ i_{M1} \cdots i_{Mn_M-1})^T$, we obtain the relationship :

$$\mathbf{j} = \mathbf{M}_e^T \cdot \mathbf{i}_e \tag{2.13}$$

If we eliminate the redundant outer mesh by using the reduced mesh matrix $\mathbf{M}$ and note $\mathbf{i}$ the vector of the reduced mesh currents, we obtain the relationship :

$$\mathbf{j} = \mathbf{M}^T \cdot \mathbf{i} \tag{2.14}$$

For the illustrative electrical circuit, we obtain the following relationships, which are easily verified on the graph:

$$\begin{pmatrix} j_1 \\ j_2 \\ j_3 \\ j_4 \\ j_5 \end{pmatrix} = \begin{pmatrix} 0 & -1 & 1 \\ -1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & -1 & 1 \\ 1 & 0 & -1 \end{pmatrix} \cdot \begin{pmatrix} i_0 \\ i_1 \\ i_2 \\ i_3 \end{pmatrix} \Rightarrow \begin{cases} j_1 = -i_1 + i_2 \\ j_2 = -i_0 + i_1 \\ j_3 = i_0 + i_1 \\ j_4 = -i_1 + i_2 \\ j_5 = i_0 - i_1 \end{cases}$$

The mesh matrix therefore makes it possible to express the complete set of interconnection equations of a lumped components network described by a digraph. The equilibrium equations, or GKCL, are expressed by the relationships (2.13) or (2.14), in a form called *image*. The compatibility equations, or GKVL, are expressed by the relationships (2.11) or (2.12), in a form called *Kernel*.

The use of the $\mathbf{A}^T$ (respectively $\mathbf{M}^T$) dual matrix of the $\mathbf{A}$ (respectively $\mathbf{M}$ matrix) to describe GKVL (respectively GKCL) makes it relatively easy to show that the vector space $\mathcal{V}_j$ of the branch currents (through variables) is orthogonal to the vector space $\mathcal{V}_e$ of the branch voltages (across variables). This topological property on Kirchhoff's networks is known as Tellegen theorem.

Theorem 2.1.12 (Tellegen Theorem)**.** *In a network that conforms to Kirchhoff'generalized laws, the vector space of the across variables is orthogonal to the vector space of the through variables.*

Proof: *Let* $\mathbf{j}$ *and* $\mathbf{e}$ *be the vector of the through and across variables of a network that conforms to Kirchhoff'generalized laws, i.e. such that* $\mathbf{A} \cdot \mathbf{j} = 0$ *(with* $\mathbf{A}$*, the reduced incidence matrix) and* $\mathbf{e} = \mathbf{A}^T \cdot \mathbf{\Phi}$ *(with* $\mathbf{\Phi}$*, the vector of potentials in*

vertices of the network and $\mathbf{M}$*, the reduced mesh matrix) and let* $< , >_{\mathcal{E}}$*, be the inner product defined between* across *and* through *variables (the energetic product of duality used in (2.4)) :*

$$\begin{aligned} < \mathbf{j}, \mathbf{e} >_{\mathcal{E}} &=< \mathbf{j}, \mathbf{A}^T \cdot \mathbf{\Phi} >_{\mathcal{E}} \\ &=< \mathbf{A} \cdot \mathbf{j}, \mathbf{\Phi} >_{\mathcal{E}} \\ &=< 0, \mathbf{\Phi} >_{\mathcal{E}} \\ &= 0 \quad \forall \mathbf{j} \in \mathcal{V}_j \text{ and } \forall \mathbf{e} \in \mathcal{V}_e \\ &\Rightarrow \mathbf{j} \perp \mathbf{e} \end{aligned}$$

The Tellegen theorem is a topological property of networks that can be described using oriented linear graphs. This theorem makes it possible to show energy conservation in any Kirchhoff network since the scalar product $< \mathbf{j}, \mathbf{e} >_{\mathcal{E}}$ is none other than the sum of the powers consumed by all the components of this network. This theorem also implies an orthogonality between the incidence and mesh matrices:

$$\mathbf{M} \cdot \mathbf{A}^T = 0 \text{ and } \mathbf{A} \cdot \mathbf{M}^T = 0 \tag{2.15}$$

It is important to note that the two ways of describing algebraically the topological structure of a network – using the incidence matrix or the mesh matrix – are actually dual to each other. This duality does not refer here to the product of energetic duality $< , >_{\mathcal{E}}$ seen previously but to a dual graph of $\mathcal{G}$. In this case, the graph $\mathcal{G}$ is called primal and its dual is noted $\mathcal{G}^*$.

Definition 2.1.13 (Dual graph, co-vertices and co-edges)**.** *Let* $\mathcal{G}$ *be a digraph, described by a set of* n_V *vertices,* n_E *edges and* n_M *surfaces (elementary meshes). A co-vertex is a point located within each elementary mesh. There are* n_M *distinct co-vertices on this graph. Co-edges are defined as follows : each edge can be cut by a co-edge that connects two co-vertices. There are* n_E *distinct co-edges on this graph because there are as many co-edges as there are edges. The full set of co-vertices and co-edges defines a new graph called* $\mathcal{G}^*$*, the dual graph of the primal graph.*

The duality between $\mathcal{G}$ and $\mathcal{G}^*$ induces a duality between their respective incidence and mesh matrices. Indeed, when the indices and orientations of the different entities (vertices, edges, meshes, co-vertices, co-edges and co-meshes) are chosen in a coherent way, we can show that $\mathbf{A}_e = \mathbf{M}_e^*$ and $\mathbf{M}_e = \mathbf{A}_e^*$. Thus the algebraic description of the topology of the dual graph results directly from that of the primal graph (and vice versa). The construction of the dual graph of the illustrative electrical example is shown in Figure 2.6.

Extension to a broader field

The concepts and definitions we have just presented on this simple example of electrical circuit are well-known concepts in electrical engineering and do not present any new features. In particular, most of this content can be found in [Desoer and Kuh, 1969a]. They have been presented in this dissertation to make this document readable by people outside the electrical engineering community. This allowed us to recall the main topological properties of planar networks. In

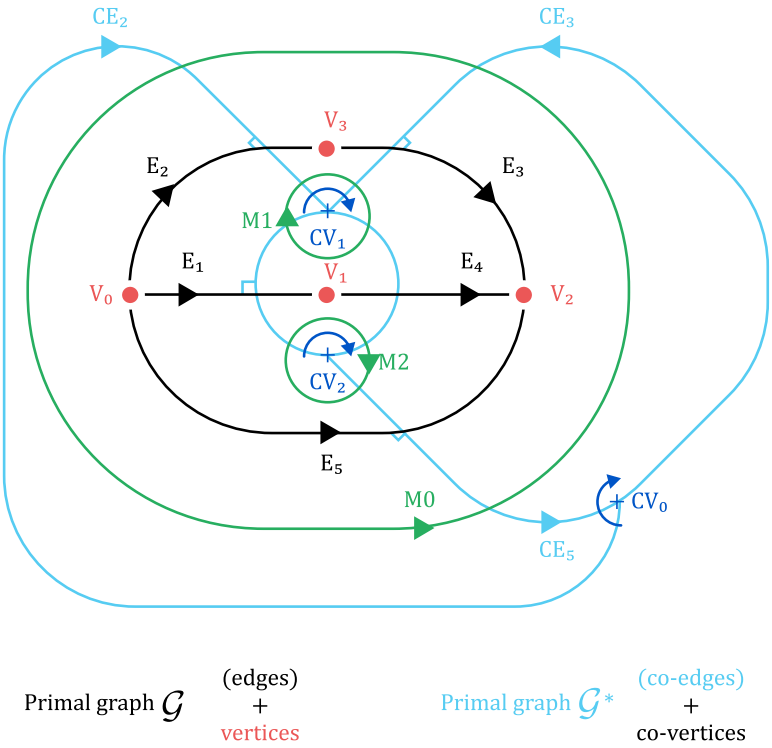


Figure 2.6: Description of the primal and dual graph of the illustrative electrical example.

summary, we can say that in planar topology (2D), there is three king of directed geometric entities: vertices, edges and surfaces. Moreover, an isomorphism, i.e. a one-to-one mapping, exists between lumped physical systems (physical quantities) and digraph (geometric entities) [Trent, 1955]. Then, physical quantities can be linked with these geometric entities : in circuit theory, electric potential ϕ are attached *to* node/vertex, voltage e are defined *across* nodes, currents j flow *through* the cross-section of edges/branch (or along the edges) and mesh currents i *curl* along the boundaries of surfaces/meshes.

When a planar graph $\mathcal{G}$ is defined, it is possible to define its dual $\mathcal{G}^*$. In such a case and for electric network, electric charges Q are defined on surfaces of this dual graph (these *dual* surfaces surround *primal* vertices and then correspond to Gauss surfaces), currents j flow *through* its edges (*dual* edges are perpendicular to *primal* edges and then parallel to their cross-section) and mesh currents i curl around vertices of this dual graph (*dual* vertices lie inside *primal* surfaces).

When a digraph (and its dual) is defined, it is possible to deduce an algebraic structure of its topology. This is done using incidence matrices between geometric entities and their boundaries. Corresponding operators are called boundary operators and for planar graph, they correspond to incidence matrix $\mathbf{A}$ (boundary operator between primal vertices and edges) and the transpose $\mathbf{M}^T$ of the mesh matrix $\mathbf{M}$ (the boundary operator of the primal graph is $\mathbf{M}^T$ because it must map meshes-to-edges and not edges-to-meshes as do the mesh matrix $\mathbf{M}$). Because of the duality between primal and dual graph, it must be stressed that

the incidence matrix of the primal graph *is* the mesh matrix of the dual graph: $\mathbf{A} = \mathbf{M}^*$. In the same way, the mesh matrix of the primal graph *is* the incidence matrix of the dual graph: $\mathbf{M} = \mathbf{A}^*$. Therefore, on the primal graph, boundary operators are $\mathbf{A}$ and $\mathbf{M}^T$ whereas, on the dual graph, boundary operators are $\mathbf{A}^* = \mathbf{M}$ and $\mathbf{M}^{*T} = \mathbf{A}^T$.

In addition, there is a duality between geometric and physical quantities, then the algebraic structure of the network can be used for describing the topological properties of physical quantities. These topological properties are known as *Kirchhoff laws*. As a consequence, these equations used *co-boundary operators*, i.e. the duals of boundary operators. If boundary operators are written as matrices, their duals correspond to their transpose. In such a way, we obtain for lumped components networks:

$$\begin{cases} \mathbf{e} = \mathbf{A}^T \cdot \mathbf{\Phi} & \text{(quantities defined on the primal graph)} \\ \mathbf{j} = \mathbf{A}^{*T} \cdot \mathbf{i} & \text{(quantities defined on the dual graph, with } \mathbf{A}^{*T} = \mathbf{M}^T\text{)} \end{cases} \tag{2.16}$$

because :

- across variables (voltages) are defined on edges, (electric) potential on their boundaries (vertices) and $\mathbf{A}$ is the boundary operator edges-vertices;
- mesh throught variables (mesh currents) are defined on oriented surface (meshes), through variables (currents) on their boundaries (edges) and $\mathbf{M}^T$ is the boundary operator meshes-surfaces.

The first equation is known as Generalized Kirchhoff Voltage Law (GKVL) and the second one as Generalized Kirchhoff Current Law (GKCL). These methods initially developed for electricity can be extended to other physics if through and across quantities can be defined, as for thermal, mechanics or fluidic systems [Rowell and Wormley, 1997]. In (2.16), the topologic set of equations is written in *Image* (or *Range-Space*) form but it can also be written in *Kernel* (or *Null-Space*) form due to duality between a graph and its dual ($\mathbf{M} \perp \mathbf{A}^T$ and $\mathbf{A} \perp \mathbf{M}^T$):

$$\begin{cases} \mathbf{e} = \mathbf{A}^T \cdot \mathbf{\Phi} \Rightarrow \mathbf{M} \cdot \mathbf{e} = \underbrace{\mathbf{M} \cdot \mathbf{A}^T}_{=0} \cdot \mathbf{\Phi} = 0 \\ \mathbf{j} = \mathbf{M}^T \cdot \mathbf{i} \Rightarrow \mathbf{A} \cdot \mathbf{j} = \underbrace{\mathbf{A} \cdot \mathbf{M}^T}_{=0} \cdot \mathbf{i} = 0 \end{cases} \tag{2.17}$$

Thus, Kirchhoff laws in Kernel form are written as:

$$\begin{cases} \mathbf{M} \cdot \mathbf{e} = 0 \\ \mathbf{A} \cdot \mathbf{j} = 0 \end{cases} \tag{2.18}$$

This is the classic writing of Kirchhoff current and voltage laws. To illustrate these topologic structures, a graphic was proposed by Tonti, that is now called a *Tonti diagram*. Such a diagram is proposed in Figure 2.7 for electric circuit (this figure is inspired by [Alotto et al., 2013]). Physical quantities in the left part of this diagram belong to configuration variables defined on primal graph. Physical quantities in the right part of the diagram belong to configuration variables defined on dual graph. The central part describes constitutives equations between

configuration and source variables. Examples of diagrams for other physics will be presented in next sections but E. Tonti in [Tonti, 2013] resumes most of them for different physical theories. In the framework of Tonti diagrams, nodal, mesh, loop or cut-set methods for network computations (see [Desoer and Kuh, 1969a] for examples in electrical engineering) are just different way to browse this diagram to solve the corresponding physical equations.

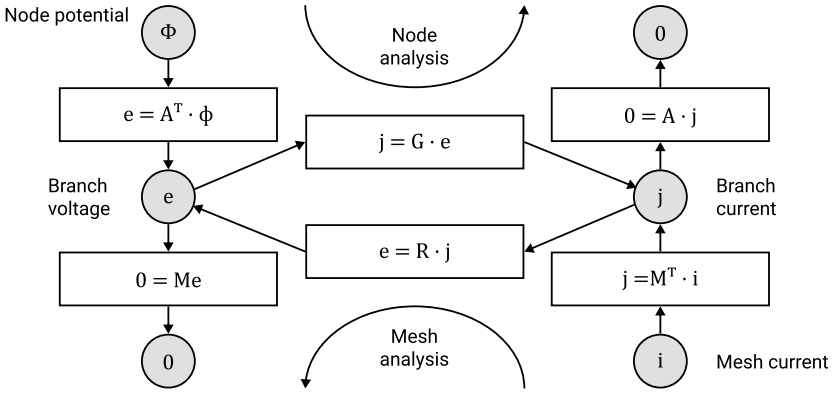


Figure 2.7: Tonti diagram of electrical circuits: two ways to travel it, using $\mathbf{\Phi}$, potentials at nodes (Node analysis) or using $\mathbf{i}$, mesh currents (mesh analysis).

If there are voltage or current sources ($\mathbf{e}_s$ or $\mathbf{j}_s$) in several branches of the electric network, it must be modified as shown in Figure 2.8. It should be noted that only the constitutive laws are affected by this inclusion, not the topological laws.

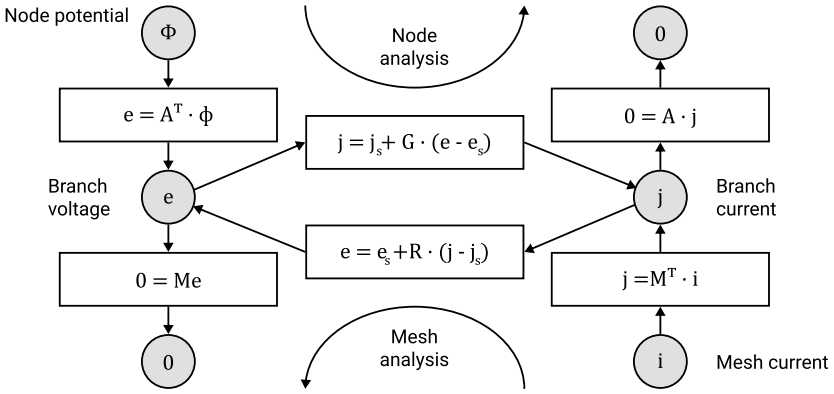


Figure 2.8: Tonti diagram of electrical circuits with current and voltage sources.

As shown in Figures 2.7 and 2.8, the node (or nodal) analysis uses the electrical potentials $\mathbf{\Phi}$ as degrees of freedom and the equations described in the upper part of the Tonti diagram:

$$\begin{cases} \mathbf{A} \cdot \mathbf{j} = \mathbf{0} \\ \mathbf{j} = \mathbf{j}_s + \mathbf{G}(\mathbf{e} - \mathbf{e}_s) \\ \mathbf{e} = \mathbf{A}^T \cdot \mathbf{\Phi} \end{cases} \tag{2.19}$$

By combining these three equations, we obtain the following linear system in $\mathbf{\Phi}$ to be solved:

$$(\mathbf{A} \cdot \mathbf{G} \cdot \mathbf{A}^T) \cdot \mathbf{\Phi} = (\mathbf{A} \cdot \mathbf{G} \cdot \mathbf{e}_s - \mathbf{A} \cdot \mathbf{j}_s) \tag{2.20}$$

$(\mathbf{A}\cdot\mathbf{G}\cdot\mathbf{A}^T)$ is similar to a *stiffness* matrix of FEM and $(\mathbf{A}\cdot\mathbf{G}\cdot\mathbf{e}_s - \mathbf{A}\cdot\mathbf{j}_s)$ corresponds to source terms. This system can be solved in a unique way by setting boundary conditions on known electric potential ϕ_{BC}. If extended incidence matrices are used, a boundary condition must be imposed in at least one node, usually the node of the ground/reference potential. As soon as the electric potentials are known, all other quantities ($\mathbf{e}$ and $\mathbf{j}$) can be deduced from them.

The mesh analysis is the dual version of the node analysis. It uses the mesh currents $\mathbf{i}$ as degrees of freedom and the equations described in the lower part of the Tonti diagram:

$$\begin{cases} \mathbf{M} \cdot \mathbf{e} = \mathbf{0} \\ \mathbf{e} = \mathbf{e}_s + \mathbf{R}(\mathbf{j} - \mathbf{j}_s) \\ \mathbf{j} = \mathbf{M}^T \cdot \mathbf{i} \end{cases} \tag{2.21}$$

By combining these three equations, we obtain the following linear system in $\mathbf{i}$ to be solved:

$$(\mathbf{M} \cdot \mathbf{R} \cdot \mathbf{M}^T) \cdot \mathbf{i} = (\mathbf{M} \cdot \mathbf{R} \cdot \mathbf{j}_s - \mathbf{M} \cdot \mathbf{e}_s) \tag{2.22}$$

This system can be solved in a unique way by setting boundary conditions on known mesh current i_{BC}. If extended mesh matrices are used, a boundary condition must be imposed in at least one mesh, usually the outer mesh. As soon as the mesh currents are known, all other quantities ($\mathbf{e}$ and $\mathbf{j}$) can be deduced from them.

These two methods, or a combination of these two (as used in loop or cut-set methods), are implemented on *Spice-like softwares*. The choice between these different methods depends on the number of degrees of freedom for each calculation, the type of boundary conditions or sources and the causality of components if transient calculations have to be computed. For transient computation, State-Space forms have to be used and solved using a EDO solver as explained in [Chua and Lin, 1975].

The strength of this modeling methodology is that the topology of any interconnection between subsystems can be defined using algebraic tools (discrete boundary operators). Moreover, because of the duality between geometric and physical quantities, this algebraic structure describes also the topologic properties between physical quantities. To resume: as soon as the geometry of the network is known, the topology of its physical equations can be deduced using dual discrete operators. Mathematicaly, this special structure is known as homology/co-homology structures [Munkers, 1984, Hatcher, 2002].

2.1.3 Field computation for third level models

To understand the physical working of complex devices (with complex geometry), it is necessary to compute the distribution of physical fields in a continuum

space, both inside the device and in its environment. For mechatronic and multiphysic systems, these fields arise from mechanics (stress, strain, displacement,...), thermal physics (temperature, energy, heat,...) and electromagnetics (magnetic, electric,...). Physical fields are usually modeled by Partial Differential Equations (PDE) based on conservation laws (mass, momentum, charge, energy,...). Nevertheless, PDE equations are difficult (or even impossible) to solve analytically for complex geometries. Therefore, for engineering applications, approximate solutions must be found. This is the purpose of numerical simulation tools. In this frame, three main numeric methods are currently used: Finite Difference Methods (FDM), Finite Volume Methods (FVM) and Finite Elements Methods (FEM)[2]. With several differences, each of these three approximate methods of fields computation presents these following main characteristics:

- Accurate representation of complex geometry,
- Inclusion of dissimilar material properties,
- Easy representation of the total solution,
- Capture of local effects.

All three methods are based on discretization of the space variables (and discretization of time for transient analysis) but their working are different. FDM is based on strong – differential – form of PDE and discretization of differential operators. At the contrary, FVM and FEM are based on a weak – integral – form of the PDE. Let us resume the main characteristics of each of these approximate solving methods.

- **FDM :** The FDM is the oldest of the three methods and is based upon the application of a local Taylor expansion to approximate the differential equations. The FDM uses a square network (a *structured grid*) to construct the discretization of the PDE. The discretization results in a system of equations at nodal points, and once a solution is found, then we have a discrete representation of the solution. One of the main drawback of FDM is that it requires the use of structured grid and these one are not adapted to approximate accurately complex geometries. Through the use of curvilinear transform, the method can be extended to domains that are not easily represented by brick-shape elements, but these transforms are complex and are seldom used in practice. In this way, FDM method are ever less used in engineering, except for 1D problems or for time discretization in transient problems. A significant exception exists in electrodynamics and wave computations: corresponding FDM methods are known as *Finite Difference Time Domain Method* (FDTD) [Yee, 1966a, Taflove and Hagness, 2005].
- **FVM :** A FVM discretization uses an integral forms of the PDES, e.g. conservation of mass, momentum or energy. The PDE is written in a form which can be solved for a given finite volume (or *cell*). The computational

[2]There is a fourth numeric method, the Boundary Element Method (BEM), but this method is not considered in this work because it is mainly dedicated to computation of wave propagation problems. BEM applies a transformation of PDE over volumes into equations over boundaries, i.e. surfaces, using Gauss or Divergence theorems.

domain is discretized into finite volume and then the governing equations are solved for every volume. The resulting system of equations usually involves fluxes of the conserved variable, and thus the calculation of fluxes is very important in FVM. The basic advantage of this method over FDM is that it does not require the use of structured grids, but arbitrary *mesh*. As with FDM, the resulting approximate solution is a discrete representation, but the variable are typically placed at cell centers rather than at nodal points: properties are calculated for every cell instead of at every node. In any case, the value of fields at any location are obtained using interpolation. As FVM are based on integral form of conservation laws, they can handle discontinuities in solutions and provide better conservation properties. They are very efficient in solving fluid flow problems.

- **FEM :** A FEM discretization is based upon a piecewise representation of the solution in terms of specified basis functions. The computational domain is divided into smaller domains (finite elements) and the solution in each element is constructed from the basis functions. The actual solved equations are typically obtained by restating the conservation equations in weak form: the field variables are written in terms of the basis functions, the equation is multiplied by appropriate test functions, and then integrated over an element. Again a system of equations is obtained and solved to obtain a solution. Usually, the degrees of freedom correspond to nodal values (for first order interpolation functions) but this can be extended for edge values (Nédélec or Edge elements, [Nedelec, 1980]) or surface values [Raviart and Thomas, 1977]. This extension provides better solutions, especially in electromagnetism, but are more difficult to code. Shape functions or basis functions are used to interpolate the field inside each finite element. Generally linear interpolations are acceptable. If this is not the case, quadratic or cubic shape functions can also be use at the expense of computational time (edge or surface elements may be prefered in such a case). FEM is the most commonly used numerical method and is efficient for all geometries including these with complicated shapes and features.

As noticed above, the most common method for field computation in engineering, i.e. modelling at level 3, is then the FEM. The popularity of FEM is explained by its important feature and also by the history of engineering software and computer graphics. This method can be applied to domains of arbitrary shapes with arbitrary boundary conditions. FEM method, by its nature, leads to unstructured meshes. This was the main advantage that determined popularity of FEM. Despite a wide variety of numerical computation schemes proposed in the 1960s, all except the finite element method, (1) required more sophisticated mesh and computation tools to obtain better results than FEM and (2) were applied with regular structural meshes that poorly approximates the geometry of real objects. Today, due to computer graphics progress, several methods, including FVM, propose similar *arbitrary-shape* features. Therefore, corresponding methods will be refered in the sequel as FEM-*like* Methods.

2.2 Toward multiphysic and topologic models

The aim of this section is to propose a type of model that shares the main features presented previously, e.g. (1) being multiphysic, (2) including topological properties and (3) allowing field computation. The key factor of this possibility was proposed by visionary applied mathematicians. Among them, we can cite E. Cartan, H. Whitney and J. E. Marsden. Our work was greatly influenced by their work and vision. E. Tonti and its "cell-method" also had a major influence on our work, which is why this method will be briefly presented in the next subsection, including its main characteristics and features.

In a previous subsection, we have shown that a good way to model multiphysic systems is by adopting an energy approach based on energetic scalar product (or duality product). This is the main feature, we want to retain for the following developments. When proposing its famous diagrams, Enzo Tonti in his pioneer works on *Analogies between physical theories* [Tonti, 1976a] was actually extending a vision proposed long before by thermodynamics. Its duality was not limited to a simple energy duality but it allows a larger classification of physical quantities as *configuration* or *source* variables. In the same way, this permits to stress clearly the difference between three types of equations in *every* physical theory (see Figure 2.9):

1. Compatibility equations (or Generalized Kirchhoff Voltage Law) seen as *linear constraints* between configuration variables.

2. Equilibrium equations (or Generalized Kirchhoff Current Law) seen as *linear constraints* between source variables.

3. Constitutive equations seen as *phenomenological maps* between source and configuration variables.

The first two equations are topological equations and must be preserved *stucturally* to ensure the consistency of conservation laws. The third equation is the metrics of the problem and may be approximated (or relaxed), because phenomenological equations are just approximation of reality. Although this distinction between these three kinds of laws is not a recent discovery (they are used for a long time in the field of engineering, see [Cannon, 2012] for example), the fact that it can be presented so clearly is, from our point of view, what corresponds to the essential contribution of Tonti's diagrams. These are the reasons why the numerical tools we have developed and which will be presented in the rest of the manuscript are so close to the cell-method [Tonti, 2001a].

2.2.1 A few words on the Cell-Method approach

There are many different numerical schemes based on discrete differential forms and the exterior calculus framework, but the Cell-Method is probably the most developed in terms of a multiphysics approach. For more details, we will refer to Alotto et al.'s book [Alotto et al., 2013] where the theoretical and practical aspects of Tonti's ideas are particularly well summarized. For a more in-depth theoretical basis, the books by E. Tonti [Tonti, 2013] or by E. Ferreti [Ferretti, 2014] are very

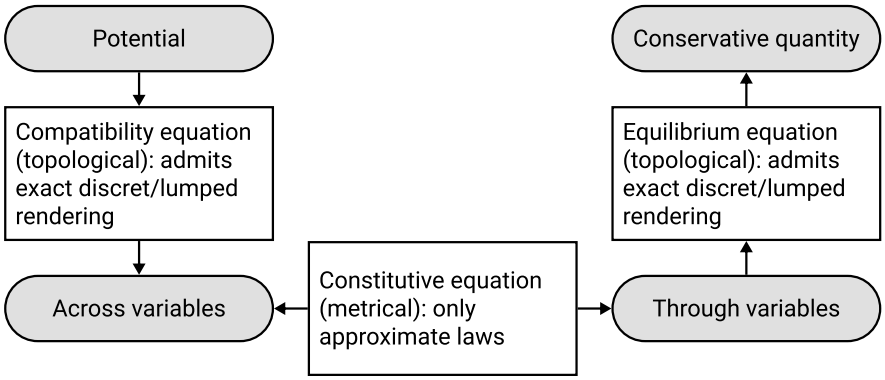


Figure 2.9: Description of the three main types of equations in the modeling of discrete/lumped physical system.

good references. Contrary to what is often believed, E. Ferreti's book shows that the Cell-Method is really based on a very solid mathematical and theoretical framework. In addition, the Tonti diagram approach and the use of a geometric and topological language make it particularly intuitive and very accessible even for researchers in the engineering community who are uncomfortable with the highly abstract mathematical concepts of differential geometry and exterior differential calculus.

As we have seen previously, Tonti diagrams are based on the idea that all physical phenomena have the same geometric framework and that all physical variables can be classified into several classes. In a sense, the Tonti diagrams rigorously extend the idea of analogies that have long been used in physics. Thus, E. Tonti recalls it in a quote from J. C. Maxwell:

> But it is evident that all analogies of this kind depend on principles of a more fundamental nature; and that, if we had a true mathematical classification of quantities, we should be able at once to detect the analogy between any system of quantities presented to us and other systems of quantities in known sciences, so that we should lose no time in availing ourselves of the mathematical labors of those who had already solved problems essentially the same. [...] At the same time, I think that the progress of science, both in the way of discovery, and in the way of diffusion, would be greatly aided if more attention were paid in a direct way to the classification of quantities.
> J. C. Maxwell, Remarks on the mathematical classification of physical quantities Proceedings of the London Mathematical Society, 1871

R. Feynman himself, in its famous Caltech's lecture courses, also notes:

> Why are equations from different phenomena so similar? ... Is it possible that ... the thing which is common to all phenomena is the space, the framework into which the physics is put? [Feynman et al., 1965]

Space is a basic element of analogies, but it is not the only one. In fact, E. Tonti tried to study the underlying reason for analogies in physics by the existence of

a geometric and mathematical structure underlying all physical theories. This allows him to find a rational explanation for properties existing in all physical theories. He also tried to classify physical variables and physical phenomenological laws into different categories. This very general attempt at classification has made it possible to give a common framework to many different physical formalisms, including:

- The generalized network theory.
- The mathematical field theory based on variational principle.
- The irreversible thermodynamics.
- The dynamics systems.
- The first quantization for a mechanical system.
- The second quantization for a field.

Figure 2.10 shows the classification of physical quantities proposed by E. Tonti. Some elements of this classification are obviously quite common since they re-use elements of the theories of analogies already existing. Other ideas are more original, in particular those that make it possible to distribute quantities between global variables and density variables or those that make configuration or source variables appear so that their product of duality gives energy. These two types of classification will be a little more detailed in the following.

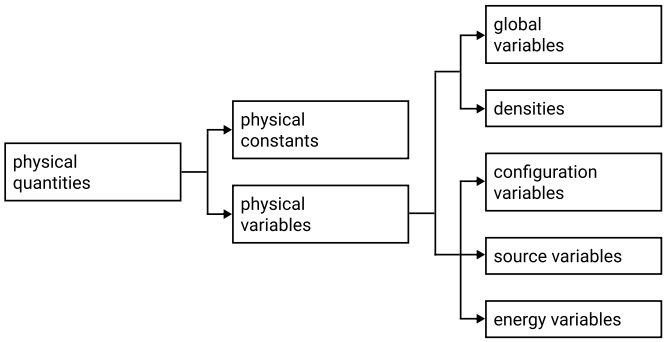


Figure 2.10: The general classification of physical variables by E. Tonti.

Global variables versus Field variables

To introduce this classification, E. Tonti starts from a question: "What are we measuring? In general, we measure "global variables" and deduce the corresponding densities from them. Let us give some examples:

- we measure a mass and we deduce the mass density,
- we measure a voltage and we deduce the electric field strength,
- we measure an electric current and after we evaluate the current density,
- we measure a force and after we evaluate a pressure or a stress,

- we measure a stretching or a displacement and we evaluate the linear strain etc.

The same applies to the classification of physical variables according to their nature, global or local. In general, global variables are variables that are neither densities nor rates of other variables [Ferretti, 2014]. This classification is quite common, but E. Tonti draws the following conclusions: a global variable can be associated with a point, a line, a surface or a volume (notation shown in Figure 2.11), while the density is always defined in points. This idea is illustrated in Figure 2.12.

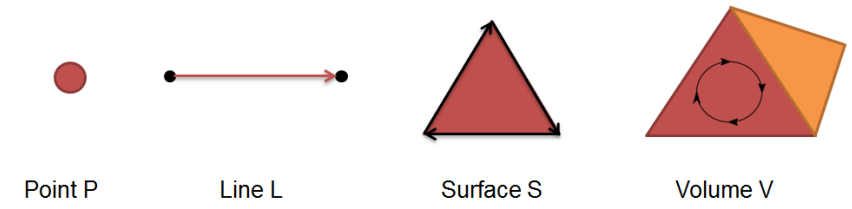


Figure 2.11: The four space elements: point, line, surface and volume.

	Global variable	Density		Global variable	Density
mass	M[V] =	$\int_V \rho(\mathrm{P})\mathrm{d}V$	entropy	M[V] =	$\int_V s(\mathrm{P})\mathrm{d}V$
current	I[S] =	$\int_S \mathrm{J}(\mathrm{P})\mathrm{d}S$	magnetic flux	Φ[S] =	$\int_S \mathrm{B}(\mathrm{P})\mathrm{d}S$
electro-motive force	E[L] =	$\int_L \mathrm{E}(\mathrm{P})\mathrm{d}L$	velocity circulation	Γ[L] =	$\int_V v(\mathrm{P})\mathrm{d}L$
temperature	T[P] =	T(P)	potential	ϕ[P] =	ϕ(P)
	Domain function	Point function		Domain function	Point function

Figure 2.12: Difference between global variables and densities.

And what about field variables? Field variables are obtained from global variables in the form of densities of global spatial variables and rates of global temporal variables. Due to their punctual nature, these are local variables. The only way not to lose the information that exists between a local variable and its correspondence with a measurement made on a point, line, surface or volume is by "coding" this correspondence directly into the local variable. But the only way to do this is actually to replace the field variables with differential forms of degree p (with p an integer between 0 and 3).

To support Tonti's contribution in relation to the classical formalism of field theory and in particular his willingness to keep this correspondence with the dimension of the measure, E. Ferretti proposes an original and very relevant argument by studying the definition of *limits* and *Cancellation rule*. Because of their rather technical nature and in order not to overload this chapter, we have prefered to report these mathematical arguments in Appendix A. Interested readers may refer to it or the book of E. Ferretti [Ferretti, 2014] for further informations. However, these arguments lead to the following conclusion:

In summary, one of the main consequences of using *classical* differential formulation instead of algebraic formulation is that we then lose the geometric dimension information associated with physical variables since we use local quantities instead of global quantities. We must note nevertheless that it is not the case if we use differential forms instead of *classical* fields. For numerical computation, E. Tonti therefore recommends using the algebraic topology framework and global variables rather than the classical differential calculus framework, this is what led him to develop the cell-method. To conclude, in [Tonti and Zarantonello, 2009], E. Tonti makes the following arguments for the benefit of using global variables in algebraic formulation:

- A global variable is continuous across the separation surface of two materials, e.g. displacements and surface forces,
- We do not need jump conditions. The jump conditions regard the field functions and they are derived from the continuity of local variables, e.g. the strains,
- Singularities do not arise. In fact, singularities come from the ratio between a finite quantity and an infinitesimal extension, typically an area or a volume. Since an algebraic formulation does not perform the limit, it is free of singularities. Hence in the apex of a fracture (mechanical example) and at the point of application of a concentrated force the stress remains finite,
- Global variables are, in general, quantities measured in laboratory, while the corresponding densities are deduced from the global quantities.

In fact, global quantities proposed by E. Tonti are nothing more than discrete differential forms proposed by mathematicians [Hirani, 2003]. This concept is therefore well defined from a mathematical point of view. Tonti's presentation of it is, however, very intuitive and therefore allows an easy introduction to engineering and applied physicists.

Configuration versus source variables

This type of classification play a central role in the Cell-method (CM) and in our work detailled in next chapters. According to this classification we can divide all physical variables in 3 classes (see Figure 2.13):

- *Source variables* are the variables witch describes the source of the field or the forces acting on the system. This variables are the origin of a physical interaction in the system. Example of this variables are forces for solid mechanics and fluidodynamics, masses for geodesy, electric charges for electrostatics, electric currents for magnetostatics, heat sources for thermal conduction. They are defined on the dual space with an *external* orientation [Ferretti, 2014].
- *Configuration variables* are the variables with describes the configuration of physical system or a field. Example of this variables are displacements for solid mechanics, spatial velocity for fluidodynamics, electric potential for electrostatics, temperature for thermal conduction. They are defined on the primal space with an *internal* orientation [Ferretti, 2014].

- *Energy variables* are the variables obtained as a product of source variable times configuration variable. Example of this variables are elastic energy density for solid mechanics, kinetic energy for dynamics, electrostatic energy for electrostatics, magnetostatic energy for magnetostatics, heat for thermal conduction.

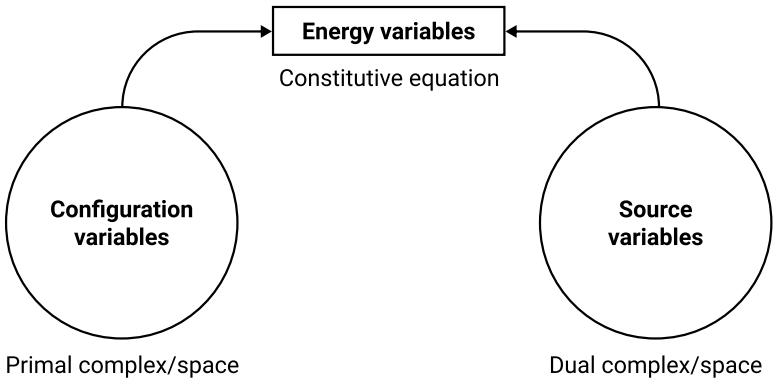


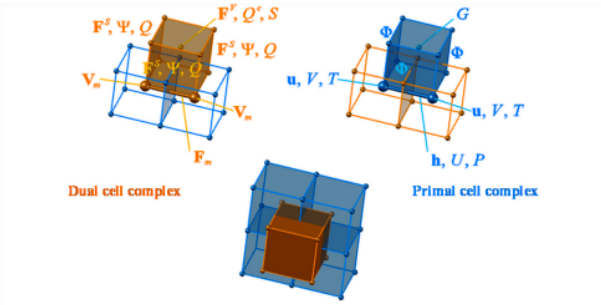
Figure 2.13: The second classification of physical variables

This clear and simple classification provides far-reaching benefits for multiphysic modelling. This helps to understand the connection we haven't seen so far. In fact, we pay a lot of attention to the primal and dual spaces[3] but without necessary explanation. Here, we have the answer ! In fact, the configuration variables are associated with the primal geometrical space (a graph, a complex or a mesh) and the source variables are associated to the corresponding dual geometrical space. The connection between these two dual spaces is assured by the *constitutive* equations witch introduce a metric space in these spaces. The constitutive equations are the phenomenological physical laws linking two energetically dual variables. Topological equations are constraints between global variables defined in the same space, configuration *or* source space. If these global variables are defined into the configuration/primal space, they are *even* or *straight* discrete différential forms otherwise they are *odd* or *twisted* discrete differential forms if they are defined into the source/dual space (see Figure 2.14).

As seen previously, topological equations contains co-boundary operators. Let $\mathcal{M}$ be a space element (point, line, surface or volume) and let be $\partial\mathcal{M}$ be its boundary. Topological equations in the primal cell complex are coboundary processes on even exterior discrete p-forms, while the topological equations in the dual cell complex are coboundary processes on odd exterior discrete p-forms. Dual coboundary operators must therefore be defined on both primal and dual spaces as shown in Figure 2.15.

Algebraic topology is a mathematical theory that provides a rigorous foundation for cell-method. The following section provides an accessible summary for engineers.

[3]For a network, these dual spaces are the primal oriented graph and its dual. In the next chapter devoted to numeric computation of fields, we will see that these dual spaces correspond to a primal mesh (called a *complex*, in algebraic topology) and its dual mesh.



Elastic field

The volime force $\mathbf{F}^{\mathrm{V}}$ is referred to Volumes $\widetilde{\mathrm{V}}$ of the dual cell complex

The surface force $\mathbf{F}^{\mathrm{S}}$ is referred to Surfaces $\widetilde{\mathrm{S}}$ of the dual cell complex

The displacement $\mathbf{u}$ is referred to Points $\overline{\mathrm{P}}$ of the primal cell Complex

The relative displacement $\mathbf{h}$ is referred to Lines $\overline{\mathrm{L}}$ of the primal cell Complex

Electric field

The electric charge content Q^{L} is referred to Volumes $\widetilde{\mathrm{V}}$ of the dual cell complex

The electic flux Ψ is referred to Surfaces $\widetilde{\mathrm{S}}$ of the dual cell complex

The potential V is referred to Points $\overline{\mathrm{P}}$ of the primal cell complex

The tension $\boldsymbol{U}$ is referred to Lines $\overline{\mathrm{L}}$ of the primal cell Complex

Magnetic field

The magnetic tension F_{m} is referred to Lines $\widetilde{\mathrm{L}}$ of the dual cell complex

The magnetic potential V_{m} is referred to Points $\widetilde{\mathrm{P}}$ of the dual cell complex

The magnetix flux $\boldsymbol{\Phi}$ is referred to Surfaces $\overline{\mathrm{S}}$ of the primal cell complex

The magnetix charge G is referred to Volumes $\overline{\mathrm{V}}$ of the primal cell complex

Thermal field

The heat production S is referred to Volumes $\widetilde{\mathrm{V}}$ of the dual cell complex

The heat Q is referred to Surfaces $\widetilde{\mathrm{S}}$ of the dual cell complex

The temperature T is referred to Points $\overline{\mathrm{P}}$ of the primal cell complex

The temperature difference P is referred to Lines $\overline{\mathrm{L}}$ of the primal cell complex

Figure 2.14: Association between global variables and space elements of the primal and dual cell complexes, in different physical theories [Ferretti, 2014].

2.2.2 Topology of network for second level models

In the previous sections, algebraic topology tools were presented on graphs (planar spaces). They were used to model lumped systems. In this section, these tools are generalized on non-planar spaces, which allows to extend the results to field calculations on discrete three-dimensional spaces (on meshes). This work can be based on algebraic topology concepts, such as *chains*, *cochains* and *homology*/*cohomology* structures. This subsection is partly based on the presentation made by C. Mattiussi in [Mattiussi, 1997, Mattiussi, 2002]. However, some aditional definitions as well as a glossary of the main concepts of algebraic topology are summarised in appendix B.

In the framework of algebraic topology, a *chain* is a linear combination of

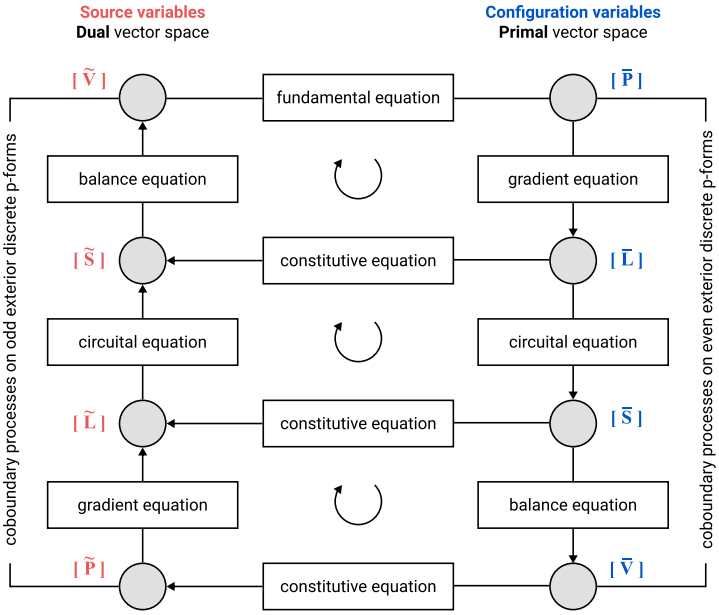


Figure 2.15: Tonti's classification diagram of the physical variables in the general case [Ferretti, 2014].

single geometric entities. This will generalized the 2D concept of oriented graph to higher dimensional entities. For computation, 3D domains must be discretized using a *mesh* made with an assembly of *cells*. In algebraic topology, 3D-mesh are usually constituted as an assembly of tetrahedrons, or tets. Because a tet is the most topologicaly simple 3D-*cell*, a tet is called a 3-*simplex*. The complete assembly of 3-simplices, i.e. the mesh, is called a 3-*complex*. Therefore a 3D-mesh is constituted with an assembly of 3-simplices. In 3D, the simplex of higher dimension is a tet (a 3-simplex). This one contains triangular surfaces (or 2-simplex) which contain edges (or 1-simplex) which contain vertices (or 0-simplex). The name p-cell (for $0 \leq p \leq 3$) is sometimes used instead of p-simplex but cells are more general than simplices because they can be based on quadrangle or more complex geometric entities than tets. Computationaly, tets are nevertheless the most usefull/efficient elementary geometric parts because they use less geometric entities, then less degrees of freedom and less computation complexity.

Chains can be seen as the algebraic counterpart of a domain oriented and weighted by an integer n. If a p-cell (or p-simplex) with index i is denoted $\mathbf{c^i}$, then a chain is defined as :

$$\mathbf{C} = \sum_i n_i \mathbf{c}^i \tag{2.23}$$

In this chain, $n_i \in \mathbb{Z}$ is the multiplicity of the cell i. For $p = 1$, a chain is a path Γ through edges of the mesh. Most of the time, the multiplicity n_i is equal to $+1$ (the chain passes along the cell number i and in the same direction), -1 (the chain passes along the cell number i and in the reverse direction) or 0 (the chain does not pass along the cell number i). This notation has some link with the intuitive idea of composing a domain by "adding" its parts. Mathematically,

a chain is in fact an element of a free module which has the cells as generators and chains generated by a given ensemble of cells can be added, subtracted, and multiplied by integer, allowing the algebraic manipulation of domains. Chains formed with $p-$cells are called p-dimensional chains or p-chains and will be noted $\mathbf{C}_{(p)} = \sum_i n_i \mathbf{c}^i_{(p)}$. Finally, chains, simplices and complex are the topological concepts used to represent the discretized geometry of a problem.

As explained in [Mattiussi, 1997]:

> "the boundary of a domain is a fundamental notion in the enunciation of physical laws and therefore it is advisable to define this concept for a chain. The boundary $\partial \mathbf{c}_{(p)}$ of an oriented p-cell $\mathbf{c}_{(p)}$ is defined as the $(p-1)$-chain composed by the $(p-1)$-cell of the cell-complex having nonempty intersection with $\mathbf{c}_{(p)}$, endowed with the orientation *induced* on them by $\mathbf{c}_{(p)}$".

Building on this definition, we can defined the procedure to calculate the boundary of a chain as a combination of its cells' boundaries :

$$\partial \mathbf{C} = \partial \left(\sum_i n_i \mathbf{c}^i \right) = \sum_i n_i \partial \left(\mathbf{c}^i \right) \tag{2.24}$$

This defines the *boundary operator* ∂ which transforms p-chains in $(p-1)$-chains. This operator is compatible with the additive and the (external) multiplicative structure of chains; in other world, it is a *linear transformation* of the module of p-chains into the module of $(p-1)$-chains over the same cell-complex:

$$\left\{ \mathbf{C}_{(p)} \right\} \overset{\partial}{\rightarrow} \left\{ \mathbf{C}_{(p-1)} \right\} \tag{2.25}$$

This boundary operator is the formal abstraction of the incidence matrices presented previously to describe the topology of networks.

As seen previously, to each oriented geometric entity (vertex, edge, surface) can be associated a physical entity (electric potential, voltage, mesh current, for example). In algebraic topology, this association is extended using the concept of *cochains*.

Given a field problem defined in a discretized region, we have to deal with various fields which reveal themselves as global quantities associated with suitable p-cells. We can therefore represent a field on a cell-complex (or more precisely the distribution of a field on a discretized domain) as a function associating global quantities with all the p-cells of a complex having a given value of p and a given kind of orientation. For example, in previous section, we define a voltage e as a physical *real* quantity associated with one geometric edge. Therefore, we can attach a voltage e^i to each 1-cell $\mathbf{c}^i_{(1)}$:

$$\mathbf{c}^i_{(1)} \overset{\text{voltage}}{\rightarrow} e^i \in \mathbb{R} \tag{2.26}$$

It will be shown in the next chapter that e^i is the discrete counterpart of $\vec{E} \cdot \vec{dl}$ integrated on the path Γ defined by the edge i: $e^i = \int_{\Gamma=\text{edge i}} \vec{E} \cdot \vec{dl}$ with $\vec{E}$ the

electric field. This simple example can be extended on every type of fields, being associated to 0-chain (electric potential field $\rightarrow$ electric potential at vertices), 1-chain (electric field $\vec{E} \rightarrow$ voltage e on edges), 2-chain (current density field $\vec{J}$ $\rightarrow$ electric current j through surfaces) or 3-chain (charge density ρ field $\rightarrow$ total electric charge Q in volumes). This allows the extension on chains for any physical quantity q such that:

$$\mathbf{C}_{(p)} = \sum_i n_i \cdot \mathbf{c}^i_{(p)} \overset{q \text{ quantity}}{\rightarrow} q^{\mathcal{C}_{(p)}} = \sum_i n_i \cdot q^i \in \mathbb{R} \tag{2.27}$$

In algebraic topology, such a transformation is called a *real-valued p-dimensional cochain* or, in short a p-cochain $\mathbf{C}^{(p)} = q^{\mathcal{C}_{(p)}}$. To emphasize the joint role of the domain and of the field in the generation of the global quantity, this is often represented as[4]:

$$< \underbrace{p^{(p)}}_{p-\text{cocell}} , \underbrace{\mathcal{C}_{(p)}}_{p-\text{chain}} >= \underbrace{\mathcal{C}^{(p)}}_{p-\text{cochain}} \in \mathbb{R} \tag{2.28}$$

$\mathcal{C}_{(p)}$ is a p-chain and $\mathcal{C}^{(p)}$ (or $q^{(p)}$ for a *single* p-cell) is a p-cochain, which is the space integration of fields over the cell-complex. It has to be noted that some cochain are not *real-valued* but *vector-valued*. This is the case in mechanics for the displacement field which is a *vector-valued 0-form*. In short, cochains constitute a representation for fields over discretized domains. As field functions can be added and multiplied by a scalar, so can cochains defined over a same cell-complex. Mathematically and collectively, all these cochains constitute a module.

The $< \cdot, \cdot >$ notation is a straightforward way to emphasize the duality between *geometric* chains and *physical* cochains but this duality product may not be confused with the energetic duality product $< \cdot, \cdot >_{\mathcal{E}}$ seen previously between *configuration* and *source* quantities. To avoid any confusion, we will note the duality product between *geometric* chains and *physical* cochains $< \cdot, \cdot >_{\mathcal{G}}$. The duality product $< \cdot, \cdot >_{\mathcal{E}}$ reveals that there is two dual *physical cochains spaces*, i.e. one for configuration variables and one for source variable. As a consequence, there must be two dual *geometric chains spaces*. For electric network, this corresponds to primal and dual graphs seen previously. This has to be generalized in the framework of algebraic topology as dual meshes. This will be considered in more details in the next chapter when we will present meshing techniques and Delaunay-Voronoi dual meshes, one being the primal mesh whereas the second being its dual.

Previously, we also defined *coboundary operators*, noted δ in the following, as dual operator of boundary operators, with respect to the dual product $< \cdot, \cdot >_{\mathcal{G}}$:

$$< q^{(p)}, \partial \mathcal{C}_{(p+1)} >=< \delta q^{(p)}, \mathcal{C}_{(p+1)} > \tag{2.29}$$

In the framework of algebraic topology, coboundary operators are a kind of generalization for Kirchhoff laws as $q^{(p+1)} = \delta q^{(p)}$:

[4]Here p-chains use bottom position indices $\mathbf{C}_{(p)}$ and p-cochains use top position indices $\mathbf{C}^{(p)}$.

- For Kirchhoff voltage laws (KVL): $e = A^T \cdot \phi$ with $A = \partial$, $A^T = \delta$ and $p = 0$ such that $q^{(p+1)} = q^{(1)} = e$ and $q^{(p)} = q^{(0)} = \phi$. This can be applied to any physical quantity derived from a potential (in electric circuit, voltages e derive from electric potentials ϕ).

- For Kirchhoff current laws (KCL): $j = M^T \cdot i$ with $M = \partial$, $M^T = \delta$ and $p = 1$ such that $q^{(p+1)} = q^{(2)} = i$ and $q^{(p)} = q^{(1)} = j$. This can be applied to any physical quantity representing flow of conservative quantities (in electric circuit, currents are charge flows $j = \frac{dQ}{dt}$ with Q *conservative* electric charges).

There are therefore two kinds of coboundary operators, one defines on the primal mesh (*Generalized* KVL) and one defines on the dual mesh (*Generalized* KCL). It is worth noting that both are dual each other because they each corresponds to dual meshes.

Anyway, a coboundary operator can be defined as an operator that transform a p-cochain into a $(p+1)$-cochain such that:

$$< \delta \mathbf{C}^{(p)}, \mathbf{C}_{(p+1)} > \stackrel{\text{def}}{=} < \mathbf{C}^{(p)}, \partial \mathbf{C}_{(p+1)} > \tag{2.30}$$

It can be shown that this is a linear transformation of the module of p-cochains into the module of $p+1$-cochains over the same cell-complex:

$$\left\{\mathbf{C}^{(p)}\right\} \stackrel{\delta}{\rightarrow} \left\{\mathbf{C}^{(p+1)}\right\} \tag{2.31}$$

For every network, we have shown previously that boundary and coboundary operators can be represented by Tonti Diagrams (see Figure 2.7 for electric networks). This is easily extended in the framework of algebraic topology, making use of $p-$chains, p-cochains, boundary ∂ and coboundary $\delta = \partial^*$ operators defined on primal mesh ($\bar{}$) and dual mesh ($\tilde{}$). This is illustrated on Figure 2.16. The first attempt of this graphical representation was given by F.H. Branin in its seminal paper [Branin, 1966]. F.H. Branin based its work on previous ideas proposed by J. P. Roth [Roth, 1955a, Roth, 1955b] and initiated by G. Kron [Kron, 1939].

In this sub-section, we consider only discrete – or global – physical quantities defined on discrete meshes. In the next subsection, we will see that this can be extended to continuous physical quantities (fields) using the concept of *linear differential forms*. In such a case, the discrete coboundary operator δ becomes the *exterior differential operator* d proposed by E. Cartan in its exterior differential calculus.

2.2.3 Links between global variables and discrete or continuous differential forms

The purpose of this subsection is to show that cell-method and the use of global variables *à la Tonti* is an intuitive use of the mathematical tools of differential

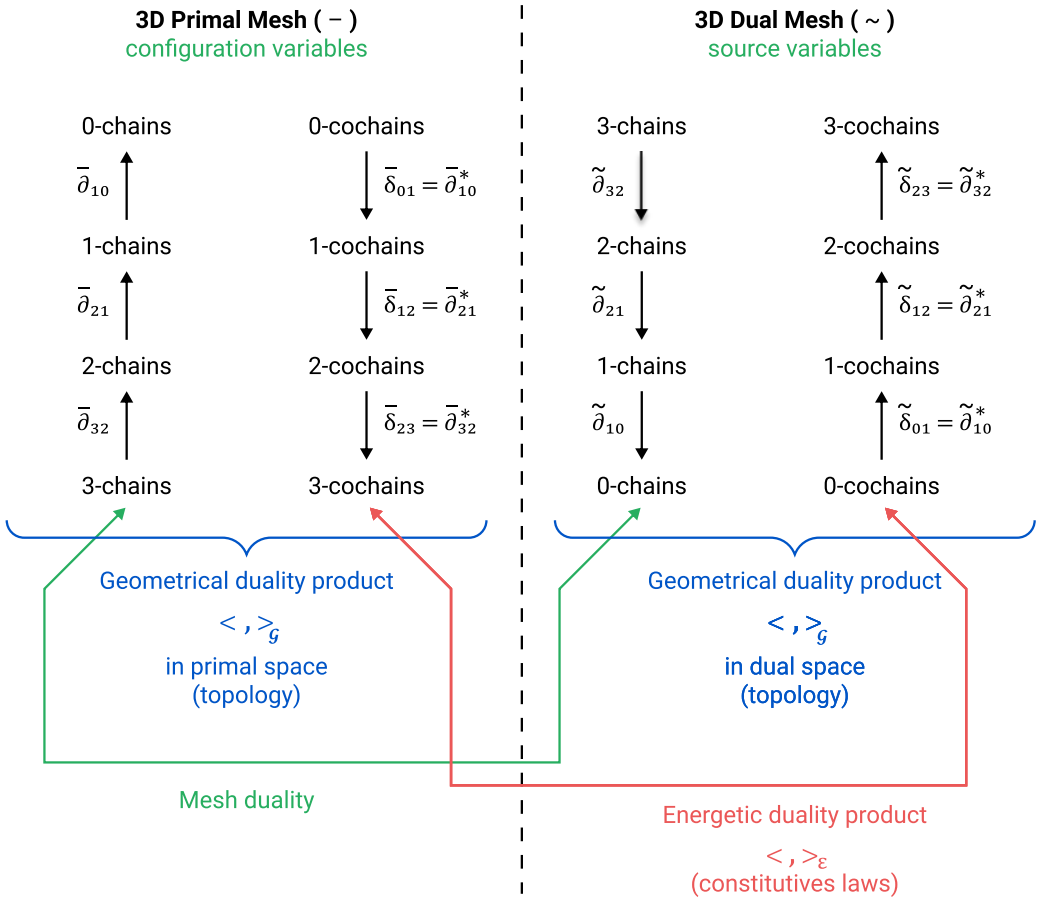


Figure 2.16: Tonti diagram for physical problem on discrete mesh (3-complex).

geometry. In this sense, this approach is much more modern and more in line with the current vision of physicists than the classical approach based on differential calculus.

Usually in engineering, the mathematical model of a physical continuous problem is first formulated as a system of partial differential equations (PDEs), with each equation having a natural connection to physical aspects of underlying problem. In the field of FEM approximation [Hughes, 2012], these PDEs are known as the *strong form*. They are based on *vector field* quantities and *differential operators*, such as **grad**, **curl** and **div**. As mentioned above, the idea of FEM is to convert a continuous operator problem (such as a PDE) to a discrete problem. In principle, the FEM eliminates all the spatial derivatives from the PDE by projecting the strong forms on test functions and integrating the result in the space. All functions are also expressed on a finite basis[5] to obtain the weak form. This computation gives a set of algebraic equations for steady state problem or a set of ordinary differential equations for transient problem. The idea of starting from PDEs and then convert continuous operators to discrete ones is used in many numerical schemes, nevertheless, we should stress that, historically, every physical theory was first discovered and formulated using global quantities and

[5]This basis is the same as the test functions for Galerkin method used in FEM.

only latter using PDEs. For example, concerning electromagnetism, it was first expressed using charge, current, voltage, electric and magnetic flux and not using fields. Only after the publication of Maxwell's treatise, electromagnetic laws were commonly written using PDEs [Tonti, 2001b].

Since the middle of the XIX century, almost all physics theory such as mechanics, fluid dynamics, thermodynamic and all other *classical* physics laws have been described in terms of PDEs and *vector analysis* tools. As a result, vector analysis became the standard language in which the geometric laws of physics were defined. But a new mathematical langage, *differential exterior calculus*, appears in the middle of the twenty century. To stress its power and advantages, let's quote some part of the preface of [Bamberg and Sternberg, 1991]:

> ... while vector analysis is well suited to the geometry of tree-dimensional Euclidean space, it has a number of serious drawbacks. First, and least serious, is that the essential unity of the subject is obscured. Thus fundamental theorem of the calculus, Green's theorem, Gauss' theorem and Stockes' theorem are all aspects of the same theorem (now called Stockes' theorem). But this is not at all clear in the vector analysis treatment. More serious is that the fundamental operators involve the Euclidean structure (for example, **grad** and **div**) or the three-dimension structure and orientation as well (for example **curl**). Thus the theory is wedded to a three-dimensional orientated Euclidean space. A related problem is that the operators do not behave nicely under general changes of coordinates – their expression in non-rectangular coordinates being unwieldy. Already, Poincaré, in his fundamental scientific and philosophical writing which led the theory of relativity, stressed the need to distinguish between those *laws of geometry* and *laws of physics* which are "topological", *i.e.* depend only on the differential structure of space and so are **invariant** under smooth deformation, and those which depend on more geometrical structure such as a notion of distance. One of the major impacts of the theory of relativity on mathematics was to encourage the study of higher-dimension spaces, a study which had existed in the previous mathematical literature, but was not regarded as central to the study of geometry. Another was to emphasize general coordinate changes. The vector analysis was not up to these two task and so was supplemented in the more advanced literature by *tensor analysis*. But tensor analysis with its jumble of indices has a number of serious drawbacks, the most serious of which being that it is extraordinary difficult to tell which operations have any geometric significance and which are artifacts of the coordinate system. Thus, while it is reasonably well-suited for computation, it is hard to assess exactly it is that one is computing. The whole purpose of the development initiated by Hamilton – to have a calculus whose objects have a perceived geometrical significance – was vitiated. In order to make the theory work one had to introduced relatively sophisticated geometrical construct, such as an affine connection. Even with such constructs the geometrical meaning of the operations are obscure. In fact tensor analysis never displaced

> the intuitively clear vector analysis from elementary curriculum.
>
> It is generally accepted in the mathematics, and gradually being accepted in the physical, that the most suitable framework for geometrical analysis is the **exterior differential calculus** of Grassmann and Cartan. This calculus has the advantage that its computational rules are simple and concise, that its objects have a transparent geometrical significance, that it works in all dimensions, that behaves well under map and changes of coordinates, that it has an essential unity to its principal theorems and that it clearly distinguishes between the "topological" and "metric" properties. The geometric laws of physics take on a simple and elegant form in terms of the exterior calculus.

This long quotation from [Bamberg and Sternberg, 1991] highlights several points:

1. *Classical vector calculus* is unable to distinguish between primal and dual quantities as reported on Figure 2.9. This can be done by *tensor quantities* using a distinction between covariant and contravariant quantities but this is uncommon in engineering and the use of tensor loses much of the geometric intuition.

2. Vector fields are most of the time *proxies* from *differential forms* [Bossavit, 2012]. In such a case, using forms instead of vector field is more correct from a physical and mathematical point of view.

3. Sources quantities are *energetic* differential forms of configuration quantities. Then, they can be considered as energy by unit of configuration quantities. Therefore, once again, exterior calculus is the right way to fit our modeling into energetic framework.

As a consequence of all these remarks, differential forms [6] are even more used to describe engineering problems as can be seen in several recent papers [Perot

[6] This footnote recalls some basics about differential forms. It is intended for engineers and is strongly inspired by page 1 of [Flanders, 1989]. Exterior differential forms are the things which occur under integral signs. For example, a line integral

$$\int Adx + Bdy + Cdz$$

leads us to the one-form

$$\omega = Adx + Bdy + Cdz$$

a surface integral

$$\int\int Pdydz + Qdzdx + Rdxdy$$

leads us to the two-form

$$\omega = Pdydz + Qdzdx + Rdxdy$$

and a volume integral

$$\int\int\int Hdxdydz$$

leads us to the three-form

$$\omega = Hdxdydz$$

and Zusi, 2014, Warnick and Russer, 2014]. As said before, global variables proposed and used by E. Tonti are no other than *discrete* differential forms. From the modern geometric point of view, they are the most convenient and natural mathematical quantity to be used for numerical computation on meshes.

In the following of this thesis, a p-form will be usually noted ω^p to remind its dimension. Therefore and intuitively, p-form are just the *infinitesimal* counterpart of *p-cochains* and the exterior derivative d is just the *infinitesimal* counterpart of the *coboundary operator* δ seen previously. In such a way, by adopting the exterior differential calculus framework, it can be seen that there is a *geometric smooth continuity* between *distributed parameters models* (corresponding to the 3rd level models) and *lumped parameters models* (corresponding to the 2nd level models). In such a way, there is only one definition for differential operator d applied to *differential forms* (nevertheless, this one is applied differently in accordance with the dimension of forms). This is also true for the discrete version of the previous subsection: there is only one definition for *boundary operator* ∂ and only one definition for *co-boundary operator* δ (its dual) in algebraic topology (but these one are applied differently in accordance with the dimension of the chains or cochains). In this framework, the Poincaré Lemma is just a direct consequence of the duality between *boundary* and *co-boundary* operators:

> The boundary of a boundary is always null ($\partial\partial = 0$) $\overset{\text{by duality}}{\Rightarrow}$ the co-boundary of a co-boundary is always null ($\delta\delta = 0$ or $dd = 0$).

The difference between the primal and dual space appears also in differential forms with the difference between *straight/inner* forms (configuration variables) and *twisted/odd/outer* forms (sources variables) [Burke, 1985]. *Straight forms* are forms defined on a space with an *inner orientation*. They are used to described *configuration distributed quantities*. *Twisted forms* are forms defined on a space with an induced *outer orientation*. They are used to described *source distributed quantities* [Ferretti, 2014]. This mathematical framework facilitates model reduction operations, to transform field – continuum – models (EDPs, strong form) into discrete – lumped – models (EDOs, network models).

Discrete Tonti diagram of Figure 2.16 is easily extended for field models in the framework of differential exterior calculus. Figure 2.17 from [Tonti, 2013] reports the parallelism between exterior differential calculus (discrete and continuous version) and vector calculus for homology/co-homology structure, i.e. de Rham

---

Integrals used for differential forms are oriented integrals. We shall associate with each p-form ω an $(p+1)$-form $d\omega$ called the *exterior* derivative of ω. Its definition will be given in such a way that validates the general Stockes'formula

$$\int_{\partial\Omega} \omega = \int_{\Omega} d\omega$$

Here Ω is an $(p+1)$-dimensional oriented variety and $\partial\Omega$ is its boundary. A basic relation for exterior differential calculus is the Poincaré Lemma:

$$d(d\omega) = 0$$

In all case this reduces to the equality of mixed second partials for exact forms.

Theory. This is the basics of generalized Tonti diagram for configuration (inner oriented space) or sources variables (outer oriented space).

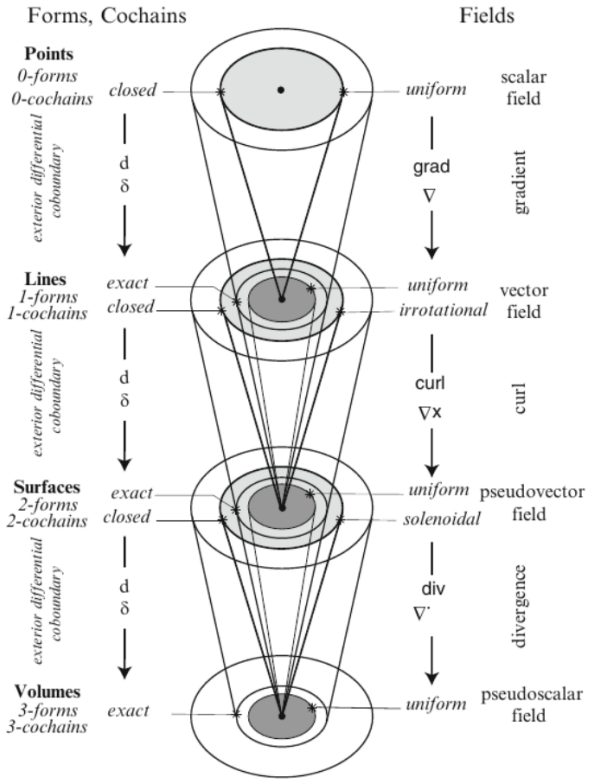


Figure 2.17: Tonti diagram for physical problem on 3D continuum space and de Rham theory [Tonti, 2013].

In terms of computational schemes, *exterior differential calculus* also provide some fundamental advantages. As said above, when applied to *field theories*, numerical methods require the solution of a system of algebraic equations. It is standard practice, to derive these equations starting from PDEs resorting one of many discretization methods (like FEM or finite difference method). The geometric vision proposed here makes possible to express the laws of physics directly by a set of algebraic equations (discrete forms), instead of obtaining them from a discretization process applied to differential PDE equations. This is the point of view adopted by E. Tonti to propose its cell method and also by several similar *mimetic* numerical methods.

2.3 Outline of the next chapters

The purpose of this chapter was to sketch the tools and theory we want to use in the next chapters. Based on the state of the art and a carefull review of the literature, a rationale was proposed between second and third levels models if we adopt the point of view of algebraic topology and differential exterior calculus instead of the more classic *vector analysis* framework.

Now, it is time to propose an efficient simulation tool to compute multiphysic

fields and to propose an efficient method for model order reduction in this framework. The multiphysic field computation is the main purpose of the chapter 3. This chapter will present many developments concerning the generation of dual meshes as well the building of multiphysic numeric solvers based on ideas from cell-methods [Tonti, 2001a] and Discrete Exterior Calculus (DEC) methods [Hirani, 2003]. Later on, we will call this computation method, a cochains' method. The model reduction is the main purpose of the chapter 4. This chapter will present a field interpolation based on Whitney elements. Then, clustering and classification algorithms will be presented to propose a model reduction method to generate automaticaly network models from fields results computed by the solver proposed in chapter 3. The resulting network models can serve as the basic model for *Spice-like* solvers, optimization or design tools.

Chapter 3

Simulation Tools for field computation

The purpose of this chapter is to present the first part of the simulation software developed during this doctoral thesis. This part concerns the *simulation of fields for multiphysical systems*. This chapter, builds on the theoretical tools presented in the previous chapter, focus on several sequential steps as displayed in Figure 3.1. This schema follows the classic schema of FEM method with some extensions. As in the classic approach, first we should produce a geometric model in *Computer Aided Design* (CAD). Since we have only used free software as the primary source, we can only use open geometry formats such as `.geo`, `.iges` and `.step`. With a bit of luck, almost all CAD software (AutoCAD, CATIA, etc.) can export geometry in this format. This first step is the only one that remains unchanged compared to the classical FEM method. Following steps contain important changes that are described in this chapter.

Figure 3.1: Sequential steps for field computation.

Since this thesis concerns engineering aspects, it is essential to present functional tools to illustrate theoretical developments. In order to accelerate software development and in particular to be able to reuse code elements developed by other teams, this work has chosen to rely as much as possible on libraries already available. This allowed us to focus on new aspects without having to redevelop existing parts that are perfectly functional and efficient. In this way, the numerical framework developed during this work provides some routines that help us test our general approaches. The idea of the framework is to link and adapt different *stand-alone* software to test our methodological approach. As we needed, in most cases, to adapt the interface of each software and sometimes to modify algorithms, we decide to only adopt freewares or open source softwares. It must be emphasized that the code developed is only a first prototype and therefore does not provide a specific graphical user interface to guide the user through all stages of the modeling and simulation process. The objective of this chapter is

therefore to explain in a more general way the deployment of software to facilitate the use of our framework for multiphysical fields modeling.

3.1 Historical review of the evolution of topological methods in field computation

In the early stages of the development of the numerical models based on discrete operators the structured cubic and square mesh was adopted. Due to simplicity of this type of meshes, the duality relations is defined in a very simple manner. This helped to test such operators to prove stability and convergence of numerical schemes. Besides, the structured mesh is not always suitable to resolve many engineering problems for complex geometry. It becomes necessary to introduce such a methods to unstructured non-cubic meshes. To this purpose, some improved numerical schemes were proposed.

Specially, these methods are most commonly used in computational electromagnetics. In [Dodziuk, 1976], a finite difference method on simplicial meshes based on Whitney forms and discrete Hodge theory is developed. The paper [Yee, 1966b] is the foundation of an entire class of methods for computation electromagnetics called *finite difference time domain* (FDTD). In this numerical scheme, the author proposes to solve a time-dependent Maxwell's equations on a rectangular mesh using edge unknowns as a degrees of freedom for the electric field and face unknowns for magnetic field. The other work [Weiland, 1977], introduces another approach to the discretization of the Maxwell's equations which uses the primary mesh for the discretization of Faraday's induction law and a dual mesh for the discretization of the Maxwell-Ampère's law. An interpolation of the electric field $\mathbf{E}$ and the magnetic field $\mathbf{H}$ between the meshes is needed do discretize the constitutive relations $\mathbf{D} = \epsilon\mathbf{E}$ and $\mathbf{B} = \mu\mathbf{H}$. This interpolation helps to avoid problems with orthogonality of mesh. In fact, this allows to use the *staggered grids* which may not always conform to the object of study.

The alternative approaches are based on compatible discretization methods, which means that at the discrete level they reproduce, rather than merely approximate, certain essential structures of continuous problem. For example,the *mimetic methods* aimed to construct high order approximation schemes, that is to propose matrices that can mimic all the desired analytic properties of operators from discrete vector and tensor calculus [Lipnikov et al., 2014]. In additional, *mimetic methods* begin by discretizing the underlying continuum theory of the problem of interest, instead of discretizing the proposed equation or system of equations directly [Sanchez et al., 2014].

This idea of incorporating properties of the continuum calculus in the design of numerical schemes appears in various methods. One of the first articles was the articles [Tonti, 1976c] and [Tonti, 1976b] by E. Tonti published in seventies. He observed that most of physical theories have a similar formal structure from the geometrical and topological points of view. The equations can be reformulated thus in a finite framework using the concept of algebraic topology as fully discrete system of equations (cochains) defined on grid objects (chains) rather than in the continuum.

Indeed, the idea of using exterior calculus and differential forms recasted in

the classical approach of Finite Element methods techniques. The pioneering paper of Raviat and Thomas [Raviart and Thomas, 1977] proposed the first stable finite element formulation for approximate differential equations with the help of differential forms for solving the scalar Laplacian in 2D. This type of formulation was called *the mixed formulation*. Even that this theory was invented by geometers and physicists for describing physical phenomenas, the community of finite element and computational engineers reinvented various special cases of Whitney and differential forms and developed new variants of them during 1970-80s [Arnold et al., 2010]. For higher dimensions, the introduction of mixed formulation and Whitney 1- and 2-forms for FEM was made by Nédélec [Nedelec, 1980]. In particular, the mathematical relationship between differential forms used by mathematicians and some formulation of the mixed finite element that has already proposed for electromagnetics was explained in the paper [Bossavit, 1988] by Alain Bossavit and Kotiguga's Ph.D. thesis in electrical engineering [Kotiuga, 1984]. This approach was also adopted, in particular in [Boffi et al., 2013], [Demkowicz et al., 2010], [Monk et al., 2003], [Arnold and Chen, 2017], [Monk et al., 2003] and [Dular et al., 1998] among others.

3.2 Mesh generation

The numerical calculations of the fields is based on discretization of the 3D space, hence the need to generate a mesh structure of the given space. We saw in the previous chapter that for our approach the primal and the dual mesh are needed. The different approaches of how we can get such type of discretization of space will be explained in this section.

We focused on generating unstructured tetrahedral meshes. This type of mesh is widely used in simulation and provides a better approximation of the real geometry. As described in the previous chapter, we need to obtain two meshes: one primal and its dual. In addition, these two meshes must be orthogonal to facilitate the implementation of several operators, particularly for the calculation of Hodge star operators (constitutive laws). As will be shown, these Hodge star operators are trivial for dual meshes and can be expressed as diagonal matrices. Scalar products (energy calculations) are also simplified because it is no longer necessary to make projections to calculate them.

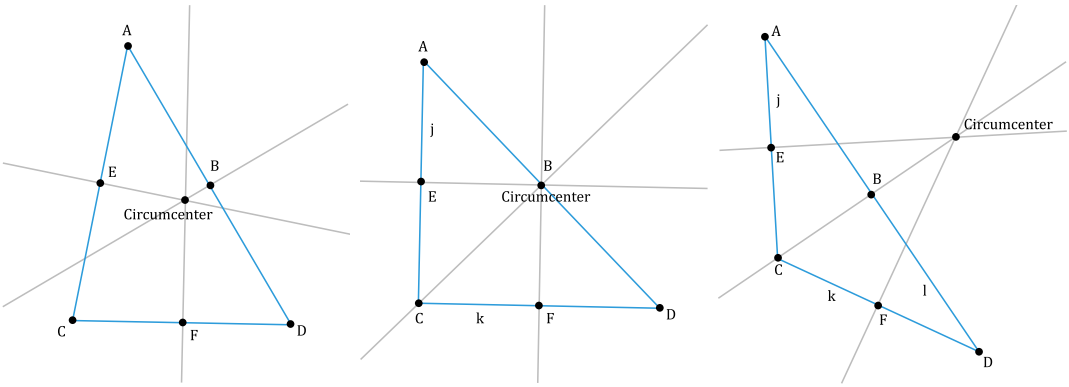
Among the various mesh generation techniques, we will focus on *Delaunay triangulation* technique. This triangulation has numerous application: surface approximation problems (with interpolation), computer graphics issues, finite element computations, communication issues (to minimize the total length of the triangles edges), *etc.* As an historical example, British physician John Snow used a sort of Delaunay triangulation in 1854 to illustrate how the majority of people who died in the Broad Street cholera outbreak lived closer to the infected Broad Street pump than to any other water pump. More applications of Delaunay triangulation can be found in P.L. George's book [George and Borouchaki, 1997].

For understanding the properties of Delaunay triangulation, we need definitions from computational geometry:

> **Definition:** *The Delaunay triangulation for a given set* **P** *of discrete points in a plane is a triangulation* **DT(P)** *such that no point in* **P**

is inside the circumcircle of any triangle in **DT(P)**.

Definition: *If a circumcenter of triangle lies in its interior, we call such a triangle self-centered. In other terms a self-centered triangle is always an acute or a right triangle (see figure 3.2): in (a) and (b) the circumcenter is located inside the triangle, in (c) outside.*



(a) Acute triangle. (b) Right triangle. (c) Obtuse tirangle.

Figure 3.2: Different possible location of the circumcenter.

In the article of V.T. Rajan [Rajan, 1994] and Lemaire's thesis [Lemaire, 1997] we can find the following properties of Delaunay triangulations:

1. The Delaunay triangulation of a set of points in $\mathbb{R}^n$ is a triangulation such that the circumscribed circle of each triangle of this triangulation contains no points of the set in its interior [Rajan, 1994]. In other words, a circle circumscribing any Delaunay triangle does not contain any other input points in its interior. The Figure 3.3 gives a intuitive illustration of this property.

2. The triangulation is unique for a set of points;

3. The union of all simplices in the triangulation is the convex hull of the points;

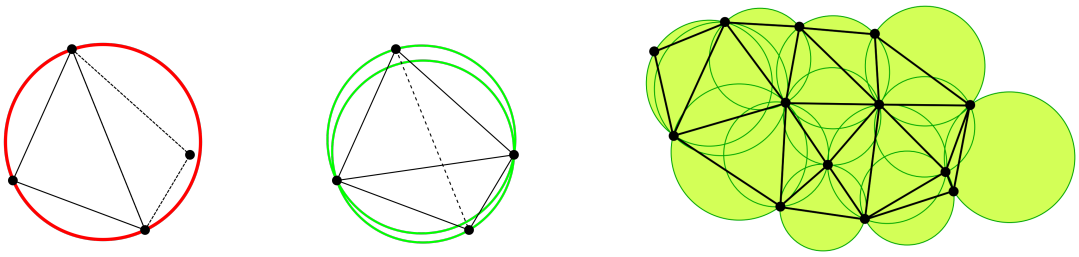
4. In the plane, the Delaunay triangulation maximizes the minimum angle. In fact, it try to get a triangulation, where most of simplices have acute angles. Compared to any other triangulation of the points, the smallest angle in the Delaunay triangulation is at least as large as the smallest angle in any other. However, the Delaunay triangulation does not necessarily minimize the maximum angle [Lemaire, 1997].

5. The calculation complexity of classical method to compute a Delaunay triangulation is $O(n \log n)$ (for *Divide and conquer* algorithm);

6. Great for interpolation: for a bounded-curvature function f with piecewise linear interpolant g, Delaunay minimizes the worst-case bound on the interpolation error $||f - g||$ [Jonathan Richard Shewchuk, 2018];

7. Dual tessellation for Delaunay triangulation exists and is orthogonal and unique (*e.g.* Voronoi tessellation or Voronoi diagram);

8. Convex Delaunay triangulation does not always exists. [Rajan, 1994].



(a) (b) (c)

Figure 3.3: Graphic representation of Delaunay condition: (a) this pair of triangles does not meet the Delaunay condition (the circumcircle contains more than three points)[Community of Wikipedia, 2018]; (b) flipping the common edge produces a valid Delaunay triangulation for the four points [Community of Wikipedia, 2018]; (c) the circumscribing circle of every Delaunay triangle is empty [Jonathan Richard Shewchuk, 2018].

In our case, the most important feature of Delaunay triangulation is a unique and orthogonal dual tessellation, called *Voronoi diagram*. His duality presents two aspects: duality in the sens of projective geometry and also as a dual graph (topology).

> **Definition. Voronoi diagram:** *Suppose that S is a set of n points $S = \{s_1, s_2, \cdots s_n\}$ (called* sites*) in the Euclidean space $\mathbb{E}^k$. Each site s_i is associated with region $V(s_i)$ consisting of all points closer to that s_i than to any other site $s_n \in S$: if $V(s_i) = \{M : dist(M, s_i) \leq dist(M, s_j), \forall i \neq j\}$, so V is called a* Voronoi region *of site s_i. The totality of such regions forms the* Voronoi diagram. *An example of Voronoi diagram is illustrated in Figure 3.4.*

The *Voronoi diagram* has the following properties:

1. The circle containing Voronoi vertex v and passing through the three points s_1, s_2 and s_3 is empty (see Figure 3.5(a)). For the proof, please reference to pp. 204–211 of [Preparata and Shamos, 1985] and [Prof. Roberto Tamassia, 1993].

2. The graph constructed by connecting all vertices in a set S across the edges of their Voronoi polygons is a triangulation of S [Prof. Roberto Tamassia, 1993] and also S it is a Delaunay triangulation (see Figure 3.5(b));

3. The duality is orthogonal in geometric sense: as Delaunay and Voronoi tessellations are based on circumcenter calculation the orthogonality properties are based on the definition of the circumcenter (see Figure 3.5(c));

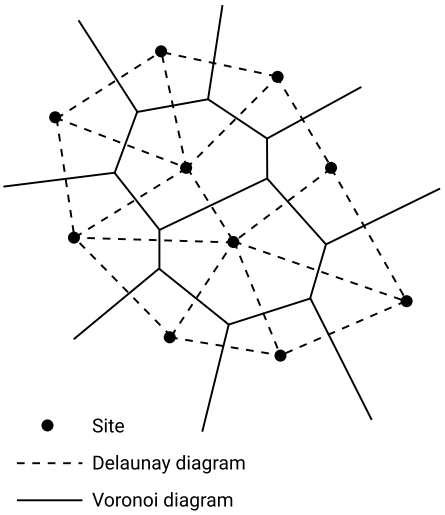


Figure 3.4: Voronoi diagram of plan sites [Lemaire, 1997].

4. The regions of the Voronoi diagram may be either bounded or unbounded. Every point contained in an unbounded region of the diagram lies on the convex hull of the set S. This is particularly clear in an example where all points but one lie on the convex hull [Prof. Roberto Tamassia, 1993] (see Figure 3.5(d)).

In addition, when using discrete differential geometry operators, Meyer et al. [Meyer et al., 2003] recommend using a Voronoi dual (circumcentric). These authors show that the use of a Voronoi dual allows error minimization when calculating simple geometric attributes such as normal vectors and discrete surface curvatures. In addition, another advantage of the Delaunay/Voronoi duality is that convexity and its non-intersection makes it easier to use barycentric coordinates [Warren et al., 2007]. This property is very useful for fluid simulation, as noticed in [Elcott et al., 2007].

Even, when Delaunay triangulations maximize the minimum angle of all triangle angles in the triangulation; they thus tend to avoid *acute* or *obtuse* triangles [Rajan, 1994]. Such triangulation exists for each set of points and is the simplicial decomposition of the convex hull where the vertices of the triangles are contained in the set of points, but we need to make sure all edges and surfaces appear (see Figure 3.6). However, to avoid this type of problem when a dual vertex is not included inside the primal simplex, it is often necessary to return to a dual barycentric instead of a circumcentric [Mullen et al., 2011]. It's a good choice for some methods, like *Mixed Finite Element Method*, that don't need the orthogonality of two simplex: primal and dual. Nonetheless, in our work, we need to maintain orthogonality because in this case the model reduction will be much more easier, if feasible.

To conclude, it appears rational to use the Delaunay/Voronoi triangulation as a primal/dual mesh for topological methods, but there are two major problems with the basic Delaunay/Voronoi algorithms. The first is about "degenerated" triangles/tets. The second is about unbounded Voronoi cells (points at infinity in the border, see Figure 3.5(d)).

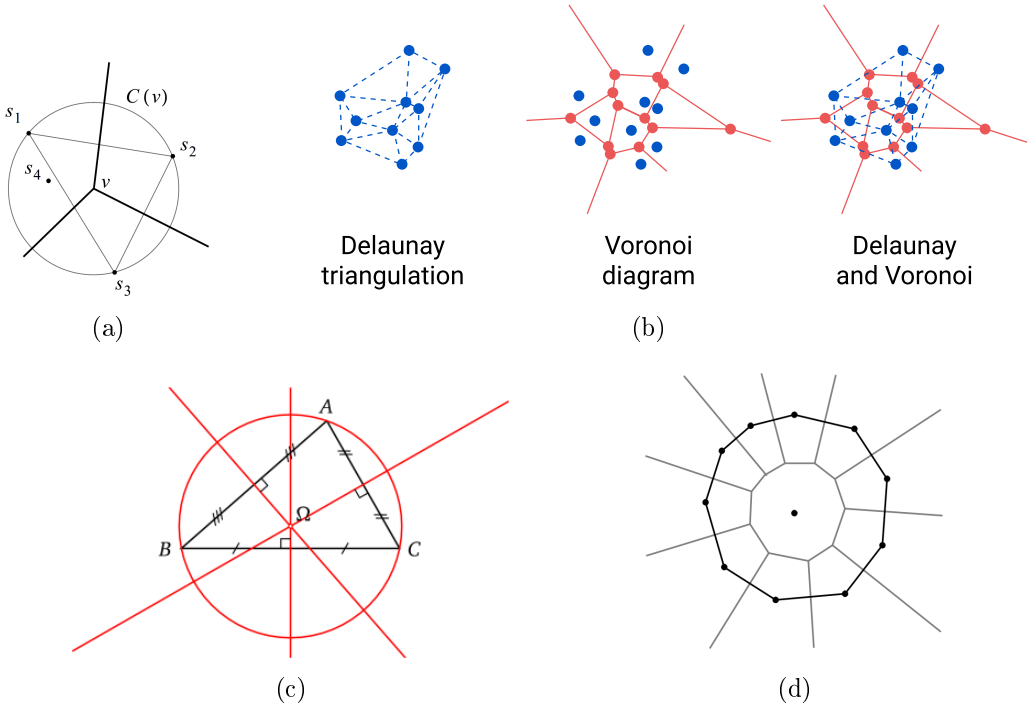


Figure 3.5: Graphic representation of Voronoi properties: (a) the circle $C(v)$ contains no other point of S [Prof. Roberto Tamassia, 1993]; (b) the graph constructed by connecting all vertices in a set S across the edges of their Voronoi polygons is a triangulation of S [Weisstein, Eric W., 1999]. And in the opposite, the dual graph for a Voronoi diagram (in the case of a Euclidean space with point sites) corresponds to the Delaunay triangulation for the same set of points so that the Delaunay triangulations are dual to the Voronoi diagram [Delaunay, 1934]; (c) orthogonality properties of circumcenter Ω of triangle $\triangle ABC$; (d) unbounded regions contain the points on the convex hull of the set S [Prof. Roberto Tamassia, 1993].

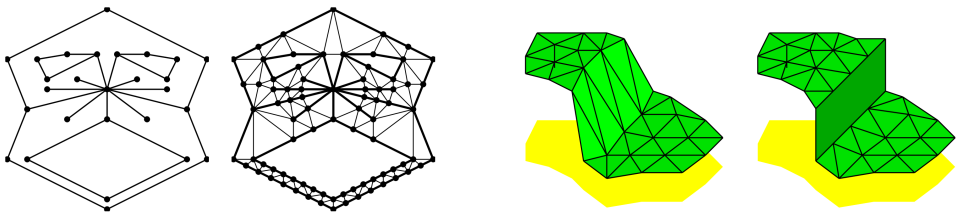


Figure 3.6: Typical problems of Delaunay triangulation for a given set of points: (a) non convex shapes; internal boundaries; (b) discontinuities in interpolated functions [Jonathan Richard Shewchuk, 2018].

3.3 Re-meshing techniques

In this section we will introduce some re-meshing techniques that can ensure that dual and primal meshes are orthogonal.

Basic algorithms use circumcenter of a primal cell to determine a dual vertex

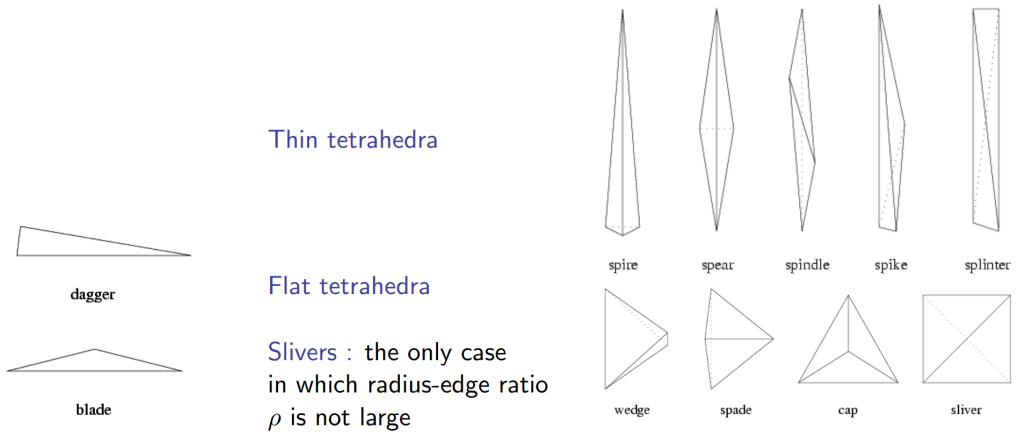
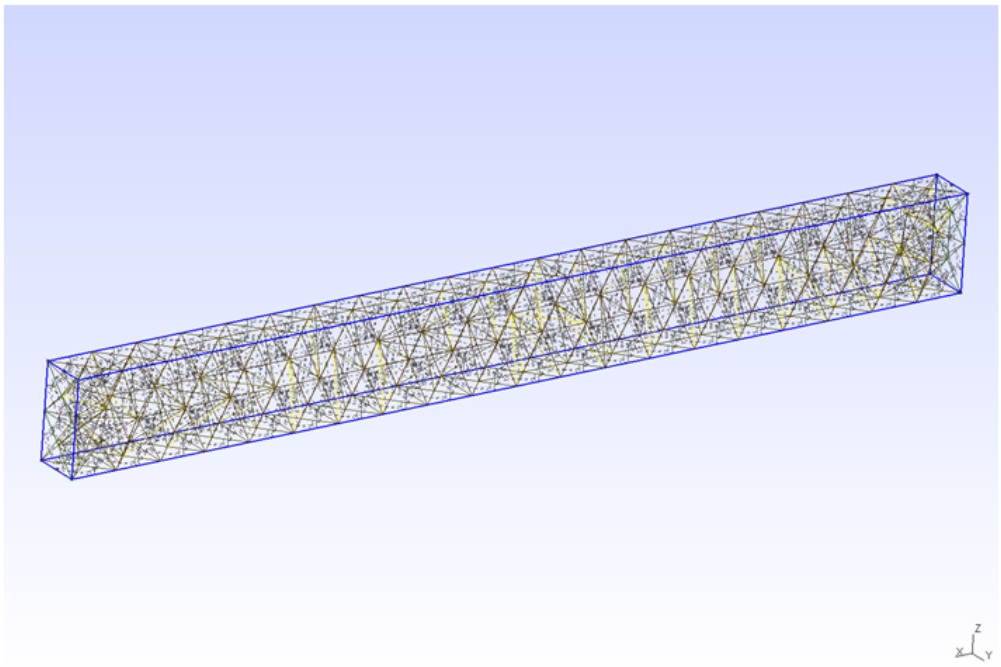


Figure 3.7: Typical problems of Delaunay triangulation for a given set of points: (a) in both cases the radius-edge ratio is large; (b) slivers: the only case in which radius-edge ratio ρ is not large [Jonathan Richard Shewchuk, 2018].

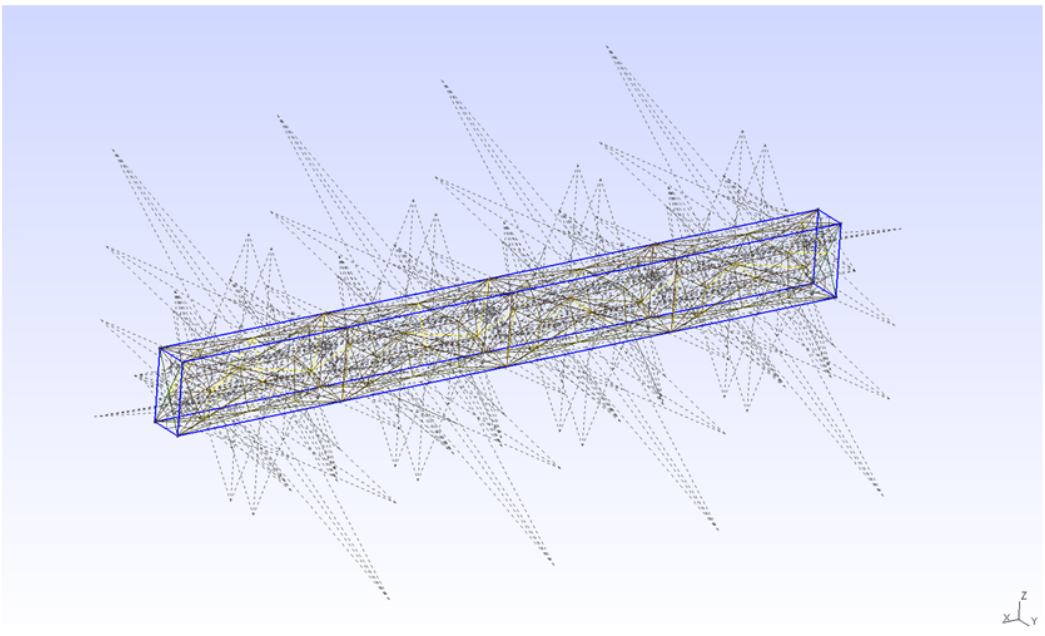
as in Figure 3.2 (a). In practice, it is difficult to obtain a "self-centered" Delaunay triangulation for which each circumcenter is within its associated triangle/tetrahedron [Rajan, 1994] even though the Delaunay triangulation maximizes the minimum angle. This type of mesh is called "degenerate". This type of problem appears when the triangle/tets are *sliver*. In this case, a circumcenter is located outside the triangle as in Figure 3.2 (b).

For the degenerated triangles/tets, it exists a criteria to identify "bad" point set for Voronoi/Delaunay meshing. In 2D, we can say that only acute or right triangles are causing problems, and this is equivalent to having a small angle between two edges. That's why in 2D we can use radius-edge ratio $\rho = \frac{\text{circumradius}}{\text{shortest edge lentgh}}$ as mesh quality factor (in Figure 3.7(a), in both cases the radius-edge ratio is large, but this doesn't hold for tetrahedrons in 3D - see Figure 3.7(b)). An example of a correct (a) and degenerate (b) cases for the mesh of a 3D simple beam is shown in Figure 3.8. This beam corresponds to the geometry of the MEMS device, studied at the end of this chapter. Whether in the case of a triangle (2D) or a tetrahedron (3D), the source of dual mesh problems always comes from the difficulty of finding a mesh such that the calculation of circumcenters ensures that they are inside the triangle (2D) or the tetrahedron (3D).

Another difficulty with dual meshes is the consideration of cells on the boundary of the domain (see Figure 3.9). To simplify, in the following, we will only consider the 2D case, however all the proposed solutions are adapted to both 2D and 3D cases. When calculating the Voronoi diagram of a set of points in a compact domain, all Voronoi cells on the boundary of the domain are located at an infinite distance from the boundary. However, in practice, only a part of the cells located within the domain are required, as is the case when calculating the centroidal Voronoi Tessellation. Such a Voronoi diagram limited to a compact diagram is called a cut-out Voronoi diagram. To calculate this, we can choose an algorithm based on the Restricted Voronoi Diagram (RVD) for several reasons. This type of diagram provides for a set of sites with respect to a compact 3D volume, assuming that the volume is represented by a tetrahedral meshes (for



(a)



(b)

Figure 3.8: Delaunay-Voronoi tetrahedral mesh: (a) correct dual mesh; (b) non correct dual mesh: points consigned to triangles lying outside the tetrahedra.

more information see [Lévy and Schwindt, 2018]). In the following paragraphs we will see some techniques that helps to get orthogonal meshes.

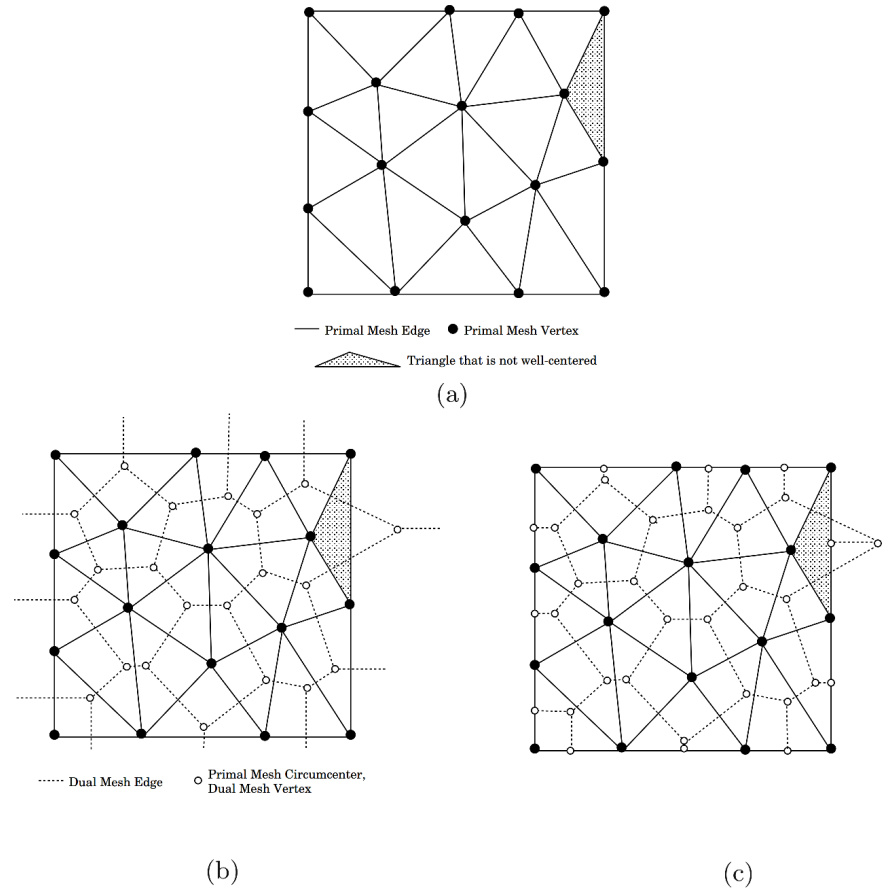


Figure 3.9: (a) Delaunay mesh (b) Voronoi diagram with open (infinite) regions at the boundary (c) Voronoi mesh created by truncating Voronoi diagram by boundary of primal mesh [Garimella et al., 2014].

Optimal transport criteria The basic idea of mesh optimization criteria is based on optimal transport. The optimal transport problem goes back to the French mathematician Gaspard Monge, inventor of descriptive geometry and father of differential geometry. For a description of the extensive literature on this subject, we refer the reader to [Villani, 2008]. The corresponding key question is how to determine the best way to move a pile of dirt M into a hole N of the same volume. Initially, the objective function to be minimized is the integral distance over which dirt is transported. While the variational formulation of Monge's problem assumed that all dirt should be moved using point-to-point mapping (road map) to a location, this restriction has been relaxed by Kantorovich who has replaced mapping with a linear problem on a convex set so that the limits of the maps may not be maps, but the limits of the measures remain measures. This extension has marked a renewed interest in optimal transport problems, as they are general enough to be applied to many scientific applications [Su et al., 2014, Ceron and Splendore, 2018]. The study of this scientific framework naturally leads to an efficient computation algorithm, which uses classical notions of computational geometry, such as the generalization of Voronoi diagrams.

Isotropic adaptation An interesting algorithm for mesh improvement is the isotropic adaptation of the mesh. In general, an isotropic mesh is supposed to avoid *sliver* triangles in order not to cause problems with the calculation of Delaunay/Voronoi diagrams. The *intuitive* definition of mesh isotropy is shown in Figure 3.10. The purpose of isotropic re-meshing is to obtain equal edge lengths and equilateral triangles. For most of the geometries we have used, this approach has been adopted. Almost all modern open source meshing software such as TetGen [Si, 2013], MeshLab [Cignoni et al., 2008] and GMSH [Geuzaine and Remacle, 2009] can generate a tetrahedral mesh based on Delaunay triangulation and an isotropic tetrahedral adaptive re-meshing. The idea of isotropic mesh generation is well described in the document [Karkulik et al., 2013].

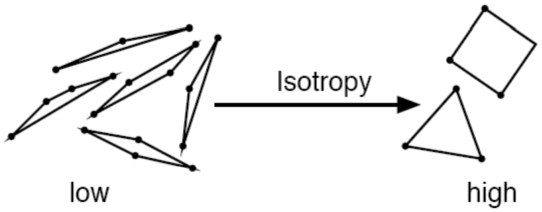


Figure 3.10: An *intuitive* definition of isotropy [Ben-Chen, 2012].

There are several isotropic re-meshing techniques:

1. Parameterization-Based Re-meshing;
2. Direct Surface Re-meshing.

Re-meshing techniques based on parameterization are essential to parametrize the original mesh in order to obtain a bijective mapping and minimize distortion due to the flattening process. There are different objective functions to optimize, for example [Du et al., 1999] proposes to optimize an *energy* function. Given a density function defined over a bounded domain Ω, where each site coincides with the centroid (*i.e.*, the center of mass) of each element of the initial mesh (e.g., initial Voronoi tessellation). The centroid c_i of an initial mesh element is calculated as follows:

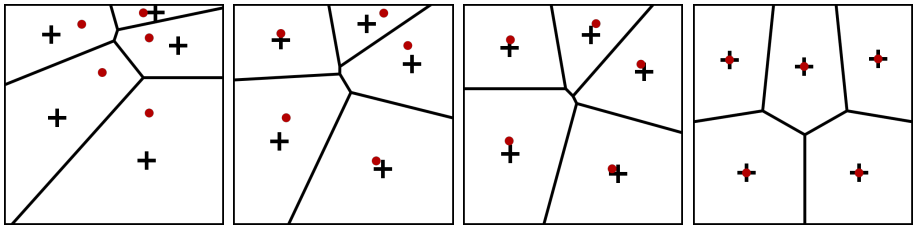
$$c_i = \frac{\int_{V_i} \rho(x) \cdot x \, dx}{\int_{V_i} \rho(x) \, dx} \tag{3.1}$$

where $\rho(x)$ is the density function. This structure is proving to have a surprisingly wide range of applications for numerical analysis, location optimization, optimal resource allocation, cell growth, vector quantification, etc. (for more information, see [Surazhsky et al., 2003]). This follows from the mathematical importance of its relationship with the energy function:

$$E(z, V) = \sum_{i=1}^{n} \int_{V_i} \rho(x) \mid x - z_i \mid^2 dx \tag{3.2}$$

where $V \in \Omega$ and $z \in V$. It is verified in [Surazhsky et al., 2003] that (i) the energy function is minimized at the center of mass of a given region and (ii) for a given set of centers $Z = z_i$, the energy function $E(V, Z)$ is minimal when V is a Voronoi tessellation. Thus, the Lloyd's algorithm can be used. The Lloyd

algorithm, also known as Voronoi iteration or relaxation, is an algorithm named after Stuart P. Lloyd to find sets of uniformly spaced points in subsets of Euclidean spaces and partitions of these subsets into convex cells of uniform shape and size [Lloyd, 1982]. A good example where this algorithm works is shown in Figure 3.11.



(a) 1st iteration (b) 2nd iteration (c) 3rd iteration (d) 15th iteration

Figure 3.11: Example of Lloyd's algorithm [Dominik Moritz, 2018]. The plus signs denote the centroids of the Voronoi cells.

In the last image of Figure 3.11-d, the points are very near the centroids of the Voronoi cells. If we can get a Delaunay triangulation of this set of point we will have a primal and dual mesh that is (1) orthogonal due to Delaunay/Voronoi relation and also (2) we will have no problem with outstanding points as in Voronoi dual because all point is near barycenter.

Using the Lloyd optimization [Surazhsky et al., 2003] we can optimize complex meshes as the one shown in Figure 3.12 but it is not possible to guaranty that it will work for all 3-D objects.

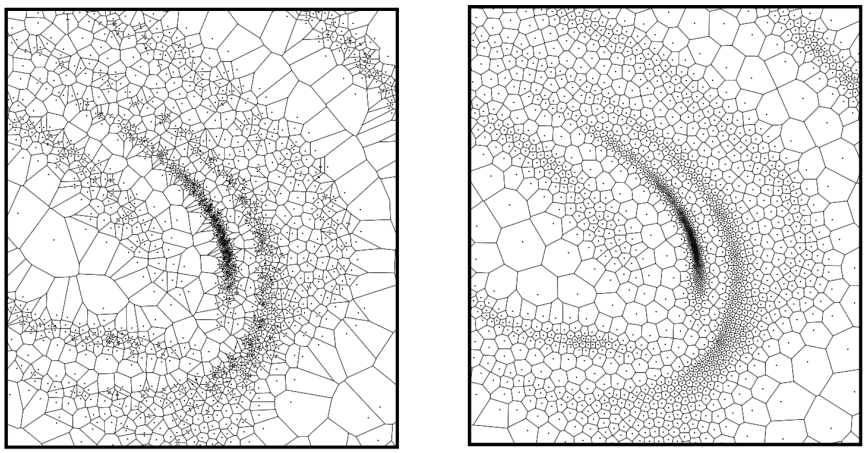


Figure 3.12: (a) Left: Ordinary Voronoi tessellation of a point set sampled from some density function (b) Right: Point set and its corresponding weighted centroidal Voronoi tessellation for the same density function. Each site coincides with the center of mass of its Voronoi cell. The sample set on the right was generated by Lloyd iterations applied to the sample set on the left. [Surazhsky et al., 2003].

In all our test, the methodology of [Yan et al., 2010] were working perfectly. However, the authors can not confirm the perfect orthogonality for all cases. An

increasing number of numerical methods need strict control over *both* primal and dual meshes especially in discrete differential operators in modeling (*e.g.* [Meyer et al., 2003]).

As explained in the previous chapter, we need primal/dual orthogonality to calculate the ratio between primal and dual geometry entities as it defines Hodge star operator. Once we calculate this operator for a geometric instance, we can used it for discrete exterior calculation. In [Mullen et al., 2011] the authors introduce Hodge-optimized triangulation (HOT). This optimization was designed for fast and accurate computations in computer graphics. In comparison with [Yan et al., 2010] it can guaranteed a well-shaped primal-dual pairs of complexes. In addition, as it is based on Hodges-star notation, we can use this information directly in simulation.

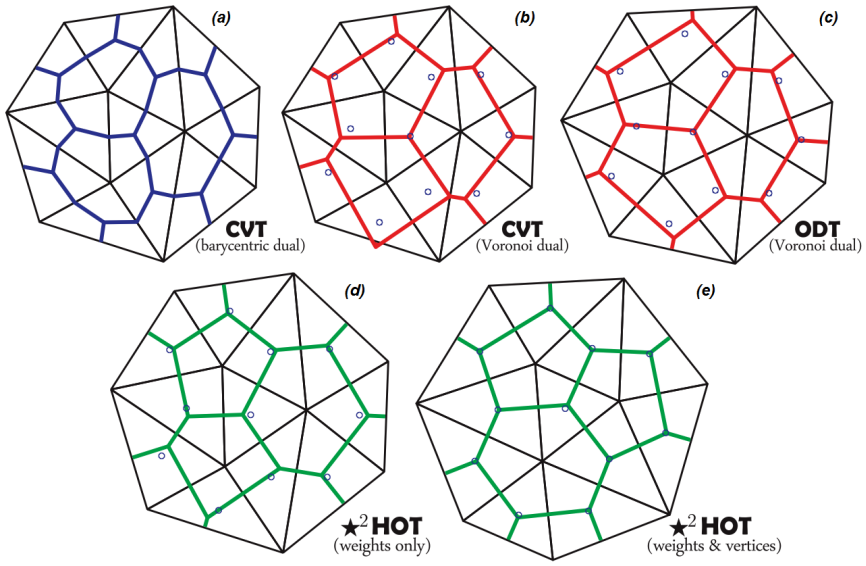


Figure 3.13: Primal/Dual Triangulations: (a) Centroidal Voronoi Tessellations (CVT) – barycentric dual; (b) CVT – circumcentric dual; (c) Optimal Delaunay Triangulation (ODT); (d) Hodge-Optimized Triangulation (HOT) – optimized dual mesh alone; (e) HOT – optimized both the primal and dual meshes.

From the paper [Mullen et al., 2011], we can see in Figure 3.13, that using the barycentric dual (Figure 3.13 - a) does not generally gives orthogonal dual meshes to the primal one (as discussed above). Circumcentric duals, both in Centroidal Voronoi Tesselation (CVT - Figure 3.13 - b) and Optimal Delanay Triangulatios (ODT - Figure 3.13 - c), can lead to dual points far from barycenters of triangles and always guaranteed the orthogonality between primal/dual but ODT doesn't provide a good quality of mesh in the case of silver triangles (circumcenters out of triangles). Changing the freedom provided by weighted circumcenters, the HOT can optimize the dual mesh alone (Figure 3.13 - d) or both the primal and dual meshes, *e.g.* to make them more self-centered (to avoid out space points) while maintaining primal/dual orthogonality (Figure 3.13 - e).

Even if the approach of HOT is the most promising in terms of calculation we were not able to test it because of unavailability of code. Developing this approach by ourselves seems not to be so easy and it was not the main focus of our research.

For this thesis, we have chosen the Gmsh software created by *Christophe Geuzaine* and *Jean-François Remacle* as mesh generator. *Gmsh* is a free 3D finite element mesh generator with a built-in CAD engine and post-processor. Its design goal is to provide a fast, light and user-friendly meshing tool with parametric input and advanced visualization capabilities input and advanced visualization and computing. *Gmsh* is sufficient to realize the Delaunay mesh and it Voronoi dual for most of the geometry we used. However for some cases it did not proved to be sufficiently effective and we had to supplement it with another software.

For such purpose and among the various computational algorithms available in computer graphics, we have chosen the approach proposed by the INRIA Institute in Nancy, France. The *"Geogram"* application developed by the ALICE team at INRIA was created for the processing of data from 3D scanners (Stereolithography). This data processing was intended to improve the quality of mesh models. Among the proposed algorithms, we used an RVD diagram calculation described in [Yan et al., 2010]. In addition, this library offers some methods to optimize mesh quality. These algorithms were originally used to improve the meshes generated from geometric models from 3D scanners (called *ModeMeshRepairMode*). All these algorithms use the idea of redefining the primary mesh so that its dual is as orthogonal as possible to the primal. In this case, the orthogonality criterion of the Delaunay-Voronoi tessellation is used to qualify a mesh size as "good". In the rare case where *GMSH* has not be sufficiently effective, we used *"Geogram"* library in free access to ensure that dual and primal meshes were orthogonal. The basics of the software optimization algorithm of *"Geogram"* is described in [Lévy and Schwindt, 2018].

3.4 PyDEC: library and working

This section describes the algorithms of PyDEC, a Python library for computations related to the discretization of exterior calculus [Bell and Hirani, 2012]. In our work, we used this library for representing a mesh as a simplicial complex and then calculate discrete exterior derivatives and metric-dependent Hodge star operator. When combining with the exterior derivative appropriately, the vector calculus operators *div, grad, curl* can be generated from simplicial complex as adjacency matrices [Bell and Hirani, 2012]. We have chosen this library because it provides all needed algorithms for constructing the operators and objects that arise in discrete exterior calculus. In addition, it provides a well structured data objects with high-level matrix operations for represent a problem. The availability of such libraries makes it suitable for prototyping numerical methods.

3.4.1 Simplicial complex representation

One common type of discrete domain used in scientific computing is triangle or tetrahedral mesh. These and their higher dimensional analogues are implemented as n-dimensional simplicial complexes embedded in $\mathbb{R}$, $N \geq n$. Simplicial complexes are useful even without an embedding and even when they don't represent a manifold, for example in topology and ranking problems [Bell and Hirani, 2012].

The definitions here are given for simplicial complexes and generalized to the other types of complexes implemented in PyDEC. In PyDEC, only integer-valued

chains and real-valued cochains are considered, for finite (that is, ones with a finite number of cells) simplicial complex and denote its underlying space by $|K|$. Give $|K|$ the subspace topology as a subspace of $\mathbb{R}^N$. For a finite complex this is the same as the standard way of defining topology for $|K|$ [Munkers, 1984] and $|K|$ is a closed subspace of $\mathbb{R}^N$ [Bell and Hirani, 2012].

An oriented simplex with vertices $v_0...v_p$ is written as $[v_0...v_p]$ and given names like σ_i^p with the superscript denoting the dimension and subscript denoting its place in some ordering of p-simplices. Sometimes the dimensional superscript and/or the indexing subscript will be dropped. The orientation of a simplex is one of two equivalence classes of vertex orderings. We consider two orderings are equivalent if one is an even permutation of the other. For example, $[v_0, v_1, v_2]$ and $[v_1, v_2, v_0]$ denote the same oriented triangle while $[v_0, v_2, v_1]$ is the opposite one. This orientation is extremely important for calculus because the values stored are representing integrals of the associated k-form over the underlying simplices, we *must* keep track of orientation. For example, reversing the bounds of integration on $\int_a^b f(x)dx$ flips the sign of the resulting value. In our case, suppose we have a 1-form f with value f_{ij} assigned to edge $e = (i, j)$; that is the real number f_{ij} is the integral of 1-form f over the line of segment (p_i, p_j). If we query the value of this form of the edge (j, i) we should get $-f_{ij}$ (see Figure 3.14).

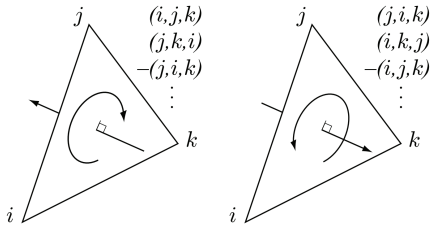


Figure 3.14: All permutations of a triple (i, j,k) refer to the same triangle, and the sign of the permutation determines the orientation [Elcott and Schroder, 2006].

To manage this we need to impose a *intrinsic orientation* for each simplex. It is with respect to this orientation that the values stored in the form vectors receive the appropriate sign. In order to do this we used a convention proposed in [Elcott and Schroder, 2006]. The idea is as follows. The *faces* of k-simplex are the $(k-1)$-simplices are incident on it, *i.e.*, the subset of one lower dimension. Every k-simplex has $k+1$ faces. Each face corresponds to removing one integer from the tuple, and the relative orientation of the face is $(-1)^i$ where i is the index of the integer that was removed [Elcott and Schroder, 2006]. To clarify:

- The faces of a tetrahedral (t_0, t_1, t_2, t_3) are $-(t_0, t_1, t_2)$, $+(t_0, t_1, t_3)$, $-(t_0, t_2, t_3)$ and $+(t_1, t_2, t_3)$;
- The faces of a triangle (f_0, f_1, f_2) are $+(f_0, f_1)$, $-(f_0, f_2)$ and $+(f_1, f_2)$;
- The faces of an edge (e_0, e_1) are $-(e_0)$ and $+(e_1)$.

To show the data object used to represent a simplicial complex let us consider the triangle mesh in 2D with vertices and faces enumerated as shown in Figure 3.15. According PyDEC notation, this example mesh is representing by arrays:

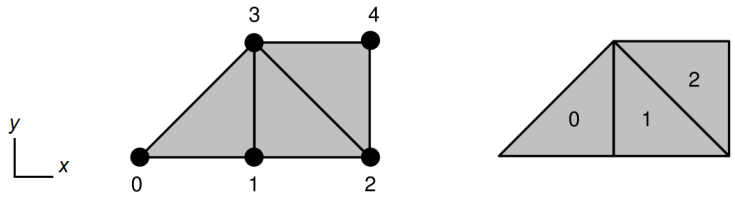


Figure 3.15: Simplicial mesh with enumerated vertices and simplices [Tutorial PyDEC, 2012].

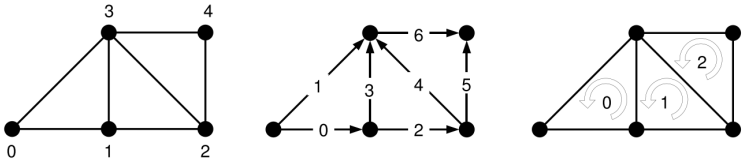


Figure 3.16: Simplicial complex with oriented edges and triangles [Tutorial PyDEC, 2012].

$$V = \begin{bmatrix} 0.0 & 0.0 \\ 1.0 & 0.0 \\ 2.0 & 0.0 \\ 1.0 & 1.0 \\ 2.0 & 1.0 \end{bmatrix}, S_2 = \begin{bmatrix} 0 & 1 & 3 \\ 1 & 2 & 3 \\ 2 & 4 & 3 \end{bmatrix}, S_1 = \begin{bmatrix} 0 & 1 \\ 0 & 3 \\ 1 & 2 \\ 1 & 3 \\ 2 & 3 \\ 2 & 4 \\ 3 & 4 \end{bmatrix} S_0 = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}. \tag{3.3}$$

where the subscript $n = 1, 2, 3$ denote the dimension of the simplices. The i-th row of V contains the spatial (x, y) coordinates of the i-vertex. The i-th row of S_2 contains the indicies of the vertices that form the i-th triangle in *anticlockwise* orientation for each (see Figure 3.16). This type of orientation is called *global* orientation in distinction from *local* orientation of each element mentioned above.

3.4.2 Algorithm for discrete exterior derivative calculation

As explained in the previous chapter, given the manifold M, the exterior derivative $d : \Omega^p(M) \to \Omega^{p+1}$, which acts on differential p-forms $\omega^p \in \Omega^p$, generalize the derivative operator of calculus. This operator is a metric independent operator and in the discrete case where M is a mesh, d becomes a discrete exterior derivative δ which is defined as a coboundary operator of algebraic topology [Munkers, 1984]. In such a case, d is the dual operator of the boundary operator ∂, *i.e.* $d = \delta = \partial^* = \partial^t$, where $\partial_p : C_p(K) \to C_{p-1}(K)$ can be defined through incidence matrices. Therefore ∂ describes the topological relation between two complexes $S_p \in C_p(K)$ and $S_{p-1} \in C_{p-1}(K)$, defined as an oriented simplical complex. The boundary operator on a p-simplex $\sigma^p = [v_0, \cdots, v_p]$ is given in terms of its $(p-1)$-dimensional faces as

$$\partial_p \sigma^p = \sum_{i=0}^{p} (-1)^i [v_0, \cdots, \hat{v_i}, \cdots v_p] \tag{3.4}$$

where $\hat{v_i}$ means that v_i is omitted.

If we take as example the mesh of Figure 3.15, we need to get 3 matrices, containing -1 and 1 only. In this case, ∂_2 is matrix with $[E, F]$ dimension with 3 non-zero elements per column, ∂_1 a $[V, E]$ with 2 non-zero element per column (one +1 and one -1). Where F, E, V are the number of faces(triangles), edges and vertices respectively. The matrix of ∂_0 is zero-row matrix. The transposed of these matrices are the *coboundary* operators $\partial^t = \partial^* = \delta = d$ [Elcott and Schroder, 2006]. The basic algorithm to get matrices is quite easy to understand (see Algorithm 1) but is difficult to implement in optimal way: we must be sure to avoid duplicates, either by using a unique associative container, or by sorting the list afterward and removing duplicates. Then the matrices are constructed as follows:

Algorithm 1 Recursive algorithm for boundary operators calculation

```
1: function BOUNDARYFACES(simplicialComplex)
2:     for each simplex s_i in simplicialComplex do
3:         for each face v of s_i do
4:             tuple = (-1)^k [v_0, ..., v̂_k, ... v_n]
5:             determine the index j of v by locating its
   representative
6:             set tuple of the appropriate matrix ∂_i at row j
7:         end for
8:     end for
9:     return ∂_{0,1,...,n}
10: end function
```

For the example the mesh of Figure 3.15, the boundary operator ∂ are

$$\partial_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{3.5}$$

$$\partial_1 = \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & -1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}, \tag{3.6}$$

$$\partial_2 = \begin{bmatrix} 1 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix}. \tag{3.7}$$

In PyDEC algorithms are also optimized to get directly matrices in Compressed Sparse Row (CSR) to save memory. For more detail please read PyDEC manual [Tutorial PyDEC, 2012].

3.4.3 Algorithm for Hodge star operator calculation

For real physical problems we need to introduce a metric, directly linked to constitutive laws. In PyDEC, the main metric-dependent operator is the Hodge star that enables the discretization of co-differential and Laplace-deRham operators [Bell and Hirani, 2012]. In the literature the hodge star is defined as the operator that transforms a primal k-form into a dual $(n-k)$-form. Here, again, we should underline when transferring a quantity from a primal simplex to a dual cell, we should assign a proper orientation sign because of these are integral values. So, we can't simply assign the value on a dual to be equal to the value on the primal as the domain of integration is unrelated. Instead, we require that the integral density be equal. So, if σ denotes the evaluation of a form on a primal k-simplex ω, then $\star\sigma$ is the value on the dual $(n-k)$-cell $\tilde{\sigma}$ [Elcott and Schroder, 2006] such that

$$\frac{\omega}{\text{Volume}(\sigma)} = \frac{\star\omega}{\text{Volume}(\tilde{\sigma})} \tag{3.8}$$

From (3.8) we can define a diagonal Hodge star as

$$\star = \frac{\text{Volume}(dual)}{\text{Volume}(primal)} \tag{3.9}$$

Actually, a definition of Hodge star in (3.9) is only a ratio between a dual and primal complex. In the case of tetrahedral primal mesh, we can see dual polyhedral in Figure 3.17.

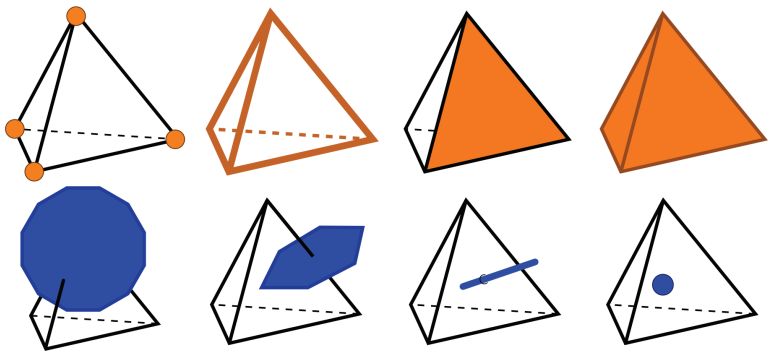


Figure 3.17: There is one dual polyhedron for every primal vertex, one dual polygon for every primal edge, one dual edge for every primal triangle, and one dual vertex for every primal tetrahedron [Elcott and Schroder, 2006].

The basic algorithm to get the volume of the dual cells is presented in the algorithm 2 [Elcott and Schroder, 2006]. It's a basic algorithm that aims to explain how we can calculate a dual volume. In the PyDEC, this algorithm was optimized in order to get all circumcenters in the same matrix to gain in calculation time.

In the case of well-centered mesh, the discrete Hodge matrix is supposed to be a diagonal. But, as circumcentric duals may only be used if the mesh satisfies the Delaunay criterion, in PyDEC was also implemented a approached Hodge star defined to be the Whitney mass matrix (sparse matrix implementation for low-order finite element exterior calculus). However, it this case, as dual edges

Algorithm 2 Computation of dual volumes

function C($x_0, x_1, \cdots, x_n$)
    `To compute the space coordinates of set of points` $x_0, \cdots, x_n$
end function
procedure CALCDUAL(*simplicialComplex*)
    `Initialize dualVolumes to 0.`
    for `each primal face` $f \in S_2$ **do** ▷ Dual edges
        for `each primal tet` t_f `incident on` f **do**
            $b \leftarrow 1.$ ▷ Dual volume of primal tet = 1.
            $h \leftarrow ||C(f) - C(t_f)||$ ▷ length of edge
            $f.dualVolume \leftarrow f.dualVolume + \frac{1}{1} \cdot b \cdot h$
        end for
    end for
    for `each primal edge` $e \in S_1$ **do** ▷ Dual polygons
        for `each primal face` f_e `incident on` e **do**
            $b \leftarrow f_e.dualVolume$ ▷ base
            $h \leftarrow ||C(e) - C(f_e)||$ ▷ height
            $e.dualVolume \leftarrow e.dualVolume + \frac{1}{2} \cdot b \cdot h$ ▷ Area of triangle = $\frac{1}{2}bh$
        end for
    end for
    for `each primal vertex` $v \in S_0$ **do** ▷ Dual polyhedrons
        for `each primal edge` e_v `incident on` v **do**
            $b \leftarrow e_v.dualVolume$ ▷ base
            $h \leftarrow ||C(v) - C(e_v)||$ ▷ height
            $v.dualVolume \leftarrow v.dualVolume + \frac{1}{3} \cdot b \cdot h$ ▷ Volume of tet = $\frac{1}{3}bh$
        end for
    end for
end procedure

are no longer straight lines (they are piecewise linear), dual faces are no longer planar, and dual cells are no longer necessary convex.

As a Hodge star is diagonal at his counterpart is sparse matrix, in this way, we can control the quality of mesh and avoid degenerated mesh cases. It must be noted that for physical problem, the Hodge star $\star$ in (3.9) must be weighted by a material property, for example, for an electrokinetics problem as proposed for solving the electrical conductivity σ gives the Hodge star $\star_\sigma = \sigma \cdot \star$ (see example presented on the end of this chapter).

3.5 Implementation and example of application for our cochain method

In resent years, more and more new algorithms used for numerical computation of field are proposed for solving physical problems with approximation of exterior derivatives and algebraic topology theory. This section gives more details on our implementation and illustrates it with several examples of multiphysics

applications.

As noted previously, the computation scheme relies on Tonti diagram and coboundary operators are described to replace the *grad*, *curl*, *div* operators applied on computational meshes.

As explained previously, the property that connect between the most significant analytic properties of a algebraic topology and the first-order differential operators expresses in a *duality* relationship. From discrete analogs of analytic properties, we can express first order operators, which budded as the *primal* differential operators ($\bar{grad}$, $\bar{curl}$ and $\bar{div}$), *i.e.* operators to be applied on primal mesh and corresponding *dual* differential operators ($\tilde{grad}$, $\tilde{curl}$ and $\tilde{div}$), *i.e.* operators to be applied on the dual mesh. To make explicit, the link between classical differential operators and coboundary operators (discrete exterior derivatives), the notation proposed by E. Tonti and the authors of Cell-Method is adopted [Alotto et al., 2013] for defining corresponded matrices:

$$\bar{grad} \rightarrow \bar{d}_0 = \bar{\delta}_0 = \bar{G}; \qquad \tilde{grad} \rightarrow \tilde{d}_0 = \tilde{\delta}_0 = \tilde{G} \tag{3.10}$$

$$\bar{curl} \rightarrow \bar{d}_1 = \bar{\delta}_1 = \bar{C}; \qquad \tilde{curl} \rightarrow \tilde{d}_1 = \tilde{\delta}_1 = \tilde{C} \tag{3.11}$$

$$\bar{div} \rightarrow \bar{d}_2 = \bar{\delta}_2 = \bar{D}; \qquad \tilde{div} \rightarrow \tilde{d}_2 = \tilde{\delta}_2 = \tilde{D} \tag{3.12}$$

The constructive use of discrete *duality* relations is one of the major design principles of this class of methods [Lipnikov et al., 2014], if we assumed for orientation of volume a positive onwards orientation, one have [Ferretti, 2014]:

$$\tilde{D} = \bar{G}^T, \quad \tilde{C} = \bar{C}^T, \quad \tilde{G} = \bar{D}^T \tag{3.13}$$

If we assume for the orientation of volume, the outward orientation (as usual in mechanics), one relation changes as $\tilde{D} = -G^T$. [1]

The constructive use of duality relationships (3.13) is one of the major design principles of the groups of numerical methods based on exterior derivatives and algebraic topology. In [Lipnikov et al., 2014], the authors prove that topological invariants are naturally preserved:

> One of the most important analytic properties of many continuum equations is the conservation of the total energy. In *fluid dynamics* this property follows from fact that **grad** operator is the negative adjoint to the **div** operator. In *electromagnetics*, it follows from the self-adjointness of the **curl** operator [...]. This duality of the differential operator is transformed to the duality of the discrete operators. Another important analytic property is the *geometric conservation law* in fluid and solid dynamics. The framework [...] ensures that the discrete divergence of the velocity is consistent with change of volume of a fluid parcel. [Lipnikov et al., 2014]

These properties allow not only to preserve flux and energy in simple way but also to avoid jump conditions (for example, in case of material separation surface) and to avoid singularities.

[1] To make the document easier to read, the bar over primal quantities will be omitted when no confusion is possible, *i.e.* $\bar{G} = G$.

To represent the physicals problems besides topological behaviors we need to introduce a metric space. This can be represented using exterior differential forms and discrete analogs of the exterior derivative and the Hodge $\star$ operator are constructed to take into account the metric of the spatial and the constitutive laws, *i.e.* the material properties.

Multiphysical application: Micro-electromechanical systems simulation To illustrate this section, we will present some examples of applications as well as the computational software designed during this thesis for field calculation. As explained previously, three steps are required for such a purpose:

- A mesh tools to build Delaunay-Voronoi dual meshes,
- A preprocessing tools to build boundary, co-boundary and Hodge star operators,
- A computational tools to build and solve mathematical equations of multiphysic problems.

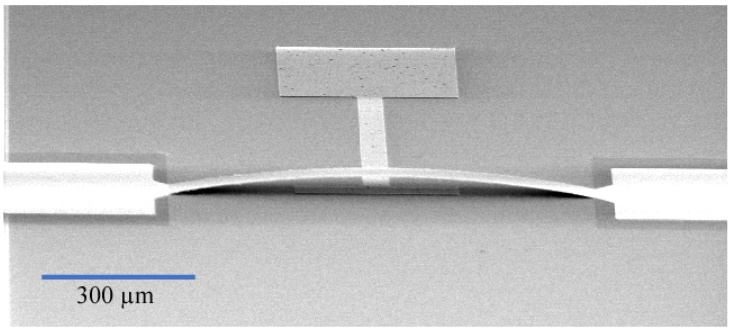
Except for the first part of the mesh tools (based on GMSH), all other tools have been developed in Python using either freely available packages (such as Numpy, Pydec, SciPy, Matplotlib and VTK) or by developing our own python packages. Postprocessing of results may be displayed by python tools (using mathplotlib package for example) or by using external softwares such as ParaView (based on VTK (Visualization ToolKit) standard).

The working of this software is nearly similar to DualLab, the software designed at the Politecnico di Torino, Italy for Cell-Method computations [Freschi et al., 2008]. The main differences are that DualLab is based on Matlab whereas our tools is based on Python. Some specifications are also different, for example, DualLab always uses barycentric meshes while we use the philosophy advocated by the discrete exterior calculus and therefore Voronoï's dual meshes when it is possible.

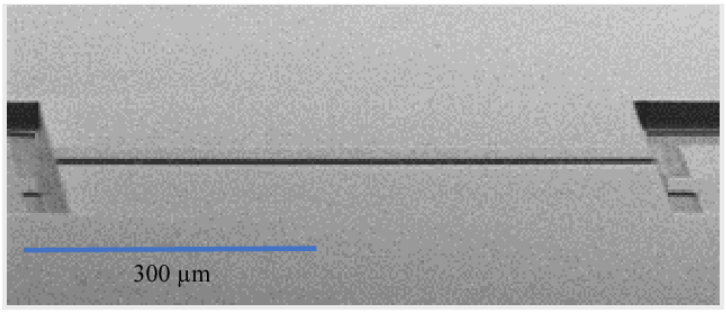
To illustrate the multiphysical potential of this development, we have chosen a multiphysical example similar to those tested by the Politechnico di Torino team, *i.e.* an example using a weak electro-thermo-mechanical coupling (for more details see [Delprete et al., 2010]). Our example is based on the working and geometry of a Micro-Electro-Mechanical System (MEMS).

The MEMS studied in this work is a technological device that generates a deformation by controlling an applied current in order to perform a sensor or actuator function. We show a electronic microscope picture of such type of actuators (Figure 3.18). Such systems are widely used in many technological fields such as automotive, aeronautics, medicine or biology. The Figure 3.19 presents an example of MEMS actuated by electrostatic effect. Nevertheless in this section we realized a model of MEMS actuated by thermal dilatation. We therefore study and compute an electro-thermal mechanical multiphysical problem.

The geometry of the beam used as MEMS is described in Figure 3.20. Figure 3.21 present the boundary conditions of electrokinetic problem. Figure 3.22 presents the boundary conditions a thermal problem and the Figure 3.23 the boundary conditions of the mechanical problem.



(a)



(b)

Figure 3.18: (a) The MEMS bridge condition after the release stage: electrode pad of size 20 μm $\times$ 160 μm is located underneath the suspended bridge. (b) The condition of the originally straight bridge structure before applying voltage [Latif et al., 2017].

The working of this device is as follow: due to the voltage applied between top and bottom surface of the beam, an electric current flows through this beam and generates Joules losses. These losses generate temperature gradients, which, by thermal expansion of the material, cause mechanical deformations. The physical and geometric properties of the device are detailed in Table 3.1. GMSH was used to mesh this geometry. Corresponding dual meshes are those represented on Figure 3.8 (a).

Propriety	Symbol	Value
Electrical conductivity	σ	$3.69{\cdot}10^{7} Sm^{-1}$
Coefficient of linear thermal expansion	α_T	$2.3{\cdot}10^{-5} K^{-1}$
Referencing temperature	T_0	$22\ C^o$
Thermal conductivity	λ	$237\ W \cdot (m \cdot K)^{-1}$
Young's modulus	E	$7.1{\cdot}10^{10} Pa$
Poisson's ratio	ν	0.344
Density	ρ	$2.7{\cdot}10^{3} kg \cdot m^{-3}$
Geometrical dimensions	$l \times w \times h$	$2.0{\times}0.2{\times}0.1\ cm$

Table 3.1: Physical and geometric characteristics of the MEMS.

As previously explained, for this example, Tonti Diagrams are great tools to present the physics and computations required for solving multiphysic computational problems [Alotto et al., 2010]. Our software has been tested on a case

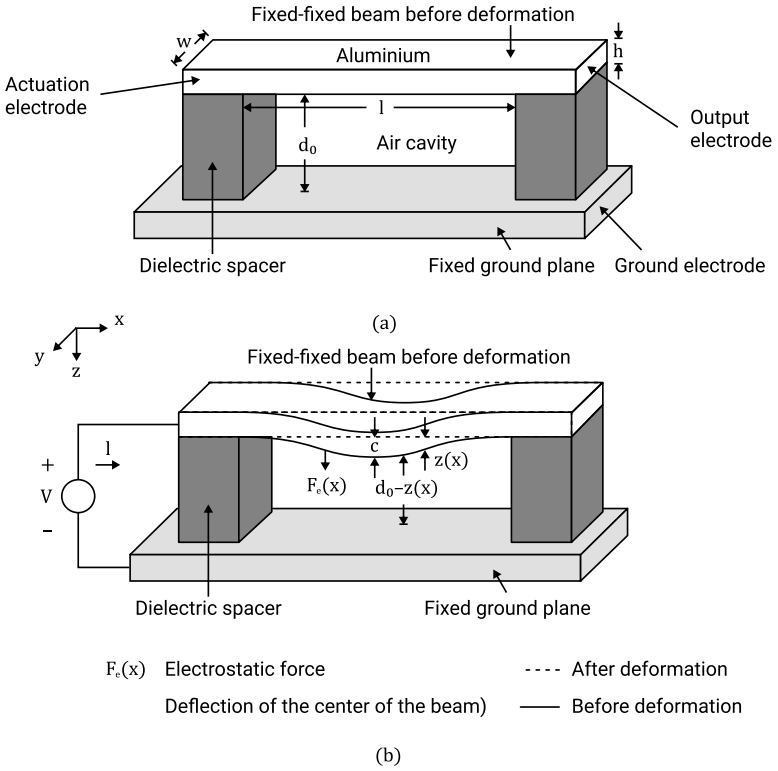
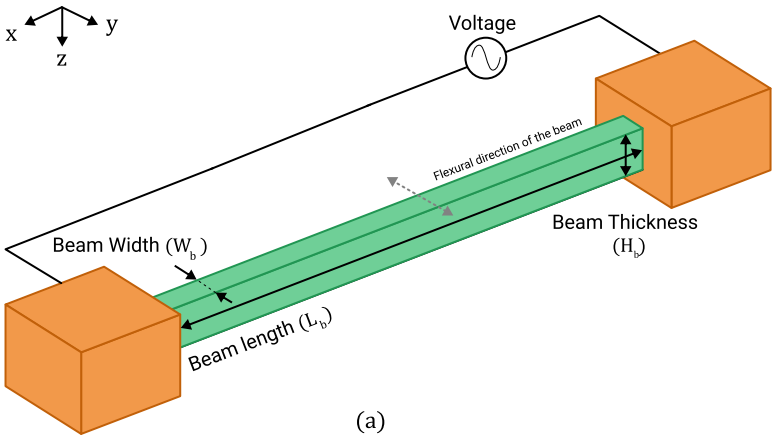


Figure 3.19: Geometry and boundary conditions of electrostatic MEMS: (a) Plot of a fixed-fixed beam separated from a fixed ground plane by dielectric spacers. (b) Plot showing the deformation profile of the beam due to an electrostatic force (constant drive voltage).

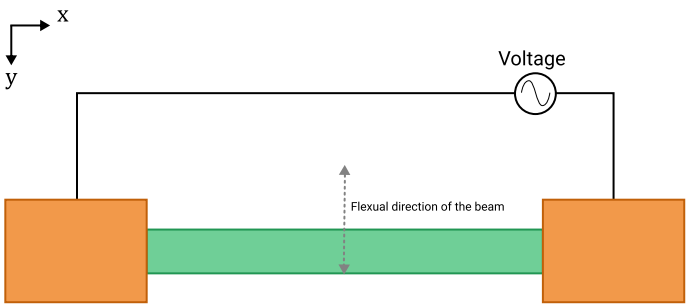
of a weak coupling between these 3 different physics: electrokinetic → thermal → elastostatic. Boundary conditions and geometries are resumed in Figure 3.21, Figure 3.22 and Figure 3.23.

The multiphysical coupling approach adopted in this work is the one of C. Delprete et al. [Delprete et al., 2010]. As can be noted, this illustration example only requires three sequential calculations (weak coupling), however this could be extended for strong coupling as done in [Delprete et al., 2013] using an iterative method (e. g. using a Newton-Raphson or fixed point method).

The first two of the three required sequential calculations have a similar computation structure while the third one has a slightly different structure, which implies a more complex calculation. Indeed, an electrokinetic calculation as well as a thermal calculation requires the computation of a scalar potential whereas this is not the case for mechanics. We will therefore present these two types of calculations separately, first the scalar field calculation (for electrical potential and temperature computation) and then the vector field calculation (displacement field computation). To make the link with the objects used in differential geometry and exterior calculus of E. Cartan, a scalar field is in fact a *scalar-valued* 0-form while a vector field is a *vector-valued* 0-form. This second case is more complex since the exterior derivative produces second order tensor fields, which is not the case for scalar-valued 0-form.



(a)



(b)

Figure 3.20: Schematic view of the thermal dilatation MEMS actuator.

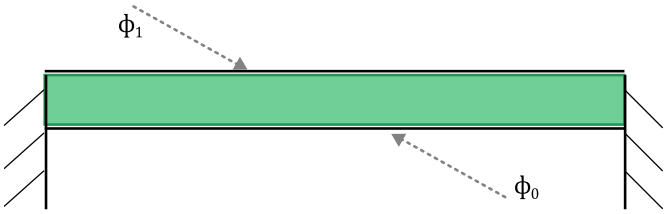


Figure 3.21: Physics of the electrokinetic problem: an electric potential ϕ_1 is imposed at the top surface while an other electric potential is imposed at the bottom surface of the beam. Others faces (front, rear, left and right) are insulated (no current flow is possible).

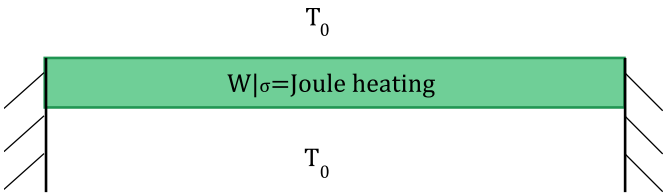


Figure 3.22: Physics of the thermal problem: imposed ambient temperature T_0 on boundary conditions and internal volumetric heat sources, produces by Joule's effect.

3.5.1 Scalar fields computation

At first, let us consider the current flow problem (electrokinetics). The physics of this problem is displayed on the Tonti diagram of Figure 3.24.

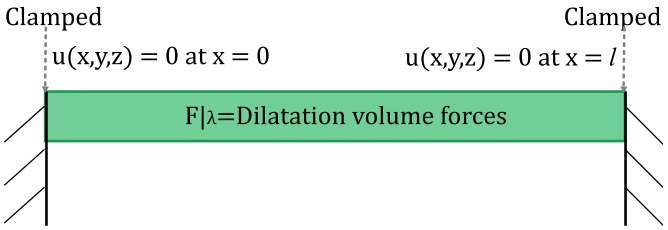


Figure 3.23: Physics of the elastostatic problem: displacements clamped at the right and left ends of the beam (clamped condition) while the other surfaces of the beam are free of stress. A volume force is generated by the thermal expansion into the beam.

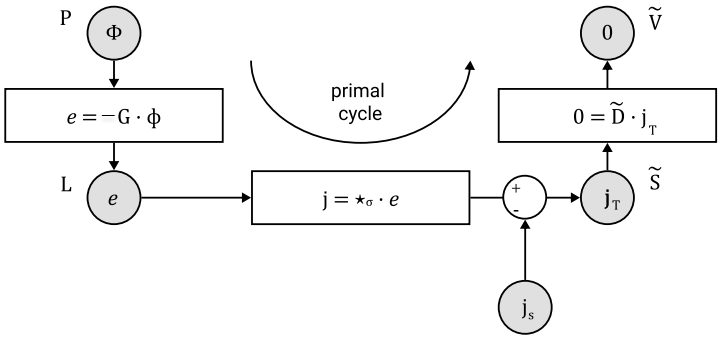


Figure 3.24: Tonti diagram of the current flow problem.

Two problems can be attached to this Tonti diagram:

- Neumann problem: finding the electric potential distribution ϕ considering a known current source j_s,
- Dirichlet problem: finding the current distribution j_T once some electric potential values are known at boundary conditions.

In the framework of CM or DEC method, we can formulate easily these two types of problems [Alotto et al., 2013]. If the physics problem corresponds to Neumann electrokinetics problem (currents j_s are imposed for Neumann problem). By using the Tonti diagrams of Figure 3.24, it is easy to see that :

$$\tilde{D} \cdot (\star_\sigma \cdot G \cdot \phi) = -\tilde{D} \cdot j_s \tag{3.14}$$

with $\tilde{D}$ the dual discrete divergence operator (volume-face dual incidence matrix equal to G^T), $\star_\sigma$ the Hodge star operator corresponding to the electrical conductivity constitutive law of the physical problem, G the discrete gradient operator (edge-node primal incidence matrix) and j_s the imposed currents flowing through dual surfaces [Tonti, 2001b].

For a Dirichlet electrokinetic problem, current sources are null but boundary conditions are imposed on ϕ (this is the case for our MEMS, see Figure 3.21) which reduces this equation to:

$$\tilde{D} \cdot \star_\sigma \cdot G \cdot \phi = 0 \tag{3.15}$$

In the classical framework of electromagnetism, these two equations correspond to standard Laplace or Poisson equations:

$$\begin{aligned} \nabla \cdot (\frac{1}{\rho} \nabla \phi) &= 0 \qquad \text{(Laplace equation)} \\ \nabla \cdot (\frac{1}{\rho} \nabla \phi) &= -\nabla \cdot \vec{J}_s \quad \text{(Poisson equation)} \end{aligned} \tag{3.16}$$

with $\sigma = \frac{1}{\rho}$ the electrical conductivity of the materials (inverse of the resistivity ρ), ϕ the electric scalar potential and $\vec{J}_s$ the imposed electric current. Let us note that the $(\tilde{D} \cdot \star_\sigma \cdot G)$ operator corresponds to the standard *Stiffness Matrix* of the Finite Elements Method.

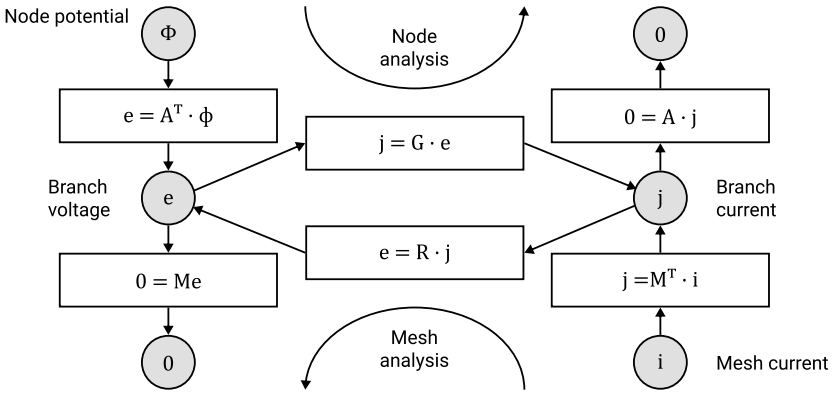


Figure 3.25: Tonti diagram of electrical circuits: two ways to travel it, using ϕ, potentials at nodes (Node analysis) or using $\mathbf{i}$, mesh currents (mesh analysis).

If we compare our formulation with the Tonti diagram of electric circuit displayed on Figure 3.25, we see exactly the same type of equations [2]:

$$\mathbf{A} \cdot \mathbf{G} \cdot \mathbf{A}^T \cdot \phi = 0 \tag{3.17}$$

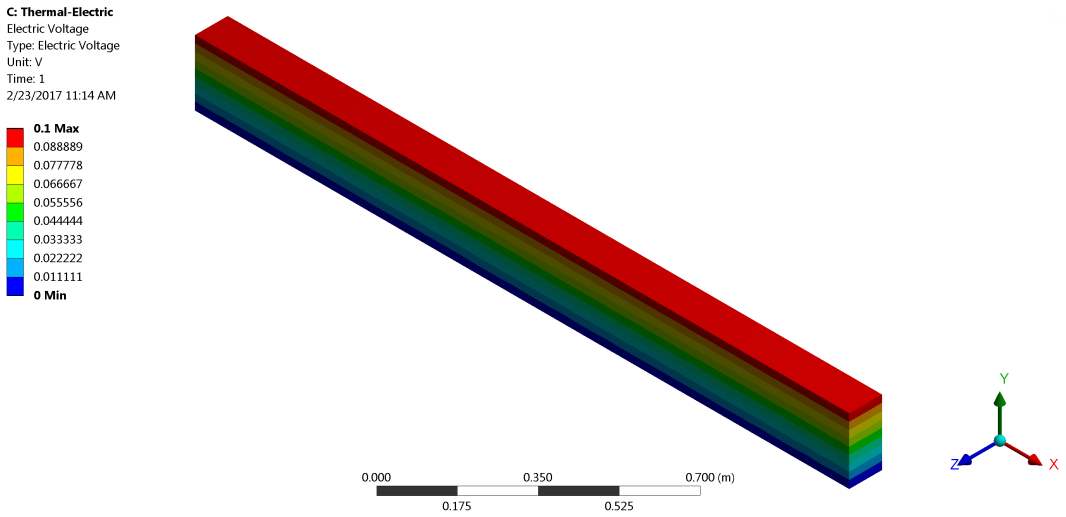
with $\mathbf{A}^T$ the branch-to-node incidence matrix and $\mathbf{A}$ the node-to-branch incidence matrix characterizing Kirchhoff voltage and current laws. Therefore, the branch-to-node incidence matrix $\mathbf{A}^T$ is identical to the discrete gradient G of (3.15) and its dual $\mathbf{A}$ is identical to the dual discrete divergence operator $\tilde{D}$. As a consequence, solving methods are fully identical and we can use *standard* node methods of electric circuit theory to solve both problems.

As a conclusion, we can say that formulations of equations are completely similar whether it is a field calculation problem (discretized on a mesh) or whether it is the calculation of a lumped components electric circuit. This similarity makes it possible to propose a high degree of consistency between the two formulations and very similar resolution tools. It is this similarity that will be exploited in the next chapter to make model reduction and automatic generation of lumped

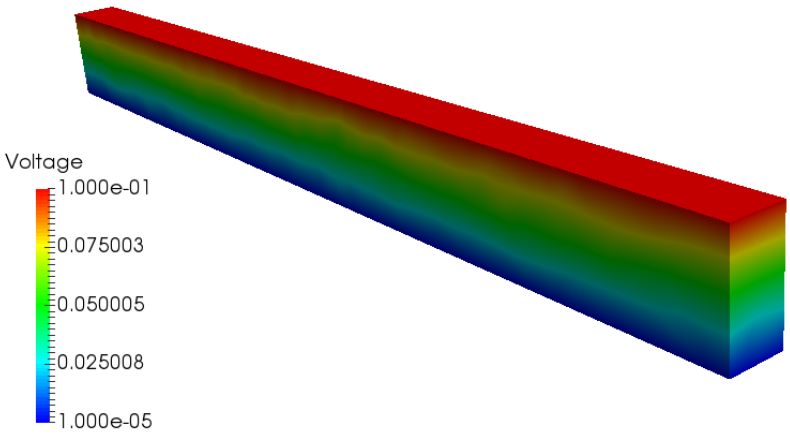
[2] Be careful, in the equation 3.17, $\mathbf{G}$ is a conductance matrix and not a gradient operator of primal mesh!

models from fields computation. The analogies between circuit and fields are not new, and can be found in the pioneer work of Branin published in [Branin, 1966]. The use of an algebraic structure is based on global variables which are the same as those used in the circuit theory. Voltages and currents are primal and dual cochains and they are in fact obtained by integrating the local quantities on the spatial element they belong to.

For validating our implementation of cochain method, a finite element model developed under Ansys was used as a reference model. The simulation results of Ansys and our cochain calculation software are presented in Figure 3.26. As we used ParaView software as the means of visualization, the color chart is not the same as in Ansys: reader can view color bar corresponding to calculus on the left of each image.



(a) Ansys.



(b) Cochain method.

Figure 3.26: Comparison between our computational results and the results given by the FEM software ANSYS: Electrokinetic problem. Display of the voltage field

As we can see from these two calculations (Figure 3.26), the results are very close and it permits to validate our solver.

The second calculation to be performed for this illustrative example is that of the θ temperature at nodes of the primal mesh when the material is subjected to a heat sources p_v generated by Joules effects. Joules effects are heat power volume sources resulting from the energetic product of duality between voltages at primal edges and currents at dual faces. Heat sources can be calculated using the results of the previous electrokinetic calculation and then assigned to the dual volumes (heat power are source variables). For this calculation, we have used the methodology proposed in [Alotto et al., 2013].

The physics of this thermal problem is displayed on the Tonti diagram of Figure 3.27 with θ the temperature at primal nodes, $\Delta\theta$ the temperature difference across primal edges, $\star_\lambda$ the thermal conductivity Hodge star operator (constitutive law), Φ the thermal flux through dual faces and p_v the imposed heat power sources defined on the dual volumes.

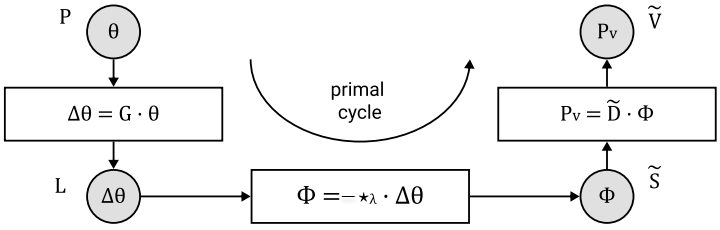


Figure 3.27: Tonti diagram of a thermal problem.

As can be seen, this Tonti diagram is very similar to the one of the electrokinetic problem except that dual volumes sources (heat power) are present instead of dual edges sources (currents) for electrokinetics. The formulation of equation are therefore very similar as well as solving methods:

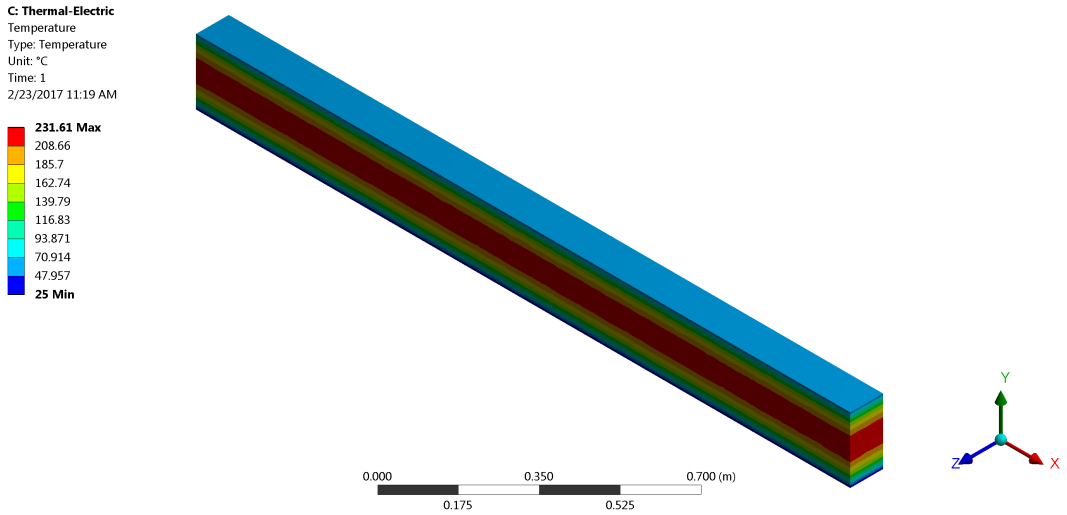
$$\tilde{D} \cdot \star_\lambda \cdot G \cdot \theta = p_v \tag{3.18}$$

The computation results obtain with our cochain method and ANSYS are reported on Figure 3.28. The results are very closed as seen on the maximal values on temperature and heat flow reported on Table 3.2.

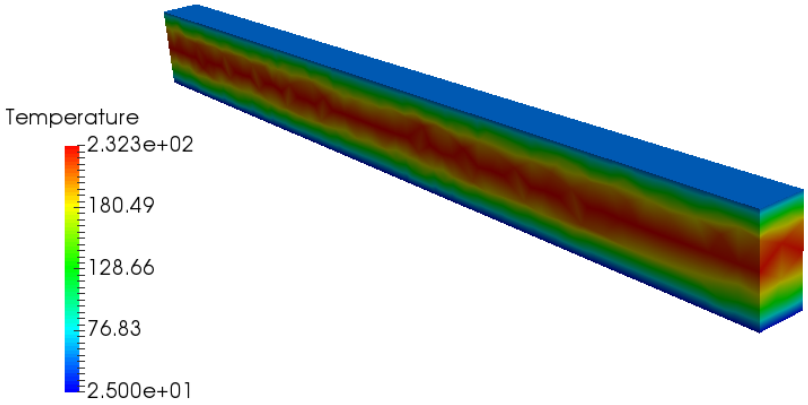
3.5.2 Vector fields computation

In previous subsection, We have seen that the computation of scalar fields on meshes is quite easy in terms of matrix operation. This is especially the case because if dual meshes are orthogonal (this is the case for Delaunay-Voronoi meshes), constitutive matrices – *i.e.* the Hodge star $\star$ – are diagonal matrices. In addition, it is quite easy to obtain the topological matrices G, C, D and $\tilde{G}$, $\tilde{C}$, $\tilde{D}$ using the PyDEC approach and package freely available. Unfortunately, this is no more so simple for tensor fields computation like those we meet in mechanics.

Let us present in a few words the origin of this difficulty for mechanical fields computation. The fundamental problem of elastostatics can be stated as follows:



(a) Ansys.



(b) Cochain method.

Figure 3.28: Comparison between our computational results and the results given by the FEM software ANSYS: Thermal problem. Display of the temperature field.

given an elastic solid in an assigned reference configuration, given the volume forces, the external surfaces forces, the material and the constraints, find the deformed configuration and the stress distribution within the solid. The main unknowns of the problem are the displacements in each point of the domain from their reference configuration, *i.e.* the displacement vector $\vec{u}(t, \mathbf{P}) = \mathbf{u}(t, \mathbf{P})$ for each point $\mathbf{P}$ of the solid (see Figure 3.29 from [Tonti, 2013]).

In our example, we suppose in the sequel a static mechanical problem. As seen in this Figure 3.29, the degrees of freedom for an elastostatic problem are the displacement field $\vec{u}(\mathbf{P}) = \mathbf{u}(\mathbf{P})$. Contrary to the previous electrokinetic and thermic problems, this one requires the computation of a vector fields instead of a scalar field at each point $\mathbf{P}$. In the framework of exterior differential calculus proposed by E. Cartan, this field is therefore a *vector-valued* 0-form instead of a *scalar-valued* 0-form. Therefore, if we compute the exterior derivative of this

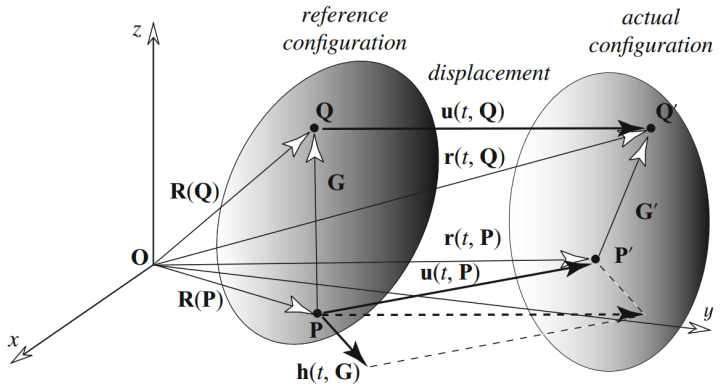


Figure 3.29: Position of an elastostatics problem according to [Tonti, 2013]: two particles $\mathcal{P}$ and $\mathcal{Q}$ in the initial configuration occupy the positions $\mathbf{P}$ and $\mathbf{Q}$. The same particles, at a later instant, occupy the positions $\mathbf{P}'$ and $\mathbf{Q}'$. The vectors in thin lines are position vectors, whereas those in thick lines are the displacements of the two particles.

0-form, we do not obtain a scalar-valued 1-form but a covariant tensor of rank 2. Moreover, this deformation gradient tensor (noted $\nabla\mathbf{u}(P)$ in the classical calculus framework) contains two types of informations, one related to rigid body rotation (antisymmetric part) and one related to deformation/strain (symmetric part). Because for elastostatic we are only interested in this second part, rigid body rotation must be removed (in elastostatic, constitutive laws links strain to stress independently of rigid body motions). Hence, we must separate rigid body contribution from strain contribution by computing intermediate quantities. If small perturbation are considered, the second rank covariant strain tensor ε is constituted by the symmetric part of the displacement gradient:

$$\varepsilon = \frac{1}{2}(\nabla\mathbf{u} + \nabla^T\mathbf{u}) = \nabla^{sym}\mathbf{u} \tag{3.19}$$

Moreover to transform tensor quantities into vector quantities, it is of common use in computational mechanics [Cook et al., 1989] to introduce the Voigt notation. These remarks on configuration variables (displacement related variables) can also be applied to source variables (force related variables). This requires the definition of two contravariant tensor of rank 2, one symmetric (with no volume torque) and one non-symmetric (that may include volume torque). These two tensors can also be rewritten as vectors using the Voigt notation.

Hence, with these modifications, the elastostatic problem can be described by the Tonti diagram of the Figure 3.30. The notations are those proposed by Tonti et al. in [Tonti and Zarantonello, 2009].

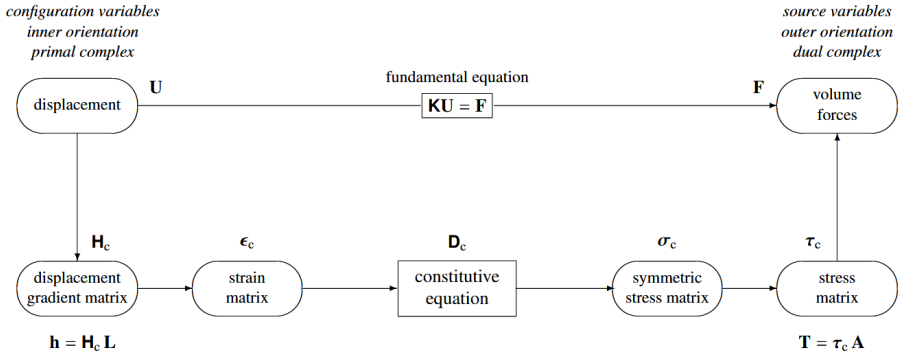
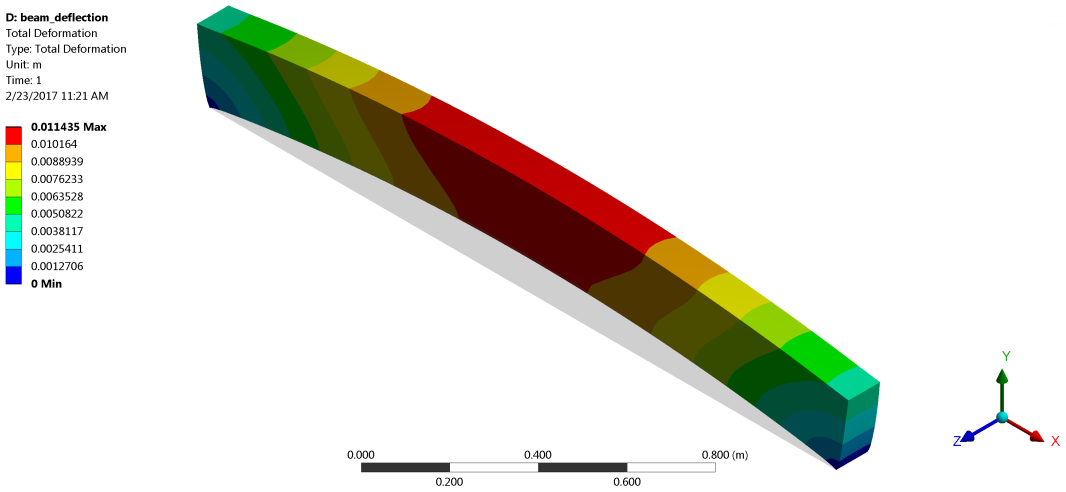
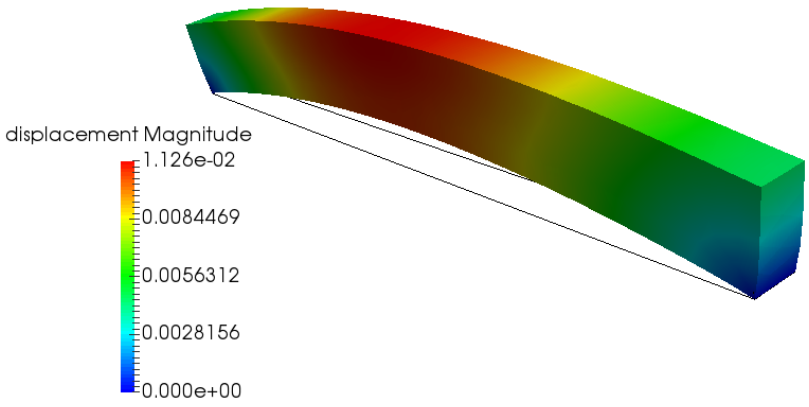


Figure 3.30: Tonti diagram of the elastostatics problem [Tonti and Zarantonello, 2009].



(a) Ansys.



(b) Cochain method.

Figure 3.31: Comparison between our computational results and the results given by the FEM software ANSYS: Mechanical problem. Display of the displacement field.

The full description of the implementation of this computation will not be detailed in this manuscript because it mimics the implementation realized in DualLab. This general procedure is then explained in [Alotto et al., 2013]. Let us note that in this work, first order interpolations are used but higher order interpolations could be used to improve the precision as it was done in [Cosmi, 2001] for plane elasticity problems. This paper also shows that *stiffness* matrix obtain in such a way is exactly the same that those given by FEM.

Results of the deflection of the beam compared with our cochain method and ANSYS results are reported on Figure 3.31. Maximal deflections are reported in Table 3.2.

Conclusion on our cochain method for fields problems computation. In the Table 3.2 we can see the difference between values calculated by our cochain method and Ansys (Finite-Element (FE) method): the results obtained were compared. The differences of less than 3% (see Table 3.2), showing the relevance of cochain method and their viability compared to conventional FE techniques (commercial code).

Parameter	Scale	Ansys	Cell-Method
Max Temperature	C^o	231.6	232.3
Max displacement	m	$1.144{\cdot}10^{-2}$	$1.126{\cdot}10^{-2}$
Max heat flow	W/m^2	$6.7904{\cdot}10^5$	$6.69{\cdot}10^5$

Table 3.2: Comparison of the results between ANSYS and the cochain method calculation.

By referring to the Figures 3.26, 3.28 and 3.31, a good agreement can be highlighted between FEM and simulated mechanical, electrical and thermal quantities by cochain method. As a conclusion, it is possible to say that cochain method has proven to be effective in multiphysics approach. In fact it gives coherent results in a simple environment. Once the Tonti diagram is assessed, the field problem can be built by making reference to linear algebra of sparse matrices, with the same algebraic operators for all theories.

Chapter 4

From fields to lumped models

This chapter aimed to introduce a new method of model reduction via Artificial Intelligence-based methods, such as data analysis, machine learning (ML), classification or clustering methods. The idea behind this model order reduction is quite new and original. Moreover, these methods are gaining popularity with the progress of Artificial Intelligence (AI). To show the importance of AI methods for physical modeling, recently (July 23, 2019), Google AI researcher center has published a paper called "Learning Better Simulation Methods for Partial Differential Equations" [Bar-Sinai et al., 2019]. In this paper, they introduce data-driven discretization, a method for learning optimized approximations to PDEs based on actual solutions to the known underlying equations. This approach uses neural networks to estimate spatial derivatives, which are optimized to best satisfy the equations on a low-resolution grid. In general, they used a classical approach "PDEs $\Rightarrow$ discretization on mesh $\Rightarrow$ approximate solution $\Rightarrow$ reducing mesh $\Rightarrow$ surrogate model". Compared to this classical scheme, we propose in this chapter a completely new approach, which can transform a spatial mesh model to lumped-parameter model by using unsupervised clustering technique that helps to retain the topological structure of lumped system networks.

Here, in the last chapter of this thesis, we can summarize all work done to implement our method (see Figure 4.1). Our methodology consists of the following stages:

1. Geometry and description of materials: creating numerical geometry of the device. It can be done in any CAD-software. For our purpose we used a free software FreeCAD.

2. Initial mesh: to discretize the geometry with *GMSH* (described in chapter 3).

3. Re-meshing by Geogram software (if necessary): we need dual meshes with orthogonal properties as described in chapter 3. The use of Geogram happens if we have complex 3D geometry of the devise and if *GMSH* does not succeed to do this.

4. PyDEC: we used it for calculating the adjacency matrices to get the best performance to start Cochain-Method calculation (see chapter 3).

5. Cochain-Method: we developed our "home-made" solver of cochain method to calculate discrete exteriors forms.

6. Clustering: used for the first step of generating lumped model. It helps to identify elements of the circuit and theirs interconnections. As it will be explained in this chapter, this step requires a phase of pre-processing of data (interpolation of field density with Whitney elements).

7. Reduced Model: as the circuit model may be vary large, in this chapter we propose some special techniques to reduce the dimensions of the lumped model.

8. Use of reduced models: once we get a reduced lumped model, we can used the solvers dedicated to system-level modeling, Spice-like, optimization or design tools.

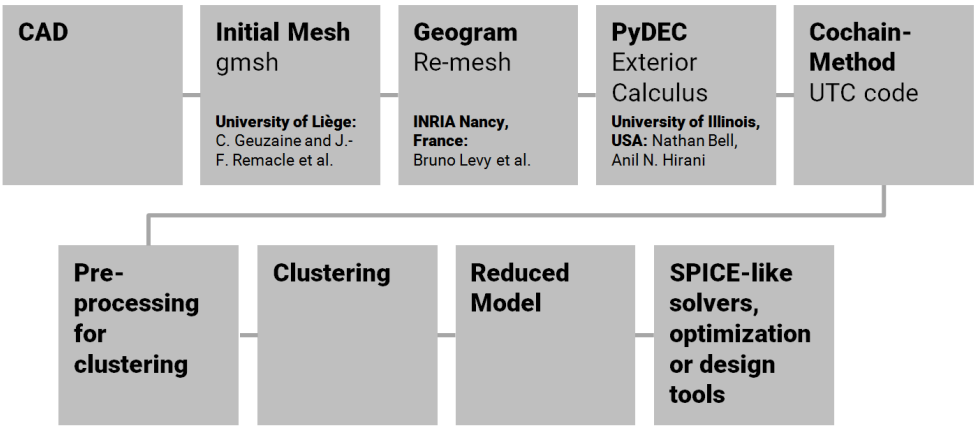


Figure 4.1: The complete structure displaying all stages of our modeling approach: from geometrical models via Cochain Method to lumped models.

To better present our method, we take for instance a electromagnetic relay illustration in this chapter. The aim is to generate a reluctance circuit from magnetostatic field. The choice of this physics was also done to complete the multiphysics aspect of our tools since we have already dealt with electrical, thermal and mechanical examples in the previous chapter.

Before presenting our new approach, it is important to introduce classical approaches for model reduction as well as common data analysis tools in the framework of AI. This will allow us to emphasis the originality of our approach.

4.1 Review of the main model reduction methods

With the increase of the capacity of computers used for numerical simulations the amount of produced data has increased as well. These large amounts of data have to be analyzed to gain a better understanding of the simulated processes. Moreover, for an efficient use of these data, they have to be reduced to highlight their main contents.

A good introduction of model reduction methods is given in [Schilders, 2008] in the field of electronic components design. As explained by W. Schilders in this paper:

> Model Order Reduction tries to quickly capture the essential features of a structure. This means that in an early stage of the process, the most basic properties of the original model must already be present in the smaller approximation. At a certain moment the process of reduction is stopped. At that point all necessary properties of the original model must be captured with sufficient precision. All of this has to be done automatically.

Only two of the main model order reduction techniques will be reviewed in the following (POD and PEEC methods) because they contains the main characteristics of almost all other model order reduction methods. What we will see from this review is that these methods always have at least one limitation: either the topological structure of the problem is lost during the reduction, or it is not possible to make a significant reduction of the degree of freedom without losing a large part of the essential properties of the original model. The unique method proposed in this thesis and described in the second part of this chapter has been devoted to describe a method that can avoid these limitations.

4.1.1 Proper Orthogonal Decomposition Method

The Proper Orthogonal Decomposition (POD) is a multi-variate statistical method that aims at obtaining a compact representation of the data. This method may serve two purposes, namely order reduction by projecting high-dimensional data into a lower-dimensional space and feature extraction by revealing relevant, but unexpected, structure hidden in the data [Kerschen et al., 2005].

The POD, also known as the Karhunen-Loève decomposition (KLD), was proposed independently by several scientists including Karhunen [Karhunen, 1947], Kosambi [Kosambi, 1943], Loève [Loeve, 1948], Obukhov [Obukhov, 1954] and Pougachev [Pugachev, 1953]. It was originally conceived in the framework of continuous second-order processes.

The mathematical formulation of the POD can be found in reference [Holmes et al., 2012]. Hence, we only presents the basics algorithms.

The physical interpretation of POD. The POD is directly computed from the system response. The signal-dependent nature of the POD can be seen as one of the weakest points of the method. This prevents us from providing a general physical interpretation of the modes extracted from the decomposition.

In terms of statistics theory, we can formulate POD method in following way. The signal $u(x)$ (a random field $u(x)$ depends on x of zero average) are discretized in space and time. Accordingly, n observations of a m-dimensional vector x are collected, and an $(m \times n)$ response matrix is formed:

$$X = \begin{bmatrix} x_1 & \dots x_n \end{bmatrix} = \begin{bmatrix} x_{11} & \dots & x_{1n} \\ \vdots & \vdots & \vdots \\ x_{m1} & \dots & x_{mn} \end{bmatrix} \tag{4.1}$$

The idea of POD is to define a new base Φ that is "more adapted" to the signal $u(x)$ discretized into matrix representation X in restricted domain D. The "more

adapted" means that the base functions $\phi_i \in \Phi$ seek to maximize the average of the signal projection $u(x)$ on the functions of this new base Φ:

$$Max_\phi \rightarrow \frac{< (u|\phi) >}{(\phi_i|\phi_j)} \tag{4.2}$$

In (4.2) we recognize a Euler-Lagrange problem of maximizations under constraint whose solution is a first order Fredholm integral equation:

$$\int_D < u(x), u(x') > \phi_j(x')dx' = \lambda_j \phi_j(x) \tag{4.3}$$

where λ_j are eigenvalues of X.

The signal is therefore written as the sum of spatial functions whose amplitude is random:

$$u(x) = \sum_{j=1}^{n} \lambda_j \phi_j(x) \phi_j * (x') = \sum_{j=1}^{n} a_j \phi^j(x) \tag{4.4}$$

Once we transform our matrix to eigenvalues basis, we can investigate it. One of the technique to reduce the matrix may be to sort out the values in X' by the eigenvalues λ: if λ_j is small that means that corresponding values x_{jn} is in the hight frequency range and so can be neglected.

To resume we can write the following formula:

$$u(x) = \sum_{j=1}^{n} a_j \cdot \phi_j \rightarrow u'(x) = \sum_{j=1}^{n} a'_j \cdot \phi'_j \rightarrow \hat{u'(x)} = \sum_{j=1}^{p<n} a'_j \cdot \phi'_j \tag{4.5}$$

where $(p < n)$ is reduced base.

To finish, let us note some remarks:

Remark 1: POD applies to a field with a zero average field, which means that in practice the average field will have to be subtracted from the total field. If the average field strength is a function of the problem itself, the results of the procedure will be unchanged whether the total signal is considered total or purely fluctuating.

Remark 2: The choose of ϕ orthogonal eigenvalues functions is not always the best way to represent a new system. Sometimes some additional *empiric* functions are needed.

Remark 3: The extraction of the proper functions is a totally autonomous and objective procedure, which does not use any internal or external criteria for conditioning signal (trigger threshold).

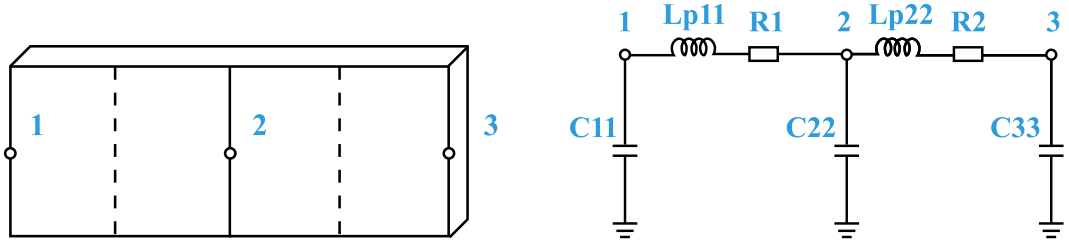
4.1.2 Partial element equivalent circuit method

Partial Element Equivalent Circuit (PEEC) is a reduction method more and more used in CEM applications. Its purpose is to generate a lumped model from Field elementary computation. It helps to express a distributed electromagnetic model

in a electric circuit model, and then solved by some standard circuit solver software. Here, *equivalent circuit models* are derived from an integral equation to establish an electrical description of the *physical geometry*. This helps to ensure functionality of electric systems and compliance with electromagnetic compatibility, *i.e.* it facilitates the solution of problems which have both an electromagnetic part as well as a circuit part. The models can be used in both the time as well as the frequency domain.

The pioneer of PEEC approach was Albert E. RUEHLI [Ruehli, 1974]. Originally,when the foundation of the PEEC method was presented, it was a method to compute partial inductions. The PEEC method was extended to more generalized problems, including dielectric material and retardation effect.

The PEEC method is not one of the most common techniques used in electromagnetic simulation software or as a research area but it has just been starting to gain recognition and for the first time there is a session at the 2001 IEEE EMC Symposium named after the technique. In the mid 90's, two researchers from the University of L'Aquila in Italy, Professor Antonio Orlandi and Professor Giulio Antonini, published their first PEEC paper and are now together with Dr. Ruehli considered the top researchers in the area. Starting year 2006, several research projects have been initiated by the faculty of Computer Science and Electrical Engineering of Luleå University of Technology in Sweden in the focus area of PEEC with the emphasis on computer based solvers for PEEC under the name MultiPEEC [wikipedia. PEEC, 2018].



(a) An orthogonal metal strip with 3 nodes and 2 cells.

(b) The corresponding PEEC circuit.

Figure 4.2: Partial element equivalent circuit method (PEEC)

The theoretical derivation starts from the expression of the total electric field in free space $\overrightarrow{E^T}(\vec{r},t)$, by using the magnetic vector and electric scalar potentials, $\vec{A}$ and ϕ respectively [Ruehli, 1974].

$$\overrightarrow{E^T}(\vec{r},t) = \overrightarrow{E^i}(\vec{r},t) - \frac{\partial \vec{A}(\vec{r},t)}{\partial t} - \nabla\phi(\vec{r},t) \tag{4.6}$$

where $\overrightarrow{E^i}$ is a potential applied external electric field. If the observation point, $\vec{r}$, is on the surface of a conductor, the total electric field can be written as:

$$\overrightarrow{E^T}(\vec{r},t) = \frac{\vec{J}(\vec{r},t)}{\sigma} \tag{4.7}$$

in which $\vec{J}(\vec{r},t)$ is the current density in a conductor and σ is the conductivity. Combining 4.6 and 4.7 results in

$$\vec{E^i}(\vec{r},t) = \frac{\vec{J}(\vec{r},t)}{\sigma} + \frac{\partial \vec{A}(\vec{r},t)}{\partial t} + \nabla\phi(\vec{r},t) \tag{4.8}$$

To transform (4.8) into the electric field integral equation (EFIE) the definitions of the electromagnetic potentials, $\vec{A}$ and ϕ are used. The magnetic vector potential, $\vec{A}$ at the observation point $\vec{r}$ is given by [Peterson et al., 1998]:

$$\vec{A}(\vec{r},t) = \sum_{k=1}^{K} \mu \int_{v_k} G(\vec{r},\vec{r}\,')\vec{J}(\vec{r}\,',t_d)dv_k \tag{4.9}$$

in which the summation is over K conductors and μ is the relative permeability multiplied by permeability of vacuum. The free space Green's function is used and can be defined as [Peterson et al., 1998]:

$$G(\vec{r},\vec{r}\,') = \frac{1}{4\pi}\frac{1}{|\vec{r}-\vec{r}\,'|} \tag{4.10}$$

In (4.9) $\vec{J}$ is the current density at a source point $\vec{r}\,'$ and t_d is the retardation time between the observation point $\vec{r}$ and the source point given by

$$t_d = t - \frac{|\vec{r}-\vec{r}\,'|}{c} \tag{4.11}$$

where $c = 3 \cdot 10^8 m/s$. The electrical scalar potential ϕ at the observation point $\vec{J}$ is given by [Peterson et al., 1998]:

$$\phi(\vec{r},t) = \sum_{k=1}^{K} \frac{1}{\epsilon_0} \int_{v_k} G(\vec{r},\vec{r}\,')q(\vec{r}\,',t_d)dv_k \tag{4.12}$$

where ϵ_0 is the vacuum permittivity and q is the charge density at the source point. Combining (4.8), (4.9) and (4.8) results in the well known electric field integral equation (EFIE) or mixed potential integral equation (MPIE) that is to be solved according to

$$\hat{n} \times \vec{E}^i(\vec{r},t) = \hat{n} \times (\frac{\vec{J}(\vec{r},t)}{\sigma}) + \hat{n} \times (\sum_{k=1}^{K} \mu \int_{v_i} G(\vec{r},\vec{r}\,')\frac{\partial \vec{J}(\vec{r}\,',t_d)}{\partial t}dv_k) + \\ + \hat{n} \times (\sum_{k=1}^{K} \frac{\nabla}{\epsilon_0} \int_{v_k} G(\vec{r},\vec{r}\,')q(\vec{r}\,',t_d)dv_k) \tag{4.13}$$

where $\hat{n}$ is the surface normal to the body surfaces. In the PEEC method the EFIE, (4.13) is discretized using a method of moments process, interpreted as an equivalent circuit and solved using circuit theory. The formulation (4.13) is the basic discretized version of the electric field integral equation for the PEEC method from which the partial elements can be identified.

Partial Element Equivalent Circuit for Conductors

[Ekman, 2003] gives more details concerning PEEC method for conductors. Here we will see only basic formulations.

In PEEC problems, there are two unknowns, the conduction current density $\vec{J}^C$, and the charge density q^F. To solve the system of equations the following partial computations have to be done:

1. Partial Inductances calculation (Lp)PEEC
2. Coefficients of Potential calculation (P)PEEC
3. Resistances calculation (R)PEEC
4. Combining (Lp)PEEC, (P)PEEC and (R)PEEC Models

Following paragraphs give some examples of used formulas and results.

Partial Inductances calculation (Lp)PEEC model for volume cell m connecting node i and j where Lp_{mm} is the partial self inductance for the volume cell and V_m^L accounts for the mutual inductance (magnetic field) coupling from other volume cells (see Figure 4.3):

$$L_{p_{\alpha\beta}} = \frac{\mu}{4\pi}\frac{1}{a_\alpha a_\beta}\int_{v_\alpha}\int_{v_\beta}\frac{1}{|\vec{r}_\alpha - \vec{r}_\beta|}dv_\alpha dv_\beta \tag{4.14}$$

for volume cell α and β.

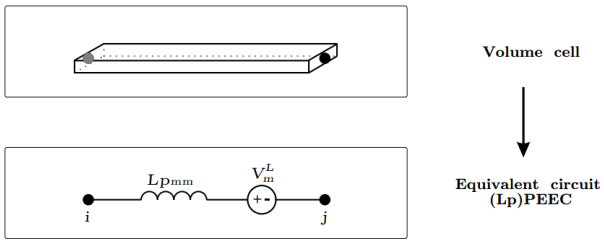


Figure 4.3: (Lp)PEEC

Coefficients of Potential calculation (P)PEEC model for one surface cell/node i where P_{ij} is the partial self coefficient of potential for the surface cell and V_iC accounts for the mutual capacitive (electric field) coupling from other surface cells (see Figure 4.4). A PEEC model only consisting of partial coefficients of potential is entitled a (P)PEEC model.

$$P_{ij} = \frac{1}{S_i S_j}\frac{1}{4\pi\epsilon_0}\int_{S_i}\int_{S_j}\frac{1}{|\vec{r}_i - \vec{r}_j|}\,dS_j\,dS_i \tag{4.15}$$

Resistances calculation A PEEC model only consisting of volume cell resistances is entitled a (R)PEEC model, see Figure 4.5

$$R_\gamma = \frac{l_\gamma}{a_\gamma \sigma_\gamma}. \tag{4.16}$$

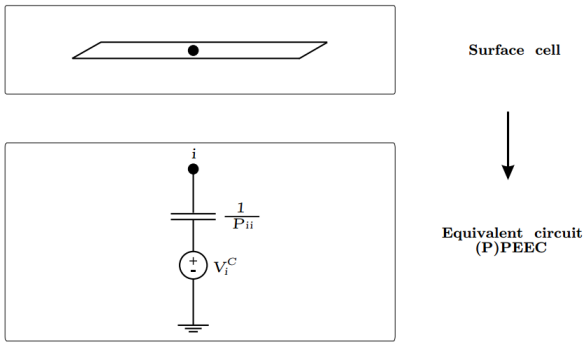


Figure 4.4: (P)PEEC.

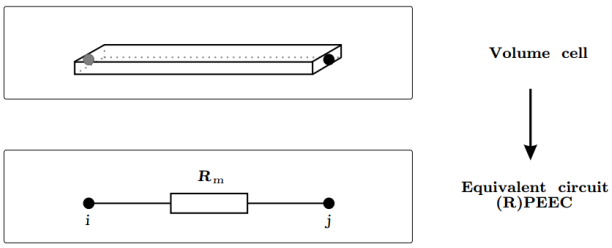


Figure 4.5: (R)PEEC

Combining models The inclusion of partial coefficients of potential results in a (Lp, R, P) PEEC model, Figure 4.6. In the figure one surface cell at each node is used to account for the capacitive coupling to corresponding node.

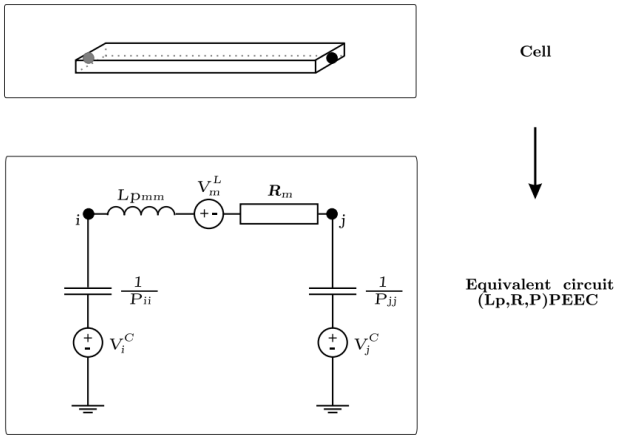


Figure 4.6: Four volume cells, separated by dashed lines, accounting for the current flowing in the direction of the arrows.

The rigorous full-wave version of the PEEC method is called (Lp, P, R, t)-PEEC, where Lp is partial inductance, P is potential coefficient (inverse of capacitance), R is resistance, and t is delay. If available, reduced model of the full-wave version can be used. For example, if the structure is electrically small, the delay term t can be omitted and the model can be reduced to (Lp, P, R)-

PEEC model. In addition, if frequency is sufficiently high so that $w * Lp >> R$, we can omit R term and use approximate (Lp, P)-PEEC model. According to various modeling situations, (Lp) and (Lp, R) models are also useful.

4.1.3 Topological PEEC approach. The limits of PEEC approach

We should underline that the original PEEC formulation is based on the definition of two orthogonal systems: first, magnetic vector and electric scalar potentials, $\vec{A}$ and ϕ respectively [Ruehli, 1974]. So, the classical PEEC approach is limited by a regular discretization. Thus, it is preferable to have a method that can deals with unstructured PEEC modeling. The first work in this direction have been done by *A. Rong et al.* [Rong and Cangellaris, 2001] in 2001 and two years later by E. Ruehli, the original author of PEEC method in [Ruehli et al., 2003]. In the first approach, they started from the basic PEEC algorithm, where a new formulation has been proposed to overcome the orthogonality constraint while keeping a structured subdivision of conductors. While in the second one, they tried to use a nonorthogonal formulation for arbitrary shapes. This two approaches are increasing significantly the complexity of the circuit because without orthogonality the use of unstructured meshes introduces a mutual coupling that need to be considered by introducing independent current sources.

In more recent works, *F. Freschi, M. Repetto et al.* propose to use a Cell-Method (CM) formulation for unstructured PEEC modeling. This method called "Dual-PEEC" [Freschi et al., 2006] because it keeps the duality properties of original approach is based on CM formulation and not on weak form.

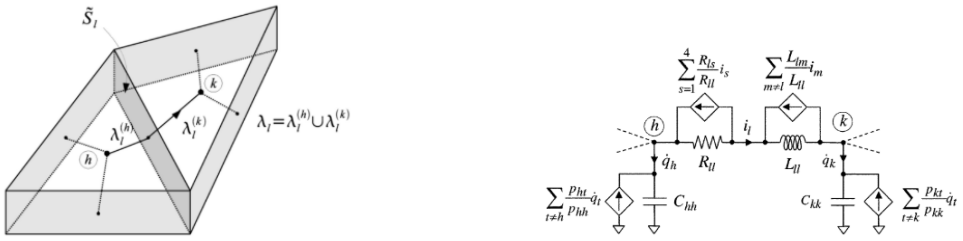
As we explained in chapter 2 and 3, the Cell-Method, as any cochain method, uses the fact that the formulation of the electromagnetic field can be expressed in terms of dual relations and how this duality can be exploited in its numerical solution. In fact, the CM formulation on the mesh can be already seen as a lumped model: CM is based on the definition of two orthogonal systems of inductive and capacitive cells. These cells are the framework on which a lumped parameter network is defined. Midpoints of capacitive cells are the nodes of the equivalent network (see Figure 4.7(a)).

Dual-PEEC, as classical PEEC, starts from integral equation (EFIE) inside a conductor:

$$\int_{\lambda_l} \frac{\vec{J}}{\sigma} \cdot d\vec{\lambda} + \frac{d}{dt} \int_{\lambda_l} \vec{A} \cdot d\vec{\lambda} + (\phi_j - \phi_h) = \int_{\lambda_l} \vec{E_0} \cdot d\vec{\lambda} \tag{4.17}$$

Here we used a vector notation for seeing the difference between Dual-PEEC and classical PEEC, where λ_l is the primal lth edge.

We will take as example the current density calculation to see the advantage of using Dual-PEEC with orthogonal mesh. When using an unstructured mesh for PEEC models based on week form, a local interpolation inside dual volumes linking the current density $\vec{J}$ in 4.17 to the global variable i is needed. So if we have a triangle [Freschi et al., 2006] (in the case of Figure 4.7(a)): a prism with δ thickness) we should calculate



(a) (b)

Figure 4.7: Dual-PEEC [Freschi et al., 2006]: (a) Unstructured discretization: consecutive triangular dual volumes. The couple primal edge (straight line)/dual face (dark gray) correspond to a two terminal circuit component. Light gray faces are related to the evaluation of mutual resistances; (b) Corresponding elementary equivalent circuit.

$$\vec{J} = \frac{1}{\delta}\sum_{m=1}^{3} i_m \vec{w_m} \tag{4.18}$$

where i_m is the current through the mth dual face, and $\vec{w_m}$ defined as

$$\vec{w_m} = (N_1 \nabla N_2 - N_2 \nabla N_1) \times \vec{n} \tag{4.19}$$

is nothing else that the facet shape function, being N_i the usual finite element ith nodal shape function, and $\vec{n}$ the unit vector orthogonal to the triangle [Freschi et al., 2006]. While using the Delaunay-Voronoi mesh, we can use an additivity property of 1-form, so the formulation (4.17) became just an algebraic summation:

$$J(x) = \frac{1}{\delta}\sum_{m=1}^{3} i_m \cdot x^m \tag{4.20}$$

where $J(x)$ is the current 2-form thought the dual faces of the cell which is independent of the integration path.

If we return to the differential forms notation, as already shown in chapter 2, we can see the similarity by comparing it with Tonti's diagrams. For example, Figure 4.8 shows this similarity if we consider a electrokinetic problem described as a components network (lumped model) or described with Maxwell equations as as field problem (distributed parameters system).

As we mentioned above, the mesh structure using in cochain method can be already interpreted in terms of network primal/dual graph, or, in over words, as lumped model.

Once the equivalent circuit of the whole system is built, the desired electrical responses are obtained via a network analysis program, usually Spice-like circuit simulators. However, this type of approach proves effective when the dimension of the problem is small, *i.e.* there are few cells and weak mutual couplings between them; in all other situations the solution has to be carried out through significantly reducing the precision of initial mesh or matrix reduction as POD, descried above.

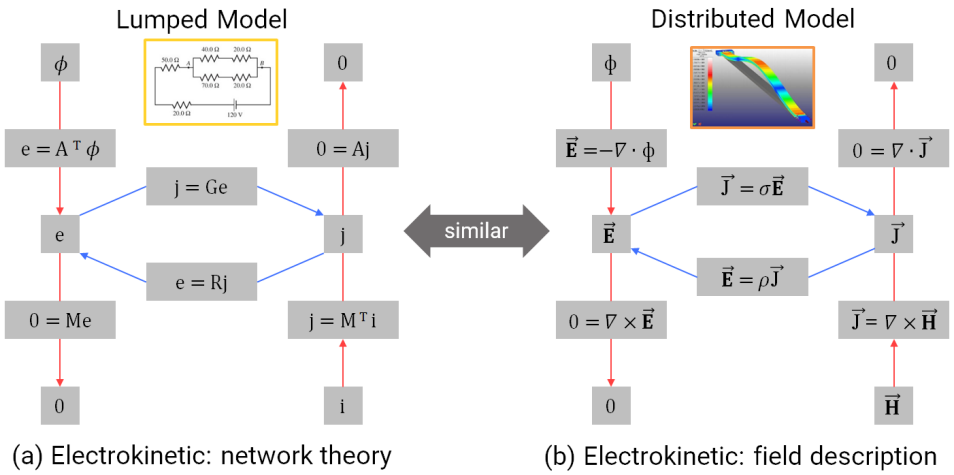


Figure 4.8: Tonti's diagram of electrokinetic for lumped and distributed parameters systems.

It should be noted that other recent work has addressed the problem of MOR in PEEC models, including [Nguyen et al., 2017] which uses an adaptive multi-point scheme.

4.1.4 Conclusion on model reduction methods.

Roughly, the main model order reduction methods are of two types: those that drastically reduce the number of degrees of freedom by projecting the dynamical system and the temporal response on a new "adapted" base. This is the case of POD methods. These methods are dedicated to dynamic systems and the focus is done on temporal responses. Nevertheless, they lose the topological properties of the initial model, *i.e.* the interconnection structure between their subsystems. Thus, the models obtained are not necessary the most appropriate to be use by designers because the physics, geometry, interconnection of subsystems and some properties, useful to system designers are lost. The second type of model reduction techniques are designed to keep the topology of the initial model. This is the case of PEEC methods. These reduced models are very useful for designer because the physical interconnections and geometry of the initial model are kept.

However, with the original PEEC method, the accuracy is rather low because the modeled geometry cannot be very complex. The dual PEEC method improves this point, nevertheless the reduction of DOF is in this case rather low. Some authors, as [Nguyen et al., 2017], propose promising methods to increase the reduction rate while keeping the precision, using adaptive multi-point scheme. In the following, we also propose a new alternative for this. Our method keep the philosophy of dual-PEEC methods but increase the reduction rate by using intermediate steps of data-analysis based on AI methods.

4.2 Review of Artificial Intelligence methods for data-analysis, classification, clustering and decision making

As noted in previous sections, one of the objectives of this thesis is to use Artificial Intelligence (AI) techniques to propose new model order reduction technique. It is then necessary to first conduct a review of the most current AI methods for data analysis, classification or clustering and decision making. Here we will see families of algorithms: Bayesian approach, Decision Tree Classifier, k-Nearest Neighbors, Neural Networks and Support-Vector Machines (Kernel approach). The last presented methods, the DBSCAN algorithms based methods, are an extension of Support Vector Machines techniques. They improve some of their characteristics and limits. The aim of this review is, for each method, to identify their strengths and weaknesses and to choose the optimal algorithms to work with model reduction. This review is mostly inspired by [Bishop, 2006].

What Is Machine Learning? Machine learning is a subfield of artificial intelligence (AI) concerned with algorithms that allow computers to learn. What this means, in most cases, is that an algorithm is given a set of data and infers information about the properties of the data – and that information allows it to make predictions about other data that it might see in the future. This is possible because almost all nonrandom data contains patterns, and these patterns allow the machine to generalize. In order to generalize, it trains a model with what it determines are the important aspects of the data [Segaran, 2007].

To understand how models come to be, consider a simple example in the otherwise complex field of pattern recognition. The field of pattern recognition is concerned with the automatic discovery of regularities in data through the use of computer algorithms and with the use of these regularities to take actions such as classifying the data into different categories [Bishop, 2006].

According to the definition, we could even say that simple correlation analysis and regression are both basic forms of machine learning.

Pattern recognition systems are in many cases trained from labeled "training" data (supervised learning), but when no labeled data are available other algorithms can be used to discover previously unknown patterns (unsupervised learning). Machine learning is strongly related to pattern recognition and originates from artificial intelligence. Pattern recognition focuses more on the signal and also takes acquisition and signal processing into consideration. It originated in engineering, and the term is popular in the context of computer vision. In pattern recognition, there may be a higher interest to formalize, explain and visualize the pattern, while machine learning traditionally focuses on maximizing the recognition rates. Yet, all of these domains have evolved substantially from their roots in artificial intelligence, engineering and statistics, and they've become increasingly similar by integrating developments and ideas from each other [Awad and Khanna, 2015].

There are many different machine-learning and pattern recognition algorithms, all with different strengths and suited to different types of problems. Some, such as decision trees, are transparent, so that an observer can totally understand the

reasoning process undertaken by the machine. Others, such as neural networks, are blackbox, meaning that they produce an answer, but it's often very difficult to reproduce the reasoning behind it [Bishop, 2006]. Many machine-learning algorithms are based heavily on mathematics and statistics.

Limits of Machine Learning. Machine learning is not without weaknesses. The algorithms vary in their ability to generalize over large sets of patterns, and a pattern that is unlike any seen by the algorithm before is quite likely to be misinterpreted. While humans have a vast amount of cultural knowledge and experience to draw upon, as well as a remarkable ability to recognize similar situations when making decisions about new information, machine-learning methods can only generalize based on the data that has already been seen, and even then in a very limited manner [Bishop, 2006].

Although they vary in their propensity for doing so, all machine-learning methods suffer from the possibility of overgeneralizing. As with most things in life, strong generalizations based on a few examples are rarely entirely accurate. As an example, we can get an indexing algorithms from Google: distinction between "Python" as a snake type, British comedy series "Monty Python" from the 1970s and an object-oriented, high-level programming language "Python". The classification algorithm is based on the appearance of words or phrases without any regard to what they mean or to sentence structures. Although it's theoretically possible to build an algorithm that would take grammar into account, this is rarely done in practice because the effort required would be disproportionately large (for scientific researcher but not for Google) compared to the improvement in the algorithm. Understanding the meaning of words or their relevance to a person's life would require far more information in their current incarnation, can access.

In the following sections, we will briefly describe AI-basic techniques and consider their strengths and weaknesses.

4.2.1 Bayesian approach

We will take as an example the naïve Bayes (NB) classifier algorithm as it is basic for all Bayesian approach algorithms. The NB classifier algorithm is a simple Machine Learning algorithm that was created for use in text classification, an area of ML where it can still be competitive with more advanced general-purpose algorithms. The name stems from the fact that the Bayes formula is applied to the data with very "naïve" assumptions about independence.

This assumption is what usually makes the algorithm less useful for general (dense) problems, because the features are rarely anywhere near independent. For sparse text features, this assumption still isn't true, but it's true enough for the algorithm to work surprisingly well in practice.

Our goal is to classify a review by finding the probability $p(C_k|x)$ of the review sentiment being "bad" ($k = 0$) or "good" ($k = 1$) based on the features x of the instance. In probability theory using the Bayes formula, this can be written like:

$$p(C_k|x) \sim \ p(C_k)p(x|C_k) \tag{4.21}$$

The part $p(x|C_k)$ is known as the joint probability of the features x if the instance was of class C_k. Because of the independence assumption (the naïve part), there's no cross-feature probability, and this becomes simply the product of the probability of each of the features given the class:

$$p(C_k|x)\ p(C_k)p(x_1|C_k)p(C_k)p(x_2|C_k)...p(C_k)p(x_i|C_k) = p(C_k)\prod_i^N p(x_i|C_k) \tag{4.22}$$

Because $p(C_k)$ is the marginal class distribution – the overall breakdown of good and bad sentiment reviews – which can be easily find from the data, it only needs to figure out what $p(x_i|C_k)$ is. We can read this expression as "the probability of a specific feature for a specific class". For example, you would expect the probability of having the word great in a good-sentiment review being higher than in a bad-sentiment review.

We can imagine learning this from the data by counting the feature (word) presence across all documents in each class. The probability distribution that generates such counts is called the multinomial distribution, and $p(C_k)p(x_i|C_k)$ becomes:

$$p(x_i|C_k)\ \prod_i p_{k_i}^{x_i} \tag{4.23}$$

If we use this in the previous equation and move to log space for convenience

$$\log\left[p(C_k|x_i)\right]\log\left[p(C_k)\right]\prod_i p_{k_i}^{x_i} = \log\left[p(C_k)\right] + \sum_i^n x_i \log\left(p_{k_i}\right) = b + w_k x \tag{4.24}$$

Here b is $log[p(C_k)]$ (known from the data), x represents the features of the instance wanted to predict, and w_k is $\log(p_{k_i})$ – the fraction of times a word appears in a good or bad document, which is learn at model build time. Please note that we have left out various constants throughout this calculation, and there are multiple implementation details to consider when coding this algorithm from scratch, but the basics outlined here remain true.

Strengths and Weaknesses Perhaps the biggest advantage of naïve Bayesian classifiers over other methods is the speed at which they can be trained and queried with large datasets. Even with a huge training set, there are usually a relatively small number of features for each item, and training and classifying items is just a mathematical manipulation of the probabilities of these features.

This is particularly true when training is incremental – each new piece of training data can be used to update the probabilities without using any of the old training data. This support for incremental training is very important for an application like a spam filter, which is constantly trained on new email messages that come in, has to be updated quickly, and may not even have access to all the email messages that have been received.

Another big advantage of naïve Bayesian classifiers is the relative simplicity of interpreting what the classifier has actually learned. Because the probabilities

of each feature are stored, we can look at database at any time and see which features are best at dividing spam and nonspam. This information is interesting to look at, and it can potentially be used for other applications or as a starting point for other applications.

The biggest downside to naïve Bayesian classifiers is their inability to deal with outcomes that change based on combinations of features. Imagine the following scenario in which we are trying to distinguish spam from nonspam email: let's say the job is building web applications, so the word "online" frequently appears in work-related email. The best friend works at a pharmacy and likes sending us funny stories about things that happen to him at work. Also, like most people who haven't closely guarded their email addresses, we occasionally receive spam containing the words "online pharmacy" [Bishop, 2006].

Here, we can see some dilemma – the classifier is constantly being told that "online" and "pharmacy" exist in nonspam email messages, so their probabilities become higher for nonspam. When we tell the classifier that a certain email message with the words "online pharmacy" is spam, those words are adjusted slightly more toward spam, creating a constant battle. Since features are all given probabilities separately, the classifier can never learn about combinations. In document classification this is usually not a big deal, since an email message with the words "online pharmacy" probably contains other spam indicators, but in other problems, understanding feature combinations can be much more important [Segaran, 2007].

4.2.2 Neural Networks

While there are many different kinds of neural networks, they all consist of a set of nodes (the neurons) and connections between them. The most of networks used in science is called a multilayer perceptron (MLP) network. This type of network consists of multiple layers of neurons, the first of which takes the input – in this case, the words entered by the user. The last layer gives the output.

To ask the neural network to get the best results for a query, the input nodes for the words in that query have their values set to 1. The outputs of those nodes are turned on and they attempt to activate the hidden layer. In turn, the nodes in the hidden layer that get a strong enough input will turn on their outputs and try to activate nodes in the output layer [Daumé III, 2012].

The network shown in the Figure 4.9 has one layer of neurons. The layers of neurons are connected to each other by synapses, which each have an associated weight. The outputs from one set of neurons are fed to the next layer through the synapses. The higher the weight of a synapse leading from one neuron to the next, the more influence it will have on the output of that neuron [Bishop, 2006].

Prediction with a neural network is a straightforward generalization of prediction with a perceptron. First we compute activations of the nodes in the hidden unit based on the inputs and the input weights. Then we compute activations of the output unit given the hidden unit activations and the second layer of weights [Daumé III, 2012].

The only major difference between this computation and the perceptron computation is that the hidden units compute a non-linear function of their inputs. This is usually called the activation function or link function. More formally,

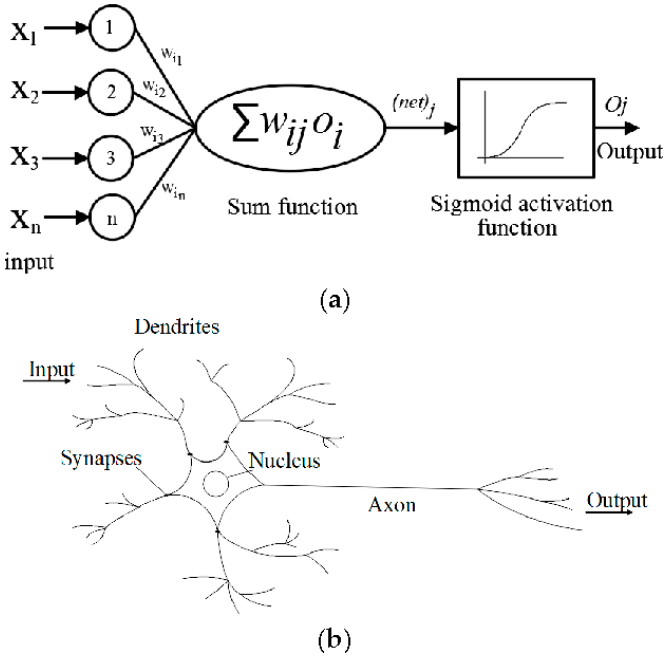


Figure 4.9: Example of one layer neural network.

if $w_{i,d}$ is the weights on the edge connecting input d to hidden unit i then the activation of hidden unit i is computed as:

$$h_i = f(w_i \cdot x) \tag{4.25}$$

Where f is the link function and w_i refers to the vector of weights feeding into node i.

A more popular link function is one of the so called "sigmoid" (because it looks like an "S", the Greek character "Sigma") functions. Some examples are hyperbolic tangent tanh-*function*. For more sigmoid function see Figure 4.10.

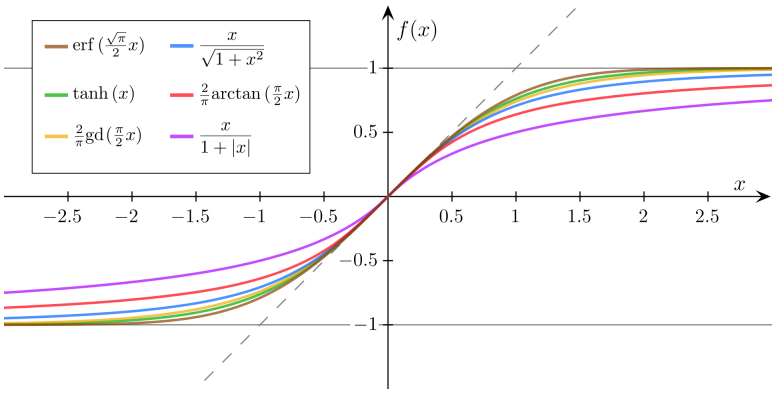


Figure 4.10: Different sigmoid functions.

Assuming for now that we are using tanh as the link function, the overall prediction made by a two-layer network can be computed using:

$$\hat{y} = \sum_i v_i \tanh f(w_i \cdot \hat{x}) = v \cdot \tanh(W\hat{x}) \tag{4.26}$$

For two layer Network, we can use the Algorithm 3.

Algorithm 3 Two Layer Network Predict

```
1: function C(W, v, x̂)
2:     for i = 1 to numberofhiddenunits do
3:         h_i ← tanh(w_i · x̂)              ▷ compute activation of hidden unit i
4:     end for
5:     return v · h                          ▷ compute output unit
6: end function
```

When working with two-layer networks, the key question is: how many hidden units should I have? If the data is D dimensional and we have K hidden units, then the total number of parameters is $(D+2)K$. (The first +1 is from the bias, the second is from the second layer of weights.) Following on from the heuristic that we should have one to two examples for each parameter we are trying to estimate, this suggests a method for choosing the number of hidden units as roughly $[\frac{N}{D}]$. In other words, if we have tons and tons of examples, we can safely have lots of hidden units. If we only have a few examples, we need restrict the number of hidden units in the network.

Training a Neural Network In the example above, we supposed that the neural network already has the appropriate weights for all the synapses. The real power of neural networks is that they can start with random weights and then learn from examples through training. The most common method of training a multi-layer perceptron network is called backpropagation.

When training a network, we always know the desired output of each node in the output layer. In this case, it should be pushed toward 1 if the user clicked on that result, and pushed toward 0 if he did not. The only way to change the output of a node is to change the total input to that node [Bishop, 2006].

To determine how much the total input should be changed, the training algorithm has to know the slope of the tanh function at its current level of output. In the middle of the function, when the output is 0.0, the slope is very steep, so changing the input by only a small amount gives a big change. As the outputs get closer to –1 or 1, changing the input has a smaller effect on the output. The slope of the function for any output value is specified by this function, which we can add to the start of $d\tanh(y) = 1 - y^2$.

Before running the backpropagation method, it's necessary to run feedforward so that the current output of every node will be stored in the instance variables. The backpropagation algorithm then performs the following steps:

For each node in the output layer:

1. Calculate the difference between the node's current output and what it should be.

2. Use the $d\tanh$ function to determine how much the node's total input has to change.

3. Change the strength of every incoming link in proportion to the link's current strength and the learning rate.

For each node in the hidden layer:

1. Change the output of the node by the sum of the strength of each output link multiplied by how much its target node has to change;
2. Use the *dtanh* function to determine how much the node's total input has to change;
3. Change the strength of every input link in proportion to the link's current strength and the learning rate.

The implementation of this algorithm actually calculates all the errors in advance and then adjusts the weights, because all the calculations rely on knowing the current weights rather than the updated weights.

Strengths and Weaknesses The main strength of neural networks is that they can handle complex nonlinear functions and discover dependencies between different inputs. Although the example only showed numerical inputs of 1 or 0 (present or absent), any number can be used as an input, and the network can also estimate numbers as outputs.

Neural networks also allow for incremental training and generally don't require a lot of space to store the trained models, since they are just a list of numbers representing the synapse weights. There is no need to keep the original data following training, which means that neural networks can be used for applications in which there is a continuous stream of training data.

The major downside of neural networks is that they are a black box method. The example network shown here was contrived to be extremely simple to follow, but in reality, a network might have hundreds of nodes and thousands of synapses, making it impossible to determine how the network came up with the answer that it did. Not being able to understand the reasoning process may be a deal breaker for some applications [Bishop, 2006].

Another downside is that there are no definitive rules for choosing the training rate and network size for a particular problem. This decision usually requires a good amount of experimentation. Choosing a training rate that's too high means that the network might overgeneralize on noisy data, and choosing one that's too low means it might never learn, given the data we have.

4.2.3 Decision Tree Classifier

Unlike most other classifiers, the models produced by decision trees are easy to interpret – the list of numbers in a Bayesian classifier will tell us how important each word is, but we really have to do the calculation to know what the outcome will be. A neural network is even more difficult to interpret, since the weight of the connection between two neurons has very little meaning on its own. We can understand the reasoning process of a decision tree just by looking at it, and we can even convert it to a simple series of if-then statements.

Decision trees are one of the simpler machine-learning methods. They are a completely transparent method of classifying observations, which, after training, look like a series of if-then statements arranged into a tree. Figure 4.11 shows an example of a decision tree for classifying fruit.

It should be clear from the figure what a decision tree does when faced with the task of classifying a new item. Beginning at the node at the top of the tree,

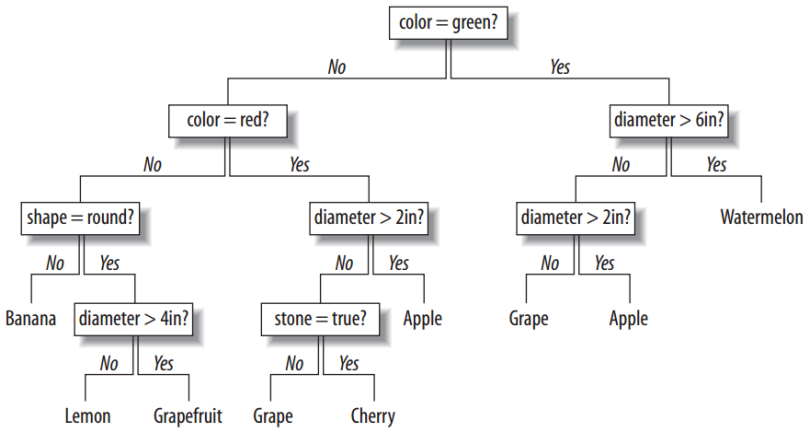


Figure 4.11: Example decision tree [Bishop, 2006].

it checks the item against the node's criteria – if the item matches the criteria, it follows the Yes branch; otherwise, it follows the No branch. This process is repeated until an endpoint is reached, which is the predicted category [Bishop, 2006].

Training Classifying in a decision tree is quite simple; training it is trickier. The algorithm described built the tree from the top, choosing an attribute at each step that would divide the data in the best possible manner. To illustrate this, consider the fruit dataset shown in Figure 4.12. This will be referred to as the original set.

Diameter	Color	Fruit
4	Red	Apple
4	Green	Apple
1	Red	Cherry
1	Green	Grape
5	Red	Apple

Figure 4.12: Example decision tree [Bishop, 2006].

There are two possible variables on which this data can be divided, either Diameter or Color, to create the top node of the tree. The first step is to try each of them in order to decide which of these variables divides the data best. Dividing the set on Color gives the results shown in Figure 4.13.

Red	Green
Apple	Apple
Cherry	Grape
Apple	

Figure 4.13: Fruit data divided by Color [Bishop, 2006].

The data is still pretty mixed. However, if we divide the dataset by Diameter (less than four inches and greater than or equal to four inches), the results divide much more cleanly (referred to as Subset 1 on the left and Subset 2 on the right). This division is shown in Figure 4.14.

Diameter < 4in	Diameter ≥ 4in
Cherry	Apple
Grape	Apple

Figure 4.14: Fruit data divided by Diameter [Bishop, 2006].

This is obviously a much better result, since Subset 1 contains all the Apple entries from the original set. Although the better variable is clear in this example, larger datasets will not always have such clean divisions. In additional, we can introduced the concept of entropy (the amount of disorder in a set) to measure how good a division is:

- p(i) = frequency(outcome) = count(outcome) / count(total rows)
- Entropy = sum of p(i)·log(p(i)) for all outcomes

A low entropy within a set tells us that the set is mostly homogeneous, and a value of 0 means that it consists of entirely one type of item. Subset 1 (diameter $\geq$ 4) in Figure 4.14 has an entropy of 0. The entropy for each set is used to calculate the information gain, which is defined as:

- weight1 = size of subset1 / size of original set
- weight2 = size of subset2 / size of original set
- gain = entropy(original) – weight1·entropy(set1) – weight2·entropy(set2)

So for each possible division, the information gain is calculated and used to determine the dividing variable. Once the dividing variable has been chosen, the first node can be created, as shown in Figure 4.15.

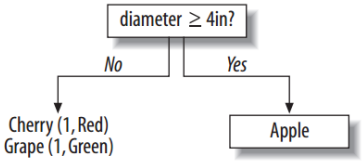


Figure 4.15: Root node of the fruit decision tree [Bishop, 2006].

The criteria is shown on the node, the data that doesn't pass the criteria gets pushed down the No branch, and the data that meets or passes the criteria is pushed down the Yes branch. Since the Yes branch now has just one possible outcome, it becomes an endpoint. The No branch still has a mixture, so it can be divided further using exactly the same method that was used to choose the top node. In this case, color is the best variable on which to divide the data. This process repeats until there is no information gain from dividing up the data on a given branch.

Strengths and Weaknesses The most striking advantage of decision trees is how easy it is to interpret a trained model and how well the algorithm brings important factors to the top of the tree. This means that a decision tree is useful not just for classification, but also for interpretation. Like the Bayesian classifier, we can look under the hood and understand why it works the way it does, and this can help we make decisions outside the classification process. For example, the model predicted which users would become paying customers, and having a decision tree that shows which variables are best at cleanly dividing the data could be useful for planning an advertising strategy or deciding what other data should be collected [Bishop, 2006].

Decision trees also work with numerical data as inputs, since they find the dividing line that maximizes information gain. The ability to mix categorical and numerical data is useful for many classes of problems – something that traditional statistical methods like regression have trouble doing. On the other hand, decision trees are not as good at making predictions for numerical results. A regression tree can divide the data into mean values with the lowest variance, but if the data is complex, the tree will have to get very large before it is able to make accurate decisions.

The main advantage that decision trees have over the Bayesian classifier is that they can easily cope with interactions of variables. A spam filter built using a decision tree would easily determine that "online" and "pharmacy" are fine in isolation but that when they're together they indicate spam. Unfortunately, using the most basic algorithm is not practical for a spam filter for the simple reason that it does not support incremental training. (Alternative algorithms for decision trees that support incremental training are an active area of research). We can take a big set of documents and build a decision tree for spam filtering, but we can't train it on individual new email messages as they come in – we would have to start from scratch each time. Since many people have tens of thousands of email messages, this would be impractical to do each time. Also, since the number of possible nodes is very large (each feature is present or absent), the trees can become extremely large and complex and would be slow to make classifications.

4.2.4 K-Means clustering

K-means clustering is a method of clustering. Originally, it comes from signal processing, that is popular for cluster analysis in data mining. K-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This type of algorithm is quite different from hierarchical clustering or decision tree classifier because it is told in advance how many distinct clusters to generate. The algorithm will determine the size of the clusters based on the structure of the data.

To illustrate the algorithm, we took an example from [Bishop, 2006].

K-means clustering begins with k randomly placed centroids and assigns every item to the nearest one. After the assignment, the centroids are moved to the average location of all the nodes assigned to them, and the assignments are redone. This process repeats until the assignments stop changing.

The Figure 4.16 shows the process in action for five items and two clusters.

In the first frame, the two centroids (shown as dark circles) are placed randomly. The second frame shows that each of the items is assigned to the nearest centroid – in this case, A and B are assigned to the top centroid and C, D, and E are assigned to the bottom centroid. In the third frame, each centroid has been moved to the average location of the items that were assigned to it. When the assignments are calculated again, it turns out that C is now closer to the top centroid, while D and E remain closest to the bottom one. Thus, the final result is reached with A, B, and C in one cluster, and D and E in the other.

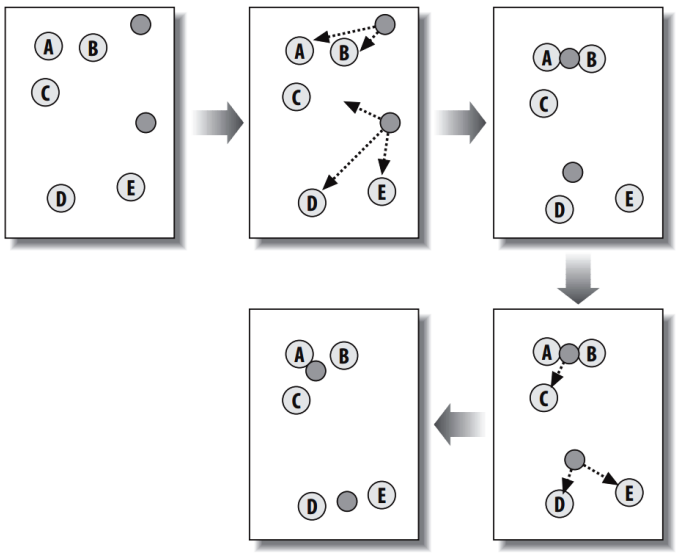


Figure 4.16: K-means clustering with two clusters [Bishop, 2006].

Strengths and Weaknesses K-Means is one of the few algorithms that will make numerical predictions in complex functions and still remain easy to interpret. The reasoning process is easy to understand, and a simple change to the code will allow us to see exactly which neighbors are being used in the calculation. Neural networks can also make numerical predictions in complex functions, but they will certainly not be able to show us similar examples to help us understand the reasoning process.

Furthermore, the process of determining the correct amounts to scale the data not only improves predictions, but also tells us which variables are important in making predictions. Any variable that gets scaled down to 0 can be thrown out. In some cases, that data may have been difficult or expensive to collect.

The major weakness of K-Means is that it requires all the training data to be present in order to make predictions. In a dataset with millions of examples, this is not just a space issue but also a time issue-every item for which we are trying to make a prediction has to be compared with every other item to determine which are the closest. This process may be too slow for some applications.

Another disadvantage is that finding the correct scaling factors can be difficult. If there are many different variables to try, it might be necessary to try millions of different scaling factors before finding the right one [Bishop, 2006].

4.2.5 Support Vector Machine

A support-vector machine (SVM) builds a predictive model by finding the dividing line between two categories. If one plots a set of values for height versus speed and the best position for each person playing basketball, it gets a graph like the one shown in Figure 4.17. Front-court players are shown as crosses × and back-court players are shown as • on the Figure 4.17. Also shown on the graph are a few lines that separate the data into the two categories [Bishop, 2006].

A support-vector machine finds the line that separates the data most cleanly, which means that it is the greatest possible distance from points near the dividing line. In Figure 4.17, although all the different lines separate the data, the one that does this best is the one labeled "Best". The only points necessary to determine where the line should be are the points closest to it, and these are known as the support vectors.

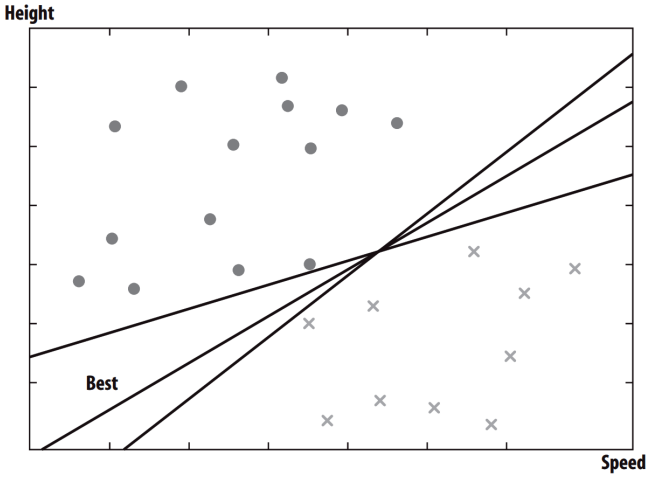


Figure 4.17: Plot of basketball players and dividing lines [Bishop, 2006].

After the dividing line has been found, classifying new items is just a matter of plotting them on the graph and seeing on which side of the line they fall. There's no need to go through the training data to classify new points once the line has been found, so classification is very fast.

The Kernel Trick Support-vector machines, along with other linear classifiers that use vector dot products, often take advantage of a technique called the *kernel trick*. To understand this, consider how the problem would change if the classification we was trying to predict was not position, but rather, whether the players would be appropriate for an amateur team in which the positions are often switched around. This is more interesting because the division is not linear. We don't want players who are too tall or too fast because they would make the game too difficult for others, but we don't want them to be too short or too slow either. Figure 4.18 shows what this might look like, where a bullet • indicates that a player is appropriate for the team and a cross × indicates that he isn't.

There is no straight dividing line here that will work, so we can't use a linear classifier to find the division without first altering the data in some way. One way to do this would be to transform the data into a different space – perhaps a space with more than two dimensions – by applying different functions to the axis

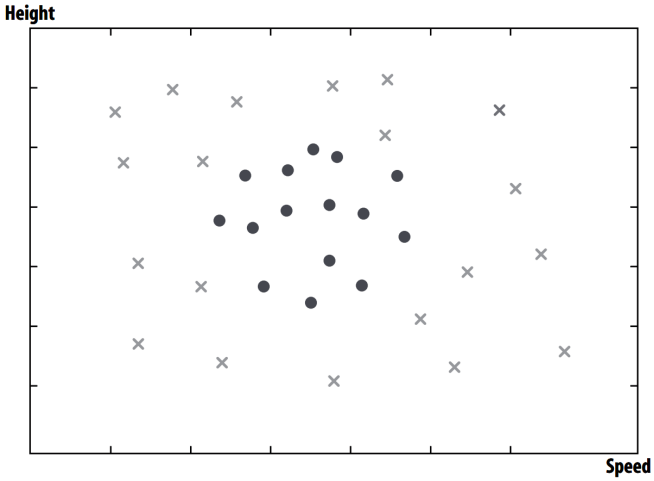


Figure 4.18: Plot of basketball players for amateur team [Bishop, 2006].

variables. In this case, we might create a new space by subtracting the average values for height and speed and squaring the height and speed values. This would look like Figure 4.19

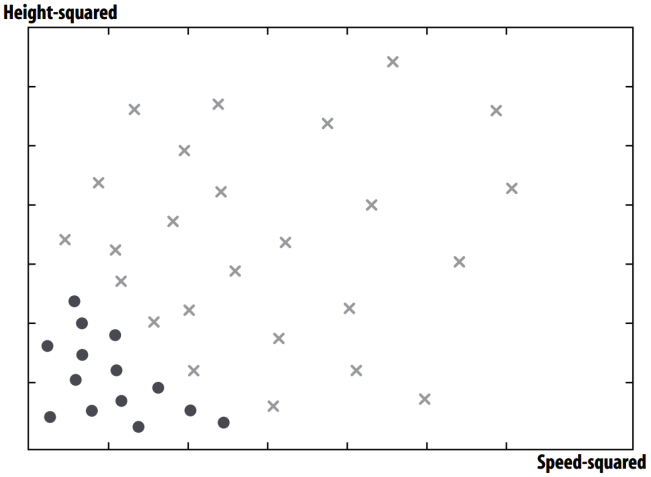


Figure 4.19: Basketball players in polynomial space [Bishop, 2006].

This is called a *polynomial transformation*, and it transforms data on different axes. It's now easy to see that there is a dividing line between the members that are appropriate and inappropriate for the team, which can be found with a linear classifier. Classifying new points would be a matter of transforming them into this space and seeing on which side of the line they fall.

The transformation works in this case, but in many examples, finding the dividing line will require transformation into much more complex spaces. Some of these spaces have thousands or even infinite dimensions, so it's not always practical to actually do this transformation. This is where the kernel trick comes in – rather than transforming the space, it replaces the dot-product function with a function that returns what the dot-product would be if the data was transformed into a different space. The dot-product could be altered by replacing the dot-product function with other functions for combining vectors, which allowed it to solve nonlinear problems [Bishop, 2006].

Strengths and Weaknesses Support-vector machines are a very powerful classifier; once we get the parameters correct, they will likely work as well as or better than any other classification method [Bishop, 2006]. Further, after training they are very fast to classify new observations, since classification is simply done by determining on which side of a dividing line a point is. By transforming categorical inputs to numbers, it can make them work with a mixture of categorical and numerical data.

One disadvantage is that the best kernel transformation function and the parameters for that function will be slightly different for every dataset, and we'll have to find them each time. Looping through possible values helps to alleviate this problem, but it does require that we have a big enough dataset to do reliable cross-validation.

Generally, SVMs are much more suited to problems in which there is a lot of data available, while other methods such as decision trees can still give interesting information with very small datasets.

Like neural networks, SVMs are a black box technique – it's actually even more difficult to interpret how SVM is doing classification because of the transformation into high-dimensional spaces. SVM may give great answers, but we'll never really know why [Bishop, 2006]. However, the last weakness is resolved in the DBScan algorithm.

4.2.6 Basic DBSCAN algorithm

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [Ester et al., 1996] is one of the most popular and widely used clustering algorithms because of its generally good performance in different scenarios according to [Chio and Freeman, 2018]. Unlike with k-means and SVM, the number of clusters is not operator-defined but instead inferred from the data. Unlike hierarchical clustering (ex. Decision Tree Classifier), which is *distance*-based, DBSCAN is a *density*-based algorithm that divides datasets up into subgroups of high-density regions [Chio and Freeman, 2018].

Terminology and description Let's consider some of the terminology introduced by this algorithm:

In DBSCAN, clustering happens based on two important parameters:

1. ϵ defines the radius around a certain point within which to search for neighbors,

2. *minPoints* is the minimum number of points required to form a cluster.

Each data point is classified as a core point, a border point, or a noise point:

1. *Core points* are points that have at least *minPoints* number of points within their ϵ-radius,

2. *Border points* are themselves not core points, but are covered within the ϵ-radius of one of core point,

3. *Noise points* are neither core points nor border points.

In naive implementations, this classification step is done by iterating through each point in the dataset, calculating its distance to all other points in the dataset, and then associating each point with its neighbors (points that are closer than ϵ distance away from it). With this information, it can tag all points as either core, border, or noise points. After classifying all the data points in the dataset as one of these three types of points, the DBSCAN algorithm proceeds as follows:

1. Select a point P at random out of all unvisited points;
2. If P is not a core point, mark it as *visited* and continue;
3. If P is a core point, form a cluster around it, and recursively find and usurp all other points within the ϵ-radius of P as well as any other point that is in the ϵ-radius of all core points usurped by this cluster.

   Let's say that there exists a core point Q within the ϵ-radius of P. Q (along with all its border points) will be added to the cluster formed around P. If there is another core point R within the ϵ-radius of Q, core point R (along with all its border points) will also be added to the cluster formed around P.
4. Repeat until all points in the dataset have been visited.

This definition becomes more intuitive if we have a look on the figure 4.20. If we set $minPoints$=5, we can see that we need different radius to enclose 5 neighboring points: point p is directly density-reachable from a point q if p in ϵ-zone and the number of neighboring points in ϵ-zone is greater than $minPoints$. If the $minPoints$=5, you can see that we need different radius to enclose 5 neighboring points. Setting ϵ helps to clearly define metric. Also, we can see, that higher $minPoints$ or lower ϵ indicate higher density necessary to set up a cluster.

To determine a cluster, DBSCAN starts with an arbitrary point p and retrieves all points in the ϵ-zone (a circle around a point in Euclidean metric). If the total number of points in the ϵ-zone is less than $minPoints$, than this point is considered as a "border" point and the algorithms passes to another arbitrary point p. Else, p is considered as a core point. In this case, all neighboring points of p will form a core of a cluster. Finally, a cluster will be the union of p with his neighboring points and also of neighboring of neighboring points. The algorithm stops when all points are visited.

At first glance, we have some random side in algorithm: the choice of starting point p and the sequencing in points choice. Nevertheless, the result of DBSCAN is stable thanks to the chosen metric. In fact, this density-reachable metric is not symmetric in general case. Figure 4.21 depicts the relation of some sample points. Although not symmetric in general, it is obvious that it is symmetric for core points. Therefore, the distance between core points remain the same and so the DBSCAN is deterministic, always generating the same clusters. In addition, this metric ensure that clusters present always simply connected space shape (no hole exists in clusters).

Strengths and Weaknesses The following highlights about DBSCAN clustering are extracted from the book [Chio and Freeman, 2018]:

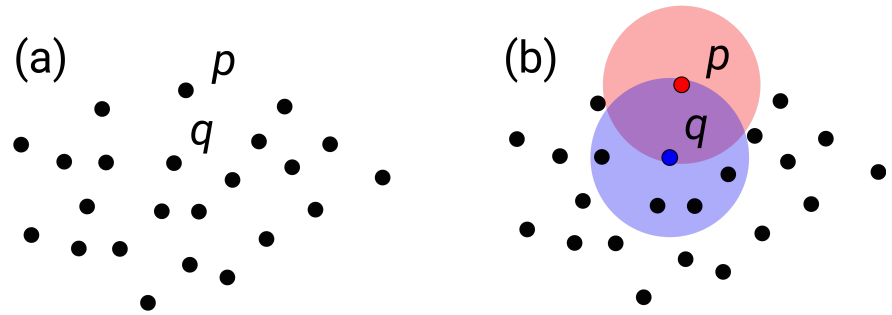


Figure 4.20: Core points and border points [Ester et al., 1996].

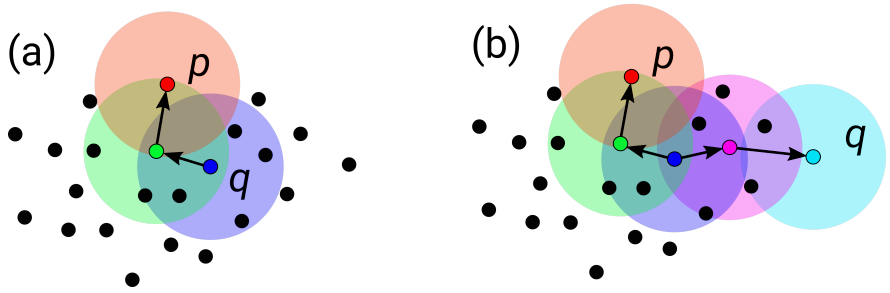


Figure 4.21: One step of DBSCAN [Ester et al., 1996].

1. DBSCAN does not require the user to set the number of clusters *a priori*.

2. DBSCAN can find any shape of clusters. The cluster doesn't have to be circular.

3. DBSCAN can identify outliers (noise).

4. DBSCAN has stable clusters: if we run the algorithm twice or more with a different random initialization, we should expect to get roughly the same clusters back, it sample shouldn't radically change the resulting cluster structure. If we vary the clustering algorithm parameters you want the clustering to change in a somewhat stable predictable fashion.

5. DBSCAN can scale to large data sizes.

The weaknesses of DBSCAN algorithm are:

1. DBSCAN does not work well when clusters in the dataset have different densities. This makes it difficult to select values of ϵ and $minPoints$minPoints that are suitable for all clusters in the data. Ordering Points to Identify the Clustering Structure (OPTICS) is an algorithm very similar to DBSCAN that addresses this weakness by introducing spatial ordering into the point selection process, albeit at the cost of speed.

2. DBSCAN performs poorly on high-dimensional data because it most commonly uses the Euclidean distance as a distance metric, which suffers from the "curse of dimensionality".

4.2.7 Conclusion on AI-based methods relevant for MOR applications.

As we explained in the first chapter of this thesis, we want to automatically generate a lumped model (network of discrete components) from a model with distributed parameters (field distribution in a continuous space, possibly meshed). This task will correspond to our model order reduction step.

As it is evident from this review of AI-based methods for data analysis, this task is in fact a clustering task based on the field information calculated from the development presented in the chapter 3.

Among all the clustering techniques presented in this review, it is now necessary to choose the most appropriate one for our purpose. In order to support this choice, we must now propose relevant and as objective as possible criteria.

It can sometimes be difficult to make sense of the results of clustering operations. Evaluating supervised learning algorithms is a much more straightforward task because we have access to the ground truth labels: we can simply count the number of samples to which the algorithm correctly and wrongly assigns labels [Chio and Freeman, 2018]. In the case of unsupervised learning, it is unlikely. We have no information in literature about evaluation criteria suitable for generating a lumped model, so we decided to introduce our own assumptions for choosing one clustering algorithm:

1. We have no any reliable expert information about quality of clustering results. The clustering need to be suitable for any electromechanical device and for any purpose in terms of future use of this lumped parameter model (example, for optimization or for design). In some literature, this property is called *scalability*,

2. Clusters need to be "physical": we want that each cluster becomes a lumped component into a complex interconnected network. So, it should be a closed compact convex sets of points with nearest values of physical variable with non-empty interior,

3. For better solution, we need to use different type of data about previous field computations (different types of physical variables: for example, conductivity, electric or magnetic fields, thermal properties, *etc*).

Besides, for human beings it seems evident to see a clusters. But computer need to have a step of image recognition. Depending on the algorithm selected, the process of recognition by the clusters may be identified in a different ways (see figure 4.22).

Based on above-mentioned assumption we will propose our own criteria for choosing a clustering algorithm dedicated to model order reduction.

Finally, we can say that in data science there are 2 big groups of algorithms for clustering: supervised and unsupervised. For our case, we choose unsupervised clustering. First of all, it doesn't need any specific *a priori* information (called expert information) for training. In fact, this information is not available for all existing electromechanical devices. Besides, this type of expertise is subjective: two experts can produce different classifications. Secondly, most of supervised algorithms present overfitting tendency: they produce an analysis results too

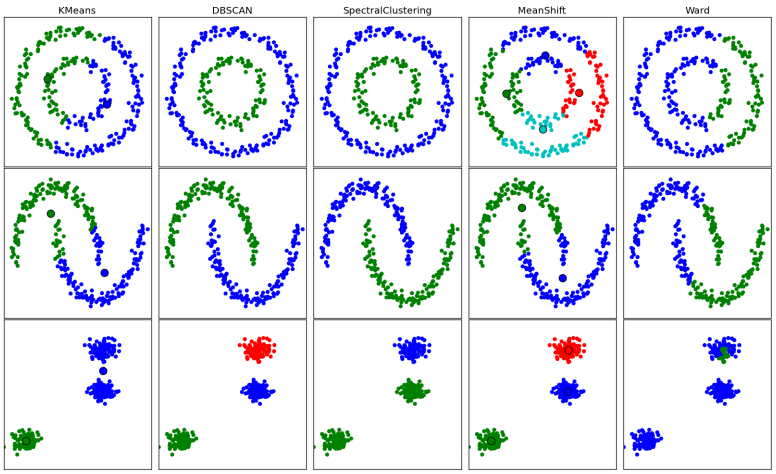


Figure 4.22: Comparing different clustering algorithms on toy datasets [Tutorial Sci-Kit, 2012].

close to a particular set of data, and therefore this result may fail to include and fit additional data or to predict reliably future observations. Finally, supervised algorithms require much more computational time (especially for training) and this is one of their main limits in terms of scalability.

Among existing unsupervised methods we choose algorithms that have very large scalability, intuitive parameters (like number of clusters or size of samples) and manages easily spatial distribution (all clusters must be convex hull). Using above-mentioned comparative characteristics, 2 methods can be selected: K-Means and DBSCAN. We compared these two methods in preliminary test before a choice was made. Finally, the DBSCAN algorithm was chosen because it does not require an *a-priori* specification of the number of clusters in the data unlike K-Means algorithm. Moreover DBSCAN includes noise reduction capabilities and is more robust to special cases allowing an improved clustering in more general contexts.

4.3 Automatic generation of lumped models

In this section, we will describe the principal stages of automatic generation of lumped (LP) model starting from Cochain Method calculation that is described in the third chapter. This correspond to lower part of our software structure of the Figure 4.1. This lower part is shown on Figure 4.23.

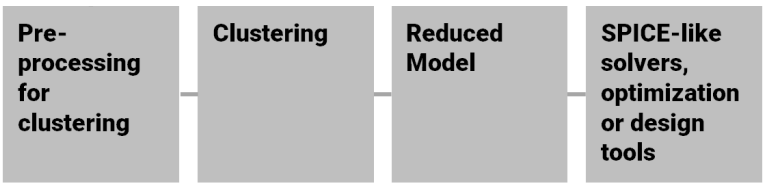


Figure 4.23: Lower part of software structure (see Figure 4.1).

To sum up, the automatic generation of a lumped model is decomposed in four steps:

1. Construction of a spatial mesh of the physical device (2D or 3D) based on its geometry and material description.

2. Solving the physical equations associated via the cochain method.

This two first steps were described in the chapter 3. Therefore this section focus mainly on the next two steps, even if the two first steps are presented for our application example:

3. Identification via data analysis of homogeneous spatial distribution of fields using DBSCAN algorithm. A final homogeneous spatial region corresponds to one cluster. This step requires preliminary data processing tasks for computing densities of fields (local spatial distribution) needed by DBSCAN algorithms.

4. The identified homogeneous regions (clusters) become the preliminary information for generating the discrete components associated with the LP network.

The final network of discrete components at the end of this fourth step is a reduced version of the model obtained at the end of the second step. As already mentioned in the introduction of the manuscript, the MBSE approach requires different models at different levels. For "detailed" levels, the FEM-like model approach is used (precise fine model of step 2): it is accurate but time-consuming. In an optimization procedure or if it is necessary to take into account the time evolution (transient analysis), a large number of model evaluations are required, which is the bottleneck of the FEM analysis. Network models become, in this case, more relevant (reduced coarse model of step 4).

The third step, proposed in this work, seems original because the literature review did not identify previous scientific work that used clustering through the DBSCAN algorithm in the context of model reduction as proposed here.

The step 4 for generating lumped components uses an approach similar to the Dual-PEEC method [Freschi and Repetto, 2008], [Alotto et al., 2011] but compared to this one, the method proposed in this thesis is conducted after the clustering step and therefore, it reduces drastically the number of DoF in the automatically generated LP network. Resulting LP network is represent the equivalent circuit of physical device.

As we will explain in the following sections, using the cochain method (such as cell-method or discrete exterior calculus) instead of classical finite element method (with nodal degree of freedom) greatly facilitates the automatic generation of the constitutive laws of the reduced lumped elements. Indeed, the topology of physical quantities defined in nodes, edges, surfaces or volumes is already fully compatible with the topology of physical quantities defined on discrete components of the lumped network.

As said in the introduction of this chapter, to illustrate the original method proposed here, an example of an electromechanical relay (magnetostatic problem) is described and solved in the following. This example will illustrate all the steps of the modeling process, from geometric modeling to the generation of the lumped (LP) network.

4.3.1 Step 1: Construction of the spatial mesh

Geometry of the case study. The example discussed in this section concerns a 2D magnetostatic case study for a simple electromechanical relay. Although the corresponding FEM model are not very time-consuming for this very simple example and do not require high-performance calculations, it is also obvious that in a context of optimization or sensitivity analysis, it will become much more demanding (time dependence for current or position control, moving parts, non-linear properties, physical or geometrical parametrization). This simple 2D magnetostatic problem was chosen in order to explain the methodology as clearly as possible.

The electromagnetic device consists of a fixed core with a Π-shape (armature), a coil in its lower part and a I-shape movable part (pallet), surrounded by air (or vacuum). In this example, we imposed a current of 0.69 A in a coil constituted with N =2000 turns. The device is 4 cm thick and the relative permeability of the core material is 1500. The geometrical dimensions are given in Figure 4.24(a): h =150 mm, g =30 mm, x =100 mm, y =70 mm and z =30 mm. The case study is solved as a 2D problem (according to a cross-section of the relay) but this method can be extended to 3D problems without significant change. As boundary conditions, the magnetic potential $\vec{A}$ is imposed null on the external boundary of surrounded air.

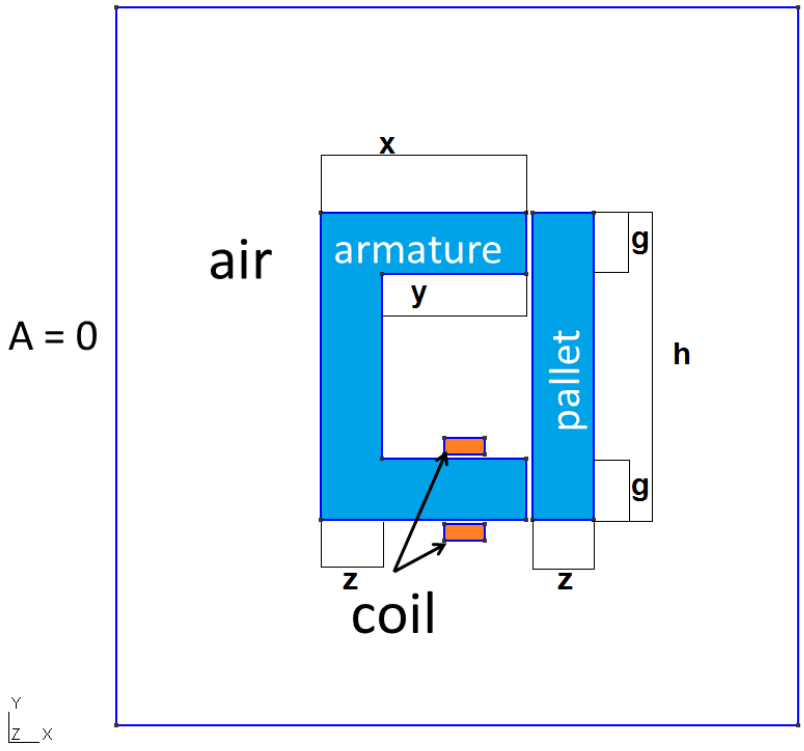


Figure 4.24: 2D Magnetostatic case study (relay): components and geometry.

Mesh generation. The GMSH software was used as meshing tool [Geuzaine and Remacle, 2009]. This allows to generate a mesh based on Delaunay-Voronoi method. Nevertheless, and as explained in chapter 3, a common difficulty with Delaunay-Voronoi meshes is that, sometimes, the circumcenter of many primal

elements (triangles) of the mesh are found outside these triangles. In such a case, for insuring the inner location of circumcenters re-meshing is required to improve the mesh duality. In chapter 3, we explained that it can be done with a software developed at INRIA-Nancy, France, [Lévy, 2014], called *Geogram*. The *Geogram* was used to improve mesh duality. We have compared the primal/dual meshing from two softwares (Geogram and GMSH) and reported no significant difference because the geometry is quite simple. For this 2D case, we finally used a mesh generated from GMSH without special adaptation (see Figure 4.25).

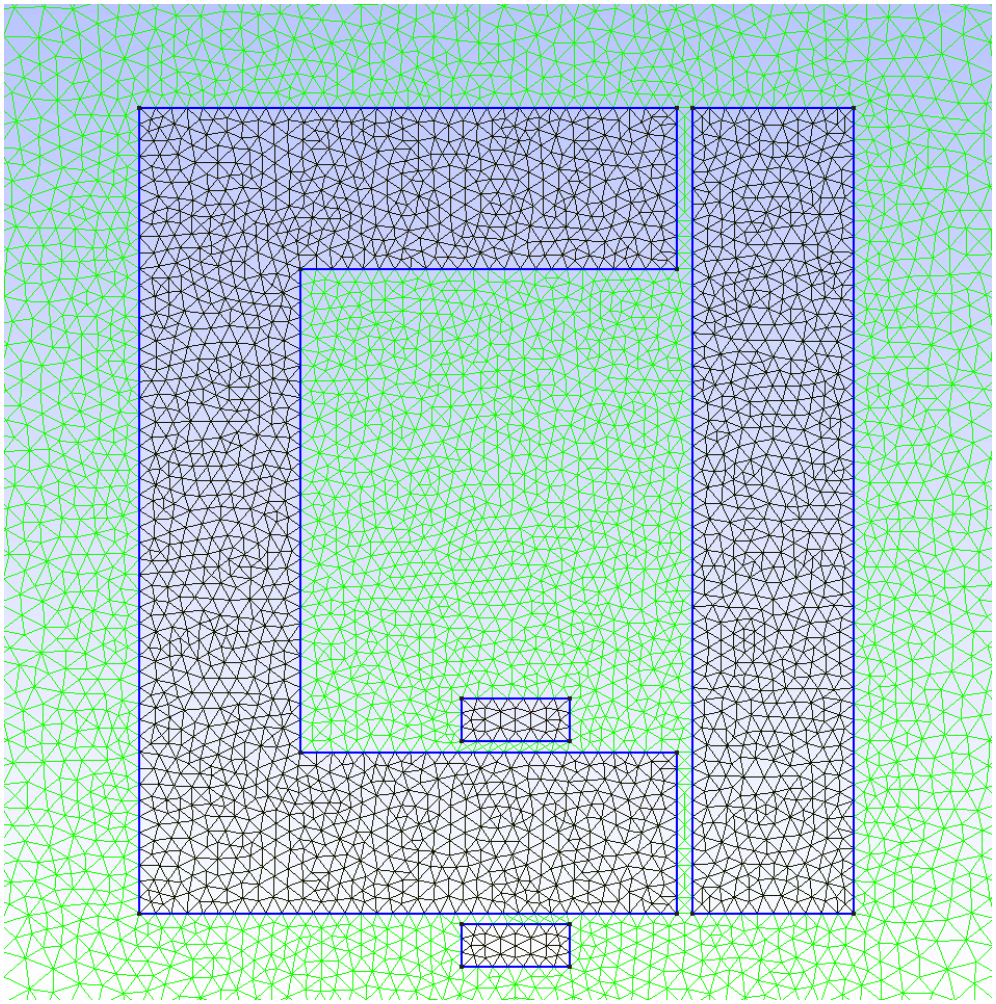


Figure 4.25: 2D Magnetostatic case study (relay): Primal mesh obtained from GMSH (13692 triangles).

4.3.2 Step 2: Solving physical equations

Cochain method calculation. The magnetostatic problem involve the use of the following two Maxwell equations:

$$curl\vec{H} = \vec{J}; \tag{4.27}$$

$$div\vec{B} = 0. \tag{4.28}$$

where $\vec{J}$ is the electrical current density in electric conductor, $\vec{H}$ the magnetic field, $\vec{B} = curl\vec{A}$ the magnetic induction and $\vec{A}$ the magnetic vector potential.

As we have already mentioned, the cochain method formulation is based on discrete exterior forms, *i.e.* cochains of algebraic topology. The cochains are identified as global variables. These global variables are defined and assigned to discrete geometric entities created by the tessellation of the geometry of the device. These discrete geometric entities are p-chains where p corresponds to the dimension of the geometric chain (nodes or 0-chains, edges or 1-chains, facets or 2-chains and volumes or 3-chains). Thus, a continuous domain Ω allows to define different discrete sets (and their duals): a set of vertices $\mathcal{N}$, a set of edges $\mathcal{E}$, a set of facets $\mathcal{F}$ and a set of volumes $\mathcal{V}$. The discrete differential forms (or p-cochains) are the dual objects of these geometric sets (the p-cochains are linear mappings that map p-chains to reals). The 0-cochains are associated with the vertex sets $\mathcal{N}$, the 1-cochains are associated with the edges $\mathcal{E}$, the 2-cochains are associated with the facets $\mathcal{F}$ and the 3-cochains are associated with the volumes $\mathcal{V}$. From vector analysis to algebraic topology, $\vec{H}$ becomes the discrete twisted differential 1-form h (defined on the dual edges, 1-chains). $\vec{J}$ becomes the discrete twisted differential 2-form j (defined on the dual facets, 2-chains). $\vec{B}$ becomes the discrete straight differential 2-form b (defined on the primal facets, 2-chains). $\vec{A}$ becomes the discrete straight differential 1-form a (defined on the primal edges, 1-chains). Straight forms are defined on the primal mesh whereas twisted forms are defined on its dual.

In our example, the magneto-static problem can be expressed using the discrete differential 1-form a (1-cochain), this is schematically shown by the upper part of Tonti diagram of the magnetostatic problem in Figure 4.26 [Alotto et al., 2013]. The a cochain corresponds to the magnetic vector potential $\vec{A}$ of the conventional differential formulation. It should be noted that in the case of a 2D mesh, this 1-cochain degenerates into a 0-cochain defined on node of the primal mesh. Using algebraic topology tools and discrete differential forms, the classical formulation:

$$curl\,\frac{1}{\mu} curl\,\vec{A} = \vec{J} \tag{4.29}$$

is rewritten as an algebraic equation with $\nu = \frac{1}{\mu}$, the reluctivity of the material and μ its permeability:

$$\tilde{C} \cdot \star_\nu \cdot C \cdot a = j \tag{4.30}$$

$\star_\nu$ is the physical Hodge star constitutive law of the magnetostatic problem. For orthogonal dual meshes, this operator is a diagonal matrix M_ν, defined and calculated in Alotto et al. [Alotto et al., 2013]. This is the inverse of the permeability constitutive operator $\star_\mu$. C is the incidence Edge-to-Face matrix of the primal mesh (corresponding to the discrete primal operator "curl") and $C^T = \tilde{C}$ is the Edge-to-Face incidence matrix of the dual mesh (discrete dual operator "curl"). As explained in the chapter 3, these discrete operators are calculated using the Python library *PyDEC* [Bell and Hirani, 2012] developed like a toolbox for Discrete Exterior Calculus [Sweldens and Schröder, 2000].

For the illustrative example, our cochain method allows to calculate the discrete form a corresponding to the magnetic vector potential $\vec{A}$. This calculation solves the algebraic equation $\tilde{C} \cdot \star_\nu \cdot C \cdot a = j$. The b cochain are then deduced

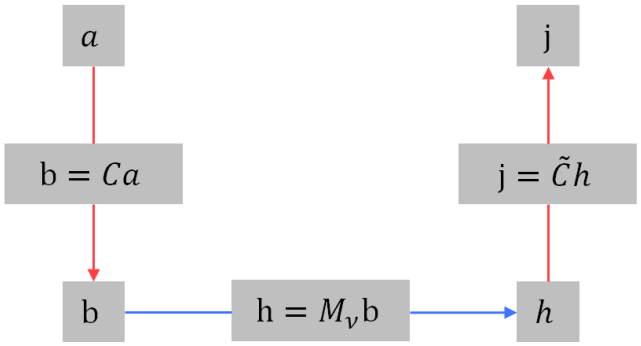


Figure 4.26: Upper part of the Tonti diagram of a magnetostatic problem [Alotto et al., 2013].

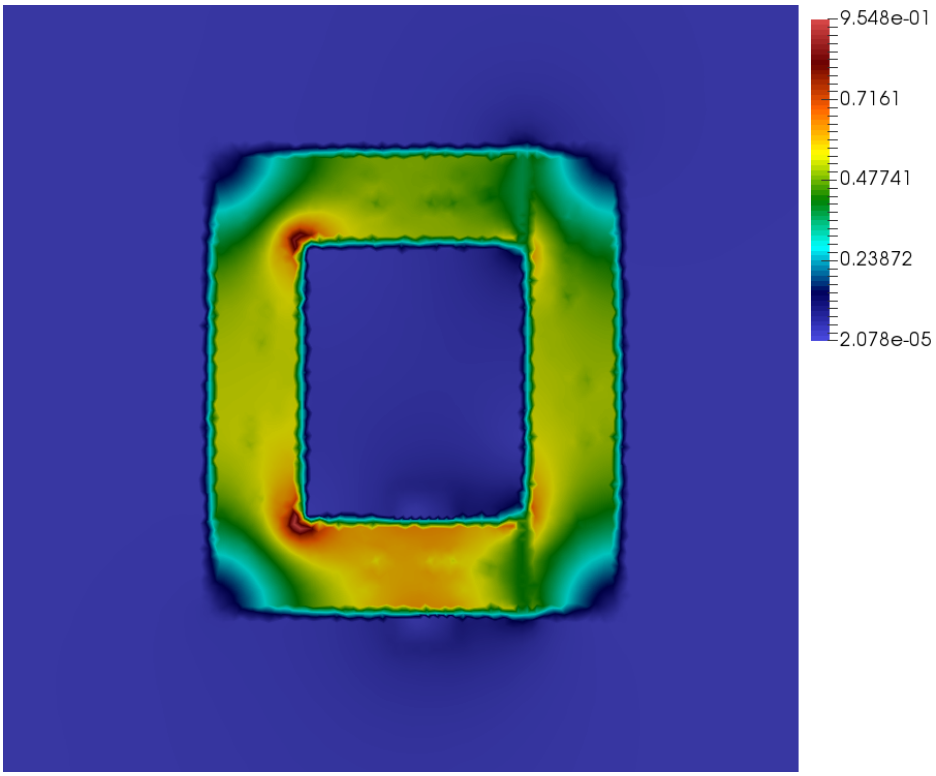


Figure 4.27: Discrete form b (linear interpolation by ParaView) for the relay example.

from this calculation by a second algebraic calculation $b = C \cdot a$ using the transpose of the Edge-to-Face connectivity matrix of the primal mesh. This second calculation is the equivalent in terms of exterior calculus of the classical vector operation $\vec{B} = \nabla \times \vec{A}$. Since our example uses 2D modeling, a is degenerated as a discrete 0-form (a 0-cochain) and b a discrete 1-form (a 1-cochain). Connectivity (incidence matrix) and co-boundary operators (transpose of incidence matrix) correspond respectively, in this case, to a Vertex-to-Edge incidence matrix and the "discrete" gradient operator. Nevertheless in the general 3D case, a is a discrete 1-form and b a discrete 2-form. Boundary and co-boundary operators correspond respectively to an Edge-to-Face incidence matrix and a "discrete" curl operator. The result of the calculation of the b cochain for the 2D relay example

is given in Figure 4.27. The results are displayed using the Paraview software. This software allows the display a cochain using linear interpolations (in 2D, b is a discrete 1-form).

4.3.3 Step 3: Clustering of homogeneous field regions

First, it is important to note that the DBSCAN clustering algorithms work with spatial distributions of physical quantities, *i.e.* with spatial densities of global quantities displayed as vector fields, and not with global quantities such as the discrete differential forms (cochains) calculated previously. For this reason, we proposed to add an additional step of interpolation of the cochains to obtain their spatial distribution, *i.e.* their density as "continuous" differential forms. It should be stressed that this additional step is not required for the computation of fields of cochains, which only considers global quantities and not field distributions. It is necessary here for the clustering procedure.

Second, we have discovered by different tests on spatial data that the main drawback of DBSCAN for our application is over-partitioning: the number of clusters may be extremely high.

Third, it could be very interesting to use different type of data in order to realize a clustering according to different physical domains in our model. So we can merge different data of clustering in one.

The following three paragraph present our proposals for these issues.

Interpolation using Whitney Elements As mentioned above, all clustering methods on space domains require knowledge of the spatial distribution of fields, *i.e.* density. It is therefore necessary to calculate the value of the "continuous" differential forms at each point in the space from the discrete differential form calculated on the mesh and its p-chains.

The interpolation of the cochain method quantities in order to obtain the spatial distribution of cochains is conducted using Whitney elements [Whitney, 1957]. These elements interpolate discrete forms within each element and link continuous differential forms to vector fields (densities of physical quantities [Bossavit, 1988]). For simplicity, only the 2D Whitney functions will be considered in this chapter. More information on the use of Whitney elements in this context can be found in Desbrun et al. [Desbrun et al., 2005].

In order to interpolate the discrete 0-forms (scalar quantities attached to the vertices) within each volume (element $\mathcal{V}_k$), we need to find a set of functions that produce a linear combination of the scalar values at the vertices. These functions are the basic piecewise linear *test functions* used in the finite element method [G. Dhatt, 1984]: for a point of coordinates $\mathbf{x} = [x_1, x_2, x_3]$ in a volume element $\mathcal{V}_k$, a piecewise linear *test function* $w^0(\mathbf{x})$ meets the following properties,

$$w^0(\mathbf{x}) = \begin{cases} 1 & \text{if} \quad \mathbf{x} \in \mathcal{V}_k; \\ 0 & \text{otherwise.} \end{cases} \tag{4.31}$$

This function is defined by a linear combination of the values at the vertices. For the case of a triangular element with vertex indices i, j, k, we have,

$$w^0(\mathbf{x}) = w_i^0(\mathbf{x}) + w_j^0(\mathbf{x}) + w_k^0(\mathbf{x}). \tag{4.32}$$

These functions, w_i^0, w_j^0, w_k^0, are usually defined for a triangle in a local coordinate frame $\boldsymbol{\chi} = [\xi, \eta]$, thus, the coordinates of points are $\boldsymbol{\chi}_i = [0,0]$, $\boldsymbol{\chi}_j = [1,0]$, $\boldsymbol{\chi}_k = [0,1]$, and the Whitney's nodal functions are defined by the next equations,

$$w_i^0(\boldsymbol{\chi}) = 1 - \xi - \eta; \tag{4.33}$$
$$w_j^0(\boldsymbol{\chi}) = \xi; \tag{4.34}$$
$$w_k^0(\boldsymbol{\chi}) = \eta; \tag{4.35}$$

the transition between the local and global frame is performed by the jacobian matrix.

Using the same ideas, we can calculate a vector field inside the same triangle (local coordinate frame) using a linear combination of the vector fields on the edges (triangle boundaries),

$$\boldsymbol{w}^1(\boldsymbol{\chi}) = \boldsymbol{w}_{ij}^1(\boldsymbol{\chi}) + \boldsymbol{w}_{jk}^1(\boldsymbol{\chi}) + \boldsymbol{w}_{ki}^1(\boldsymbol{\chi}); \tag{4.36}$$

where $\boldsymbol{w}_{ij}^1, \boldsymbol{w}_{jk}^1, \boldsymbol{w}_{ki}^1$ are Whitney's edge functions. For the edge e_{ij} the Whitney function is defined by,

$$\boldsymbol{w}_{ij}^1(\boldsymbol{\chi}) = w_i^0(\boldsymbol{\chi}) \nabla w_j^0(\boldsymbol{\chi}) - w_j^0(\boldsymbol{\chi}) \nabla w_k^0(\boldsymbol{\chi}); \tag{4.37}$$

the others edges e_{jk} and e_{ki} have similar definitions.

These Whitney functions associated with the edges verify the follow properties:

$$\int_{\sigma_1} \boldsymbol{w}_{ij}^1 \cdot \mathrm{d}\mathbf{l} = \begin{cases} 1 & \text{if} \quad \sigma_1 = e_{ij} \\ -1 & \text{if} \quad \sigma_1 = e_{ji} \\ 0 & \text{if} \quad \sigma_1 \neq e_{ji} \quad \text{or} \quad \sigma_1 \neq e_{ij} \end{cases} \tag{4.38}$$

where, σ_1 is a 1-chain (an edge).

In a magnetostatic 2D case only Whitney elements of degree 0 (to calculate the field $\vec{A}$) and 1 (to calculate the fields $\vec{B}$ and $\vec{H}$) are required. For the relay example, the field $\vec{B}$ has been calculated at each circumcenter using the results of the cochain method and a Whitney interpolation over each edge. The result obtained is shown in Figure 4.28. By comparing this figure with Figure 4.27 displayed with ParaView, we can see equivalent result because Paraview also uses interpolation (linear) to display these results.

Clustering and adaptation of the original DBSCAN algorithm. In our work, we used a Python implementation of the algorithm proposed by Sci-kit [project by David Cournapeau, 2007]. In this implementation of the DBSCAN algorithm, the *min_pts* is fixed to 4, *eps* = 0.01 (that depends on geometrical dimension of relay and mesh precision) and *min_samples* represents a quantity directly related to the minimum number of clusters. The clustering process is first carried out on the basis of information contained in the field $\vec{B}$ computed with Whitney elements at the circumcenter of each cell of the primal mesh. Figure 4.29 shows the amplitude of the field $\vec{B}$ before clustering as a function of the xy position in the plane. The size of the first dataset corresponds to the number of triangles in the primal mesh since the previous Whitney interpolation was performed at the barycentre of each triangle. As DBSCAN is a multi-data

algorithm, we can improve our clustering by using heterogeneous data sets, for example, $\vec{B}$-field and $\vec{H}$-field and then combining corresponding results.

In our example, we ultimately used information from the fields $\vec{B}$ and $\vec{H}$. These two sets of information are used together to obtain clusters that are more representative of physics. For example, we've set a goal to identify all the air gaps in our model. The clustering procedure thus merged information from field $\vec{B}$ (related to the position of the magnetic flux tubes) with information from field $\vec{H}$ (related to the position of the materials and their permeability property). Running the DBSCAN algorithm on this dataset significantly reduces the number of DoF from 6923 DoF in the field computation to 184 clusters. The clusters finally obtained are shown in Figure 4.30. One specific color correspond to one cluster (the circles, that represent a specific value, are the result of the interpolation in the cells).

However, we can note that one of the limitations of the DBSCAN algorithm remains its over-partitioning since the number of identified clusters remains relatively large. Reducing the number of clusters is the next objective.

In order to reduce the total number of clusters, we were inspired by methods used in digital image processing so, for a while, we will consider our spatial distribution of field as a color image. Usually, image processing starts with building a histogram. The histogram of an image measures the distribution of each color in the image. For a color x, the histogram allows to know the probability of falling on a point (or a pixel in terms of image processing) of value x in the image. Dividing each value of the histogram by the total number of points in the image gives a standardized histogram. The normalized histogram corresponds to an empirical probability distribution (all values are between 0 and 1 and the

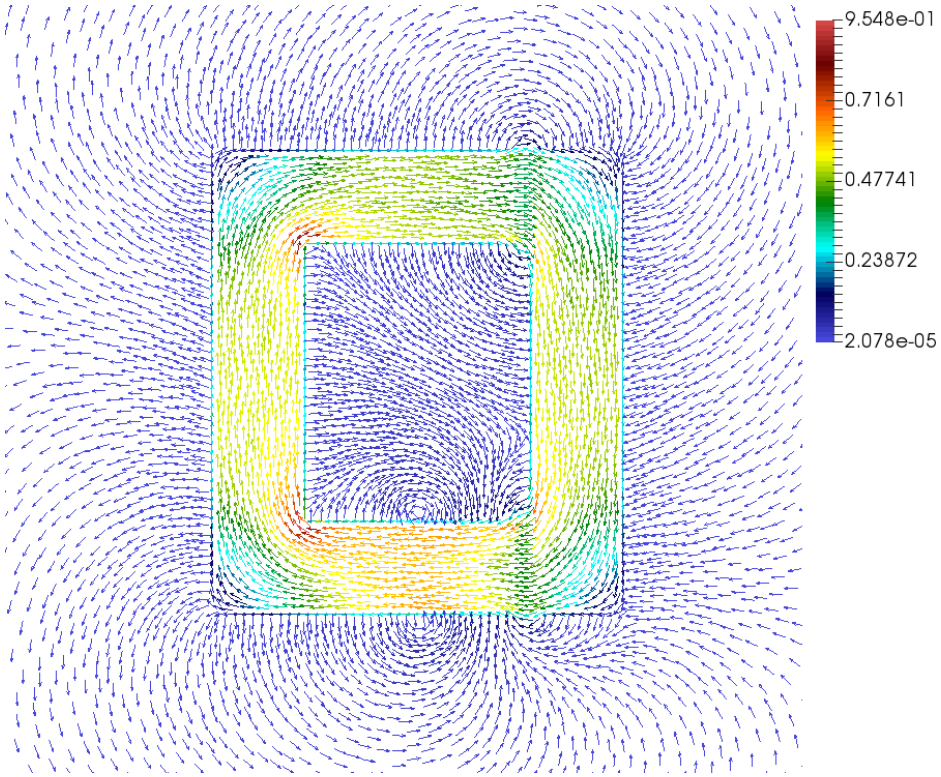


Figure 4.28: Interpolation of $\vec{B}$ field with Whitney elements.

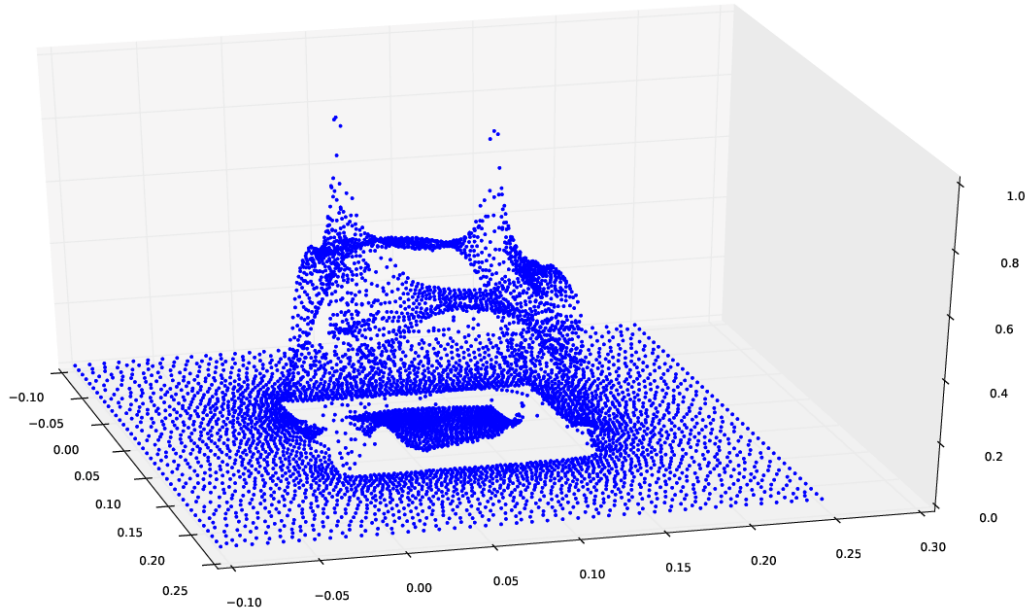


Figure 4.29: Value of the $\vec{B}$-field (z-plane is the amplitude of $\vec{B}$) in the xy-plane after Whitney interpolation in circumcenters of primal cell.

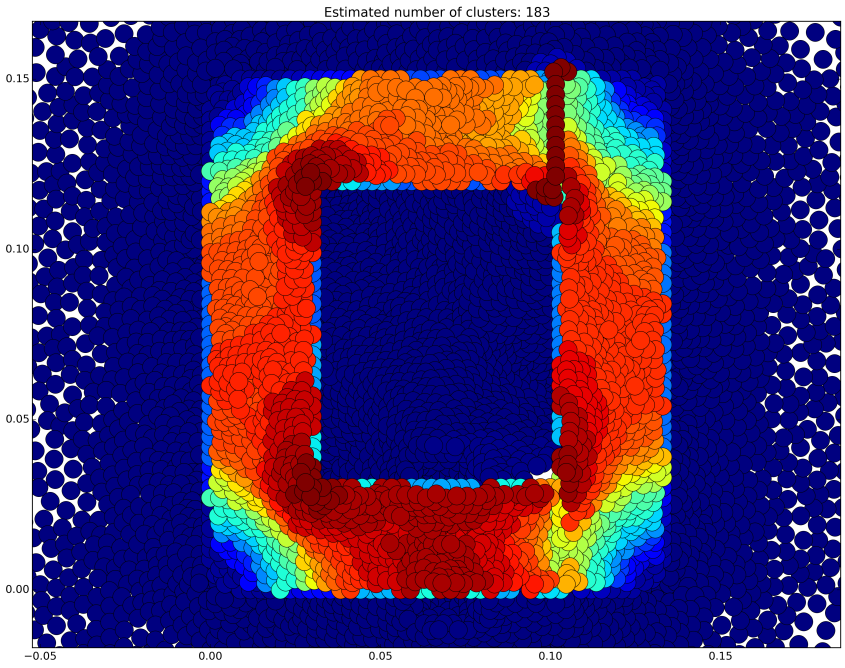


Figure 4.30: Result of DBSCAN - clustered cells: 184 classes (One specific color (and not one specific circle) corresponds to one cluster. The color range does no exactly match that of Figure 4.27 and 4.28).

sum of the values is 1). The histogram allows us to obtain a general information on the appearance of the image: how many different colors are there in the image and how often they occur. As in computer image each color is represented

by a integer numbers (the total number of colors is limited, it depends on color systems). In our case, the values of each point can have its own unique value because these values are *real* numbers (and not *integers*, like color identification in the digital image).Therefore the probability that at least two points have the same *float* value is low. But in practice, we can round normalized values up to 2 decimals: if we get a first glance to a histogram of our clustering, represented on Figure 4.31, there are a lot of points[1] with the same value of $\vec{B}$-field and so we can have already a kind of first reduction of density levels for clustering.

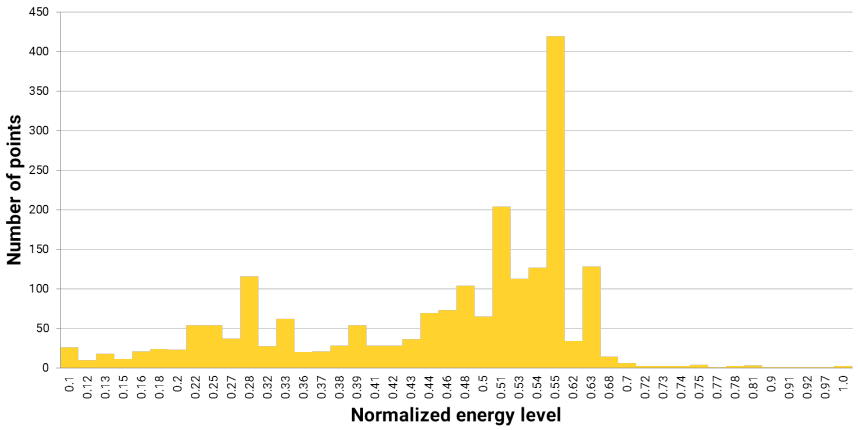


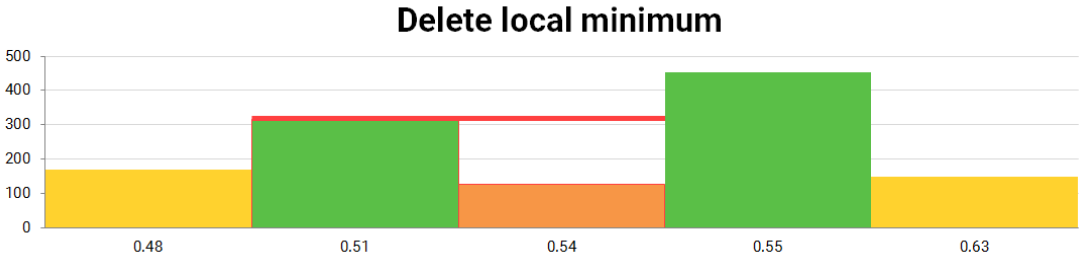
Figure 4.31: Histogram of density levels of initial clustering of Figure 4.30

If we aggregate all nearby points with the same value of $\vec{B}$-field, we can have a first grouping of clusters. A naive grouping of clusters have two disadvantages: large number of zones and, the most important, zones can de disconnected as the same value may appears anywhere else in distribution (figure 4.31).

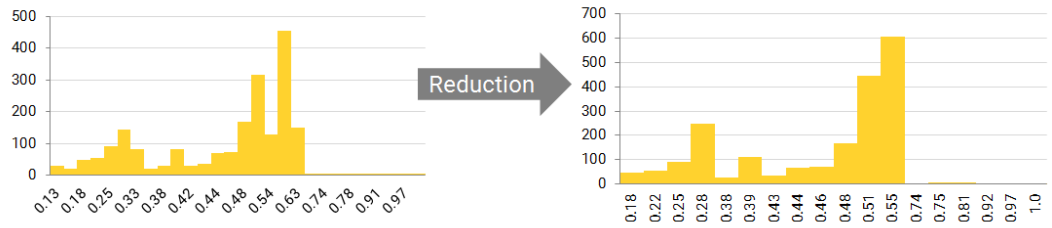
After a review of the image processing and compression literature, we were able to find a solution that allowed us to reduce the number of clusters while maintaining the desired connectivity properties: we can apply a classical algorithms which can reduce a number of level of density by an, so called, *Image Quantification algorithm*. Here we will give a trivial example that helps to understand the global strategy. Consider an image with an histogram with values: 0.21, 0.25, 0.41, 0.45, 0.48, 0.58, 0.68, 0.78, 0.81, 0.93, 0.94 (11 normalized density levels). If we round all value up to the tenths, we will get 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 (8 normalized density levels). Even this example is rough, this helps to understand a global idea. In other words, we can use a minimal optimization procedure. For our aim, we used algorithm based on local optimization which is very closed to a waterfall algorithm in mathematical morphology [Beucher, 1994]. This algorithm is iterative, it reduce all local minimal in histogram and also the total number of density levels. This algorithm is used for image compression, its steps are shown in Figure 4.32. The final histogram after this pre-processing step is given in the Figure 4.33.

This step is only a pre-processing step. For real image it is difficult to have a suitable cluster identification because of the lack of geometrical notation in

[1]Note that each circle in the Figure 4.30 is a Whitney interpolation in circumcenters of primal cell. The color reveals only cluster affiliation. The color range does no exactly match that of of Figure 4.27 and 4.28



(a) Fragment of histogram with local minimum to merge.



(b) Reduction of density levels.

Figure 4.32: Reduction of density levels by Waterfall algorithm.

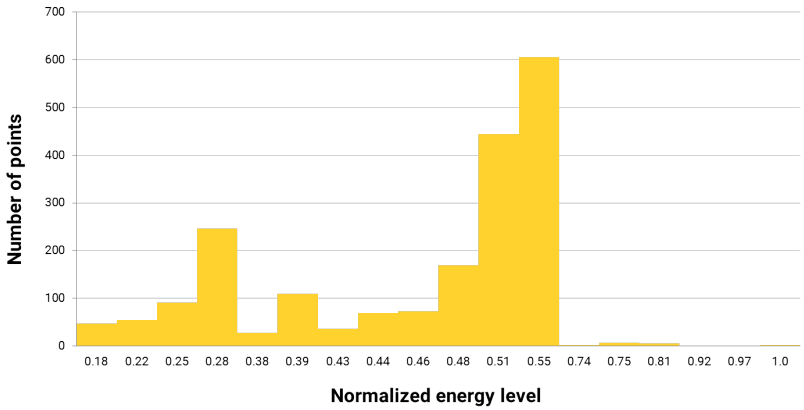


Figure 4.33: Illustration of the reduction of dofs with histogram: histogram of density levels of final clustering (corresponding for 25 clusters).

histograms and so set of point not always form a simple connected space. The aim of this pre-processing is to limit a number of density levels for following segments identification. After this pre-processing, we therefore can start to use a second time the DBSCAN clustering algorithm. Using this method on our example allows us to go from 184 clusters in the first phase (see Figure 4.30) to 25 in the final phase (see Figure 4.34). The generation of the LP network from this final result is explained in the following section.

4.3.4 Step 4: automated generation of a network model

The clustering carried out in the previous section aims to identify the best compromises in terms of magnetic circuit. A magnetic circuit is an LP model based

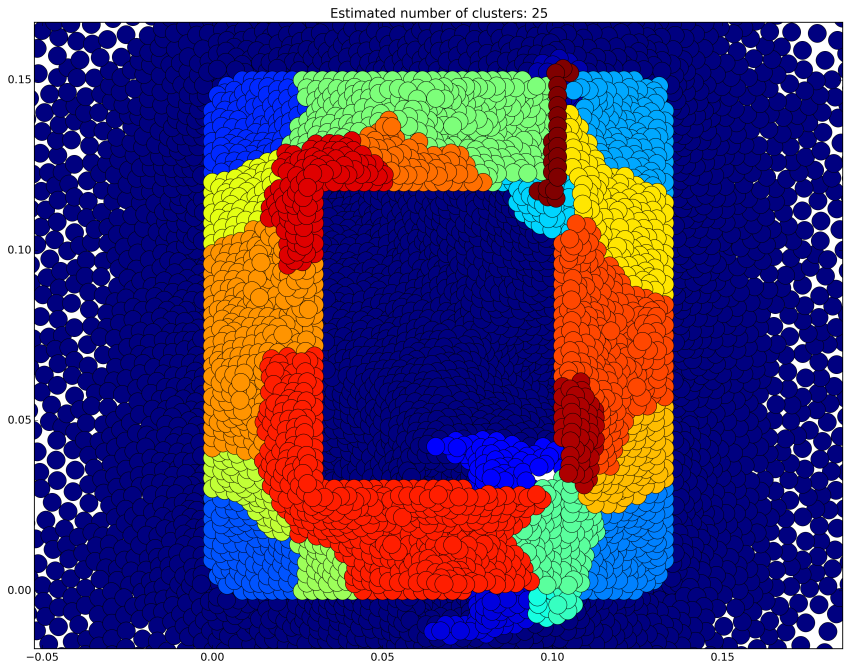


Figure 4.34: Final clustering with 25 clusters

on the lower part of the Tonti diagram of magnetostatics such as the one shown in Figure 4.35. The lower part (dual formulation) of this diagram uses global physical quantities which are of three types:

1. scalar magnetic potential (0-cochain Ψ defined on the dual mesh).
2. scalar magnetic potential difference also called magnetomotive force (mmf). Note that, summation of 1-cochain h defined on the dual mesh gives us a total mmf by element. If current sources are present, let us note that a mmf source h_s must be added like a second potential that produce a not null circulation ($j = \tilde{C}\dot{h}_s$). It was done for constitutive law of the electrokinetic problem.
3. magnetic flux ϕ given as summation of 2-cochains b defined on the primal mesh.

The generalized Kirchhoff laws applied on magnetic circuits (reluctance networks) involve the conservation of the magnetic flux at a node (Generalized Kichhoff Current Law) and the zero sum of mmf over a cycle (Generalized Kichhoff Voltage Law). A magnetic circuit is composed of two types of discrete lumped components, reluctances and mmf sources. A reluctance represents a flux tube, *i.e.* a preferred passage of the magnetic field $\vec{B}$. For each of these tubes, discrete components can be identified and positioned in this network. Because it is based on the definition of a single-valued magnetic scalar potential Ψ, the construction of a magnetic network based on the previous calculations must necessarily be carried out in two steps. Indeed, it is well known that a magnetic scalar potential is multi-valued in the portions of space where electrical currents are present

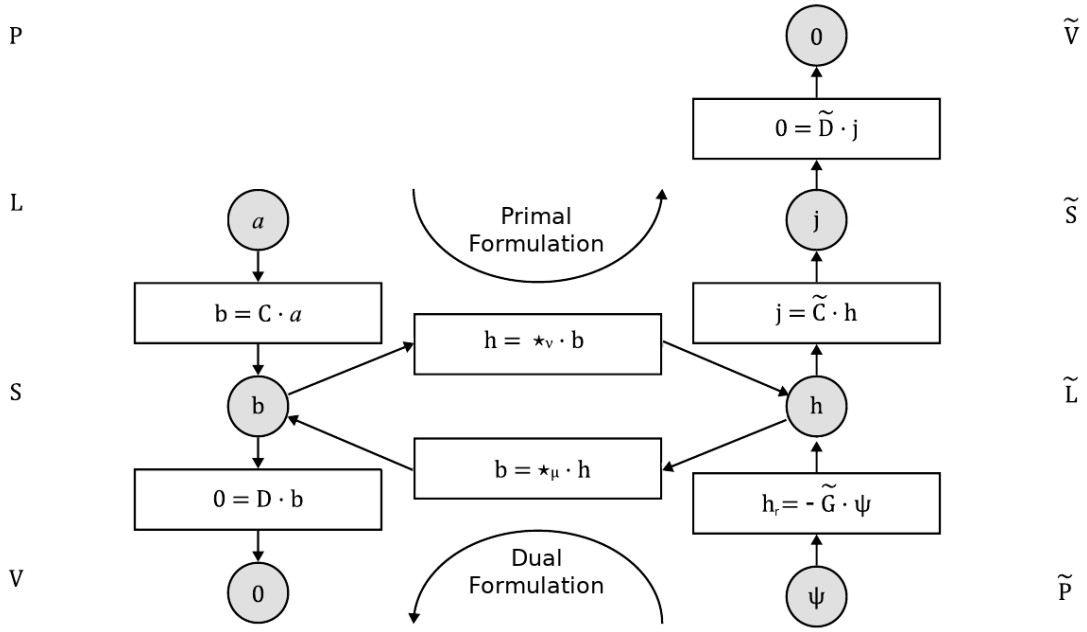


Figure 4.35: Complete Tonti diagram of magnetostatics with no current sources ($j = 0$ here). Ψ is a well defined "single-valued" magnetic scalar potential only in region with no current source [Verite, 1987].

[Verite, 1987], therefore the current and mmf sources must be treated separately from the calculation of the reluctances. A magnetic network is in fact a discrete representation of a magnetostatic problem expressed in the dual formulation of the one we used previously. It uses as DoF the magnetic scalar potential Ψ (lower part of the Tonti Diagram in Figure 4.35) instead of the magnetic vector potential $\vec{A}$ (upper part of the Tonti Diagram in Figure 4.26). The purpose of our method is not to solve the magnetostatic problem using this dual formulation but to generate a reduced model (LP model) from the previous results obtained by cochain method. In this case, we can notice that the mmf correspond exactly to the 1-cochains h integrated on the dual edges and that the magnetic fluxes ϕ correspond exactly to the 1-cochains b (in 2D otherwise, they are 2-cochains in 3D) integrated on the primal edges (on the primal surfaces in 3D). To determine the reluctances of the magnetic network, it is therefore necessary to calculate all the fluxes ϕ_k and mmf_k associated with each cluster. This is the first step of automatic generation of this network from the results of clustering.

In previous sections, we used a as a degrees of freedom to solve the magnetostatic problem. Nevertheless, to generate a magnetic circuit as an LP model, Ψ will be used as DoF instead of a. This change is equivalent, for a network, to go from a node analysis to a mesh analysis. This network uses global potential Ψ, scalar magnetic potential difference (magnetomotive force (mmf) as a sum of 1-cochain h defined on the dual mesh) and magnetic flux (ϕ as a sum of 2-cochain b defined on the primal mesh). Thus, we have to be careful because the new DoF Ψ are only defined if we separate the source of current (source of mmf) from "passive" magnetic components (reluctances). This separation is displayed on the Figure 4.36 which is the lower part of the Tonti diagram with the addition of current sources through the decomposition of h into two components h_s and h_r.

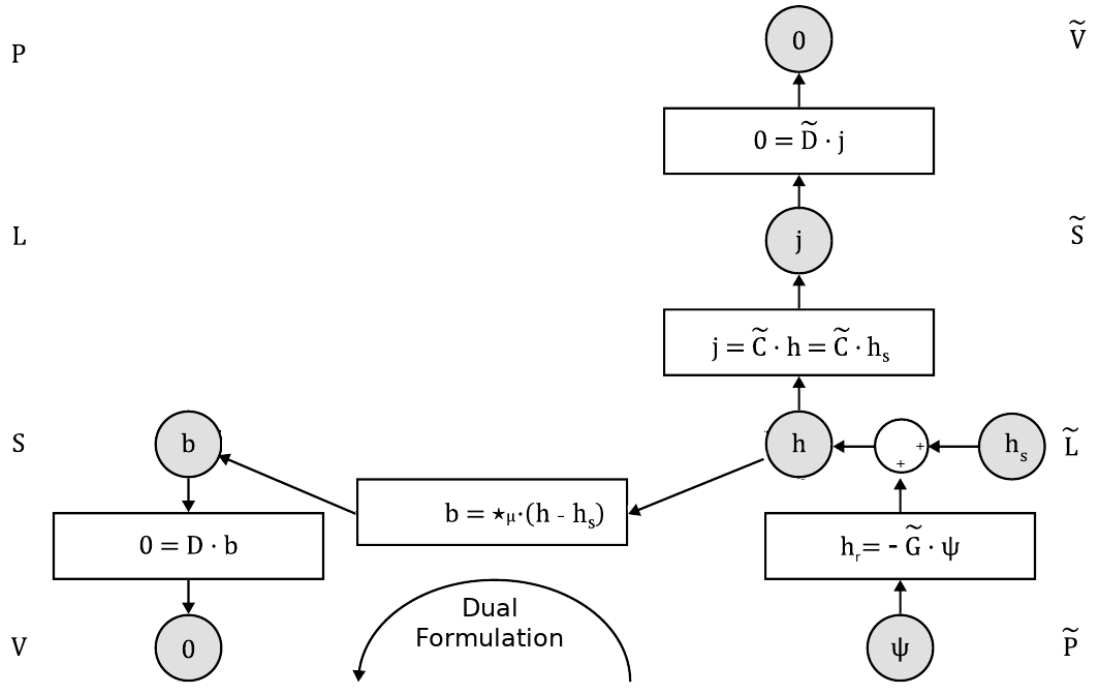


Figure 4.36: Complete Tonti diagram of magnetostatics with current sources. h_s is due to current sources such that $j = \tilde{C} \cdot h_s$ because $\tilde{C} \cdot (-\tilde{G} \cdot \Psi) = 0$.

The generated LP magnetic model network is then constituted with mmf sources and reluctances components.

When the topology of the magnetic network is generated and the values of its reluctances are calculated, the mmf sources are then added, using a method quite close to what is achieved in dual magnetostatic formulations, by decomposing the fields $\vec{H}$ into two contributions. A recent article [Nunes et al., 2017] proposes a two-step approach quite similar to what we propose here, however, this article uses a classical vector analysis approach while we use discrete differential forms in this thesis. This decomposition is in fact standard for all dual/complementary magnetostatic finite element formulations [Marmin et al., 2000a]. [Nunes et al., 2017] decompose the field $\vec{H}$ into two fields, $\vec{H}_s$ and $\vec{H}_r$. The field $\vec{H}_r$ is the field due to the sources of mmf such that $curl\vec{H}_s = \vec{J}$. The field $\vec{H}_r$ is a reaction field, based on a single well-defined magnetic scalar potential Ψ such that $\vec{H}_r = -grad\Psi$. $\vec{H}_r$ is then related to reluctance in the LP network and $\vec{H}_s$ is related with mmf sources. This construction is exactly the same as our, except that we are using differential forms instead of vector fields (see our Tonti's diagramme in Figure 4.36). In next paragraph, we will first explain the construction of the reluctance network without taking into account mmf sources. These will be added in further paragraph.

Generation of network topology and computation of reluctance values. Once a large region has been clustered, the first step is to divide its S boundary into S_k sub-boundaries containing only positive or negative magnetic flux (no sign change of the discrete differential form b defined for these sub-boundaries). This sign is then used to define incoming or outgoing flux tubes. A magnetic flux

tube associated with a magnetic flux ϕ_k is then defined for each sub-boundary S_k (with a positive or negative value as seen in the red or blue boundaries of a cluster shown in Figure 4.37). In the context of vector analysis, this flux is traditionally calculated as $\phi_k = \int_{S_k} \vec{B} \cdot d\vec{S}$. This calculation requires the definition of a metric (a scalar product), whereas this is not the case for a calculation using differential forms (continuous or discrete) and algebraic topology [Bossavit, 2012]. Consequently, the calculation is done in this framework using simple summations of the b_i cochains along a chain equal to S_k. This chain is constituted with primal cell c_i:

$$\phi_k = \sum_{\forall\, c_i \in S_k} b_i \tag{4.39}$$

Due to the homological/cohomological structure of this model, the topological properties of the magnetic cochains b ensure the conservation of the magnetic flux in a closed area S surrounding each cluster:

$$\sum_{\forall\, S_k \in S} \phi_k = \sum_{\forall\, S_k \in S} \left(\sum_{\forall\, c_i \in S_k} b_i \right) = 0 \tag{4.40}$$

In a magnetic network, mmf are the dual quantities of fluxes. This implies that a mmf_k must be defined on the dual chain of each flux surface S_k. Since the fluxes are associated with the primal mesh (primal edges for 2D geometry or primal surfaces for 3D geometry), the dual variables are associated with the dual mesh: mmf correspond to 1-cochains h (whether 2D or 3D) defined on dual 1-chains, i. e. paths Γ_k composed of 1-cell c_j^* defined on the edges of the dual mesh:

$$mmf_k = \sum_{\forall\, c_j^* \in \Gamma_k} h_j \tag{4.41}$$

To define the mmf_k, dual of the flow ϕ_k, it is convenient to define a central node/vertex in the dual mesh of the region defined by the cluster (the circle filled in black of Figure 4.37). The reluctance is then calculated using Hopkinson's law using on the one hand the flux ϕ_k of each sub-surface S_k and on the other hand mmf_k defined along a path Γ_k starting on this surface (cross of Figure 4.37) and ending on the central node defined above:

$$\mathfrak{R}_k = \frac{mmf_k}{\Phi_k} = \frac{\sum_{\Gamma_k} h}{\sum_{S_k} b} \tag{4.42}$$

In Figure 4.37 this central node is chosen close to the cluster's circumcenter. However, it is important to note that the "exact" location of this central node is arbitrary (but inside the cluster) because the magnetic scalar potential is single-valued, which means that a mmf defined on a path does not depend on the precise path but only on the values of the potentials Ψ on its ends. mmf are therefore "exact" discrete differential forms, i. e. they derive from a scalar magnetic potential Ψ such that $h = -\tilde{G}\,\Psi$ with $\tilde{G}$, the discrete differential operator corresponding to the gradient on the dual mesh. Thus, the mmf between two interconnection points on the cluster's borders does not depend on the precise position of the central node and it can therefore be placed anywhere in the cluster. The process

of calculating the reluctance network for a cluster is illustrated in Figure 4.37. When this is done for every cluster, the complete network is the assembly of all these clusters according to the complete interconnection of the devices. This assembly is nothing more than a connection process between all the reluctance networks defined on each cluster. This connection is based on incidence matrices previously calculated by PyDEC. This assembly automatically produces the reluctance network of the complete geometry of the modeled system. Figure 4.38 represents the relay reluctance network for a clustering using 25 classes. This first step of the automatic LP model generation procedure gives the general structure of the discrete network but does not yet include mmf sources. These sources must be added in a second step, this is the purpose of the next paragraph.

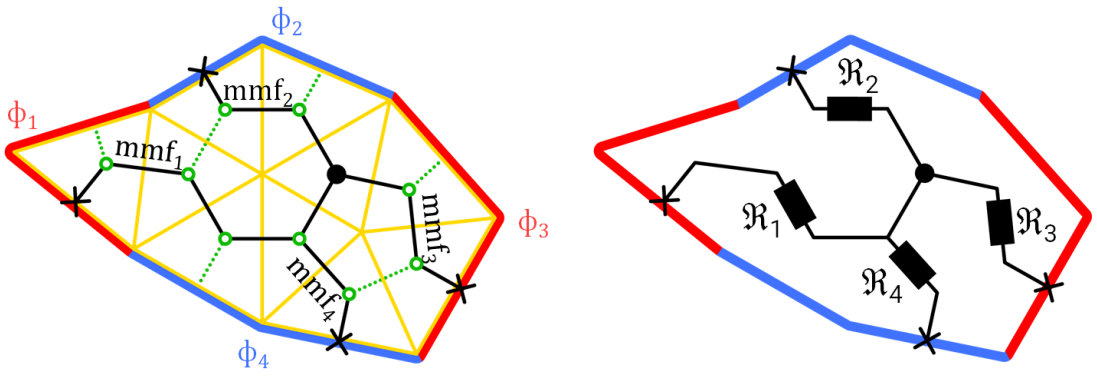


Figure 4.37: One clustered region with its boundary sub-divided into tubes of magnetic flux (One reluctance by sub-boundary).

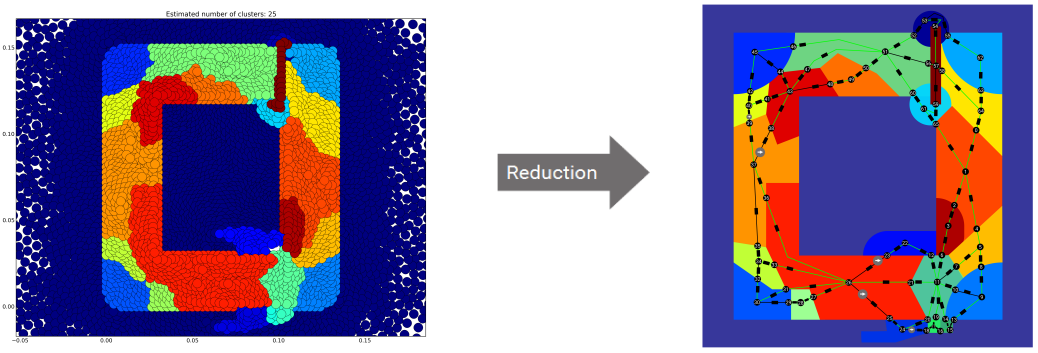


Figure 4.38: Clustered cells (25 classes) and the corresponding lumped network (see Figure 4.40 for details on the network).

Addition of magnetomotive force sources in the network In the previous paragraph, we explain how to generate a magnetic network from clustering and cochain method results. The reluctance values and network structure were calculated. Nevertheless, mmf sources must now be added to satisfy Ampère's law. As explained in [Nunes et al., 2017], this can be easily achieved by adding two contributions $\vec{H}_s$ (due to mmf sources) and $\vec{H}_r$ (due to reluctance) of the

magnetic field such that $\vec{H} = \vec{H}_s + \vec{H}_r$. The contribution $\vec{H}_s$ due to sources in Ampère's law is rewritten in integral form as:

$$curl\vec{H}_s = \vec{J} \Rightarrow \oint_{\Gamma=\partial S} \vec{H}_s \cdot d\vec{l} = \int_S \vec{J}\, dS = mmf_s \tag{4.43}$$

with Γ a closed path surrounding a surface S and defined on the dual mesh. J is the current density across the surface S and mmf_s the mmf due to the current source.

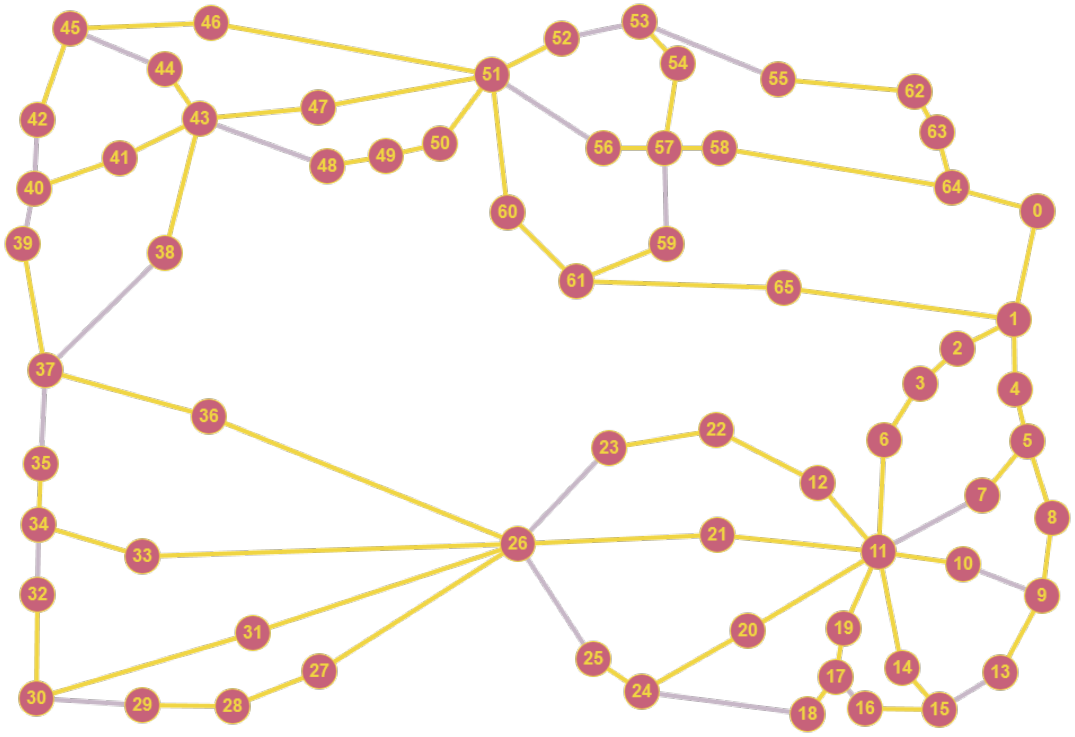


Figure 4.39: Minimum spanning tree of digraph: branches of the tree are traced in yellow and links in grey.

In order to remain consistent with the rest of the chapter, mmf sources are added using algebraic topology tools. The network of reluctances calculated in the previous section can be described by a topological space called a linear directed graph or *digraph*. As explained in chapter 2, this is none other than a 1-dimensional simplicial complex such as the one represented in Figure 4.38 (right) and Figure 4.39. This digraph consists of vertices and edges from the original dual mesh used in the cochain method. The number of mmf sources to be included in the reluctance network is not arbitrary but it results from an invariant in the digraph called its *nullity* ν. According to matroid theory [Recski, 1989], this quantity is the nullity of the directed incidence matrix associated with this graph. If the graph has n_v vertices and n_e edges, the nullity is given by $\nu = n_e - n_v + c$, where, c is the number of connected components of the graph ($c = 1$ for any standard connected mesh used in FEM or cochain method). $\rho = n_v - c$ is the rank of the directed incidence matrix, or simply the rank of the digraph. The name "nullity" is rarely used but is more commonly known as *cycle rank* or *number of cyclomatic* of the graph. It is also the first Betti number

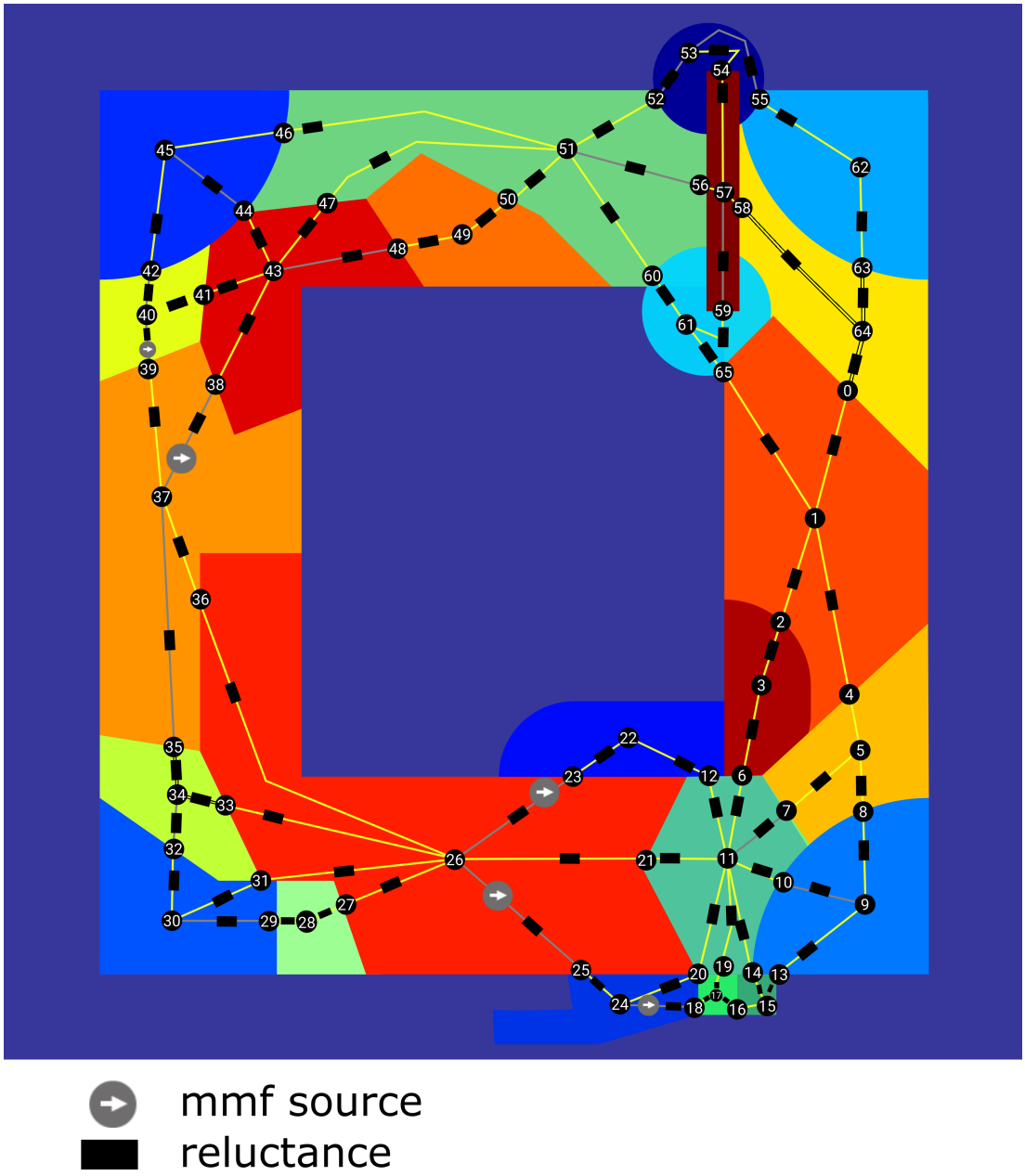


Figure 4.40: Clusters and final LP networks. Only grey links contain mmf sources .

of the graph or the rank of the first (integer) homology group of this complex [Berge, 1973]. Alternatively, ν can be seen as the number of independent cycles in the graph: we therefore need only one mmf source per independent cycle of the digraph, i. e. only ν mmf sources should be added throughout the reluctance network. The idea of using graph theory to place mmf sources is not new and has already been used in connection with finite elements. For example, Le Menach and Marmin et al. [Le Menach et al., 1998, Marmin et al., 2000b] use the notions of tree and co-tree in a way that is quite similar to what we have done here.

The placement of the mmf sources can be done using the fundamental loop and cut-set analysis of electrical circuit theory [Desoer and Kuh, 1969b, Chua and Lin, 1975]. Let be a simply-connected ($c = 1$) digraph composed of n_v vertices ($n_v = 62$ in our example) and n_e edges ($n_e = 80$ in our example). Its nullity is

equal to $\nu = n_e - (n_v - 1) = 19$ and its rank is equal to $\rho = n_v - 1 = 61$. On this digraph, we can find a fundamental, or spanning, tree that will be composed of ρ edges called its branches. The complementary of the tree is called the co-tree and it is composed of ν edges called its links. It is easy to show that the ν sources can be placed on the ν links of the spanning tree defined on the digraph (see Figure 4.39) for the following reasons:

1. Links are the complementary edges of the tree branches, then if we add a link to the tree, a loop (a closed path Γ or a cycle) is defined.

2. It is these loops that are used in Ampere's law with current source given in equation (4.43).

3. It is therefore necessary to add a source of mmf on each of these loops and this source is precisely added to the link used to create this cycle.

The value of this mmf source is equal to $\int_S \vec{J} dS = \sum_k n_k \cdot i_k$. In this expression, i_k is the current with turns number n_k of a coil k crossing the area S bounded by Γ. If no winding passes through the surface S, the corresponding mmf source is of zero value. In the network generated with 25 clusters displayed in Figure 4.38, the nullity ν is equal to 19, so it is necessary to add 19 mmf sources. These 19 links corresponds to the 19 grey edges in Figure 4.39. However, for this relay there is only one coil (two current surfaces, one positive surface and one negative), and few fundamental cycles enclose a part of this coil. Therefore most of these mmf sources are null. The final reluctance network of the relay, the selected spanning tree and the corresponding non-zero mmf sources located on its links are displayed in Figure 4.40.

To conclude this section, it is interesting to summarize the level of model reduction obtained by the proposed methodology. The table 4.1 presents the results obtained in terms of mesh size, number of DoF, size of data sets for 1) the cochain method, 2) the clustering method and 3) the LP model obtained in the end. The number of DoFs in the LP model depends on the method used to solve it, either a "Node Analysis" method or a "Mesh Analysis" method [Desoer and Kuh, 1969a]. Depending on the topology of the network, one of these two methods is generally more effective than the other.

Table 4.1: Summary of data set dimensions and DoF numbers.

Cochain model	Primal Mesh Vertices: 6923 Edges: 20614 Triangles: 13692	Dual Mesh Vertices: 13 692 Edges: 20614 Voronoi cells: 6923	Number of DoF 0-cochain a: 6923
Clustering computing	Data set on field B (interpolated in each primal triangle) : 13692	Initial clustering 184 clusters	Final reduces clustering 25 clusters
LP model	Digraph vertices: 62 Digraph edges: 80 Number of branches of the tree: 61 Number of links of the co-tree: 19	Number of components Reluctances : 80 mmf sources: 19 (of which only 5 are of non-zero value)	Number of DoF Node analysis: 61 (magnetic scalar potential) Mesh analysis: 19 (mesh magnetic flux)

4.3.5 Conclusion on the automatic generation of reduced LP model

This chapter proposes an innovative concept of automatic generation of discrete component networks from results of cochain method. The generated network is of a reduced order compared to the initial model and the reduction is based on an unsupervised clustering analysis allowing aggregation of the mesh cells. Particularly, the clustering method uses the properties related to field homogeneity calculated on the original mesh.

This method is illustrated on a 2D magnetostatic calculation applied to an electromagnetic relay. The proposed methodology allows to pass from tens of thousands of DoF (6923 DoF for a solving with vector magnetic potential) to 25 clusters containing less than a hundred of interconnected components in the final reduced network (80 reluctances and 5 sources of non-zero mmf).

This illustration proves the relevance and effectiveness of this methodology for model reduction. However, some work remains to be done to fully validate this methodology. Further work will be required for quantifying the accuracy of the reduced model compared to the initial cochain model, in particular according to the number of clusters chosen. This reduction sensitivity analysis will allow a quantification of the robustness of the clustering methodology for design using reduced topological models.

Chapter 5

Conclusion and perspectives

This thesis concerns the development of modeling and computational tools for the design of multi-physical devices in the context of Model-Based System Engineering (MBSE). As explained in the first chapter, MBSE requires different models, at different levels, for different purposes: precise (or detailed) models for design, analysis and validation phases and reduced (or coarse) models for pre-design, optimization, control or real-time co-simulation phases. In this aim, the objective of this work was to propose a multi-scale model that can simplify the modeling and simulation of multi-physical devices. This multi-scale model should be suitable for both: solution analysis and validation phases (when computation of fields are required) and design, optimization, control synthesis (when computation of physical quantities in lumped parameter models, i.e. network of components, are required). Therefore, this work is focused on several standard or new approaches for modeling tasks in engineering. This work proposes also several extensions by Artificial Intelligence (IA) based clustering algorithms to provide innovative methods for modeling, simulation and model order reduction (MOR) in the framework of MBSE.

In the chapter 2, we propose to use geometric and topological tools (differential and algebraic topology) as well as duality theory (energetic and differential forms) and advanced discretization and meshing techniques (Delaunay-Voroinoi mesh and computer graphics tools) to achieve the objectives identified in Chapter 1. A review of these different tools and theory is conducted in order to propose feasible strategies for implementation in the following chapters. Analogies so useful in multi-physics modeling are widely used through Tonti's perspective and diagrams.

The chapter 3 is fully focused on the proposal of methods for fields computation based on techniques proposed in chapter 2. Corresponding methods require the use of dual meshes. Then a review on Delaunay-Voronoi meshes and remeshing techniques was first done in order to select softwares and methods able to compute orthogonal meshes well adapted for topological modeling of fields. The computation of field are obtained with a cochain method inspired by the Discrete External Calculus and Tonti's Cell-Method. The proposed field computation method is illustrated on a weakly coupled multi-physical MEMS device: "Electrokinetic $\rightarrow$ thermal $\rightarrow$ mechanical" field computations are performed. A

comparison between results of our cochain method and results computed by a standard Finite Elements Method (with ANSYS software) allows us to validate our approach and the results of our solver.

The main contribution of this work concerns a new concept of automatic generation of lumped parameter (LP) models from field computations. This contribution concerns the chapter 4. This one proposes an original approach which generates automatically LP models and applies at the same time a MOR. The two objectives, the automatic generation of LP models and MOR, are based on a topological analysis of the physical problem and clustering algorithms. Our methodology comprises several steps. In a first step we build a spatial mesh of the physical device (2D or 3D) while in the second step, we solve the physical equations via the cochain method proposed in chapter 3. As explained in this chapter, this method computes global physical quantities defined on nodes, edges, surfaces or volumes of the dual meshes. Thus, these results adopt the same nature and topology as the physical quantities used in networks of LP models. In this way, their compatibility during MOR is ensured. In the third step of our methodology, we then use fields data in unsupervised cluster analysis for identifying some homogeneous fields regions. The clustering is realized using DBSCAN algorithm. This algorithm was selected through a rigorous review of the literature on Artificial Intelligence, machine learning and clustering algorithms. Let us note that in our application, DBSCAN requires some additional data processing steps:

- Firstly, a computation of spatial distribution of field are required because DBSCAN requires information on densities. This computation is done using interpolation based on Whitney elements.

- Secondly, intermediate steps are included to reduce the number of cluster. This is done through a “waterfall" algorithm, a technique currently used in image compression.

As noted above, because the cochain method is a topological method, it is easy to aggregate all interior vertices, edges and volumes instances contained in larger regions identified by the clustering. For each clustered regions, this aggregation makes use of simple summations on instances contained in these larger regions to generate a component of the LP model. This procedure allows a drastic reduction of the number of DOF in the modeling. Then, the LP network is generated from interconnected larger regions using these clustered regions, results of fields computation and techniques from graph theory. MOR and automatic LP generation are illustrated on a magnetostatic example (an electromagnetic relay): H and B magnetic fields are calculated using the cochain method and a reduced reluctance network is automatically generated from the field results. In this example, the number of degrees of freedom is almost reduced by a factor of one hundred.

The results and work presented in this thesis are, however, only a first step in a broad research field. The use of the cochain method seems particularly suitable for conducting an automatic generation of LP models. However, as noted at the end

of chapter 3, the use of these methods in the field of mechanics is more complex as vector-valued differential forms must be computed (tensor of rank 2 in the framework of vector analysis). This requires the use of Whitney elements much earlier in the computational procedure. This is notably the case for computing stress and strain and making use of mechanical constitutive laws. In such a case, approximations based on Whitney Elements makes the method closer to a Finite Elements Exterior Calculus Method [Arnold et al., 2010] than to a true algebraic method where topology and metrics can be clearly separated.

As noted at the end of Chapter 4, for the time being, no careful assessment of error estimation or quantification has been undertaken, except at the end of the chapter 3 during a comparison with finite elements methods. However, it is obvious that this should be the next work conducted, especially to estimate the precision of the reduced models. Nevertheless, let us note that the formalism adopted in this thesis, i.e. Tonti's diagram and cochain/dual methods, lends itself well to these estimates because many methods of error bounding in finite elements are done by a double calculation, using primal and dual methods [Marmin et al., 2000a]. This double calculation is easily accomplished with the methodology proposed in this work. In particular, the work carried out around the constitutive relation error method [Ladevèze and Pelle, 2005] could be an interesting way for error quantification because, as in our work, it aims to ensure the exact topological laws and it allows errors to be transferred to the constitutive laws, i.e. on the Hodge star operators, the metric of the physical problem. With that in mind, a cooperation in the Roberval laboratory with the numerical mechanics team could be very profitable because some researchers of this team have a previous experience in the use of these methods. A collaboration with other French laboratories that have worked on these topics, in the field of electrical engineering, could also help us in this direction.

Being only the first thesis carried out in the laboratory on this subject, it is obviously perfectible and certain important aspects could not be treated as in-depth as we would have liked. In this sense, this work still requires many developments and many improvements, corrections or modifications. Without seeking to be exhaustive, and in addition to the work concerning the quantification of model errors, we can make an initial inventory of the main themes that it will be interesting to explore in future work.

- A first research topic could be the extension of the method for "real" coupled multiphysical problems since in chapter 3, we were limited to a weakly coupled problem. These extensions seem to be available at relatively short term, since Italian teams have already carried out a lot of work in this direction with the help of Cell-Method.

- Another interesting research topic could be the more advanced use of DBSCAN algorithm for clustering. Indeed, as we noted in chapter 4, this one can "merge" data in order to improve clustering. In our work, we only have used two informations (from $\mathbf{H}$ and $\mathbf{B}$ magnetic fields), to generate clusters that are representative of the underlying physics. To go further, one could imagine merging information of different nature, for example multiphysics,

in order to generate clusters that are representative of several physics at the same time. This can be an extremely interesting perspective in the context of the reduction of multiphysics models or re-meshing techniques. One could also consider making more intensive use of the machine learning capabilities of DBSCAN algorithms by using it on many examples of standard technological device calculations to derive model generation rules or network patterns to be used in design.

Exploiting the enormous potential of AI and Machine learning techniques is, in our opinion, extremely promising for MOR methods and features extraction of reduced model for MBSE. This use must however be careful to keep as much as possible a physical approach and a good understanding of the "rules" of segmentation and learning. This is in particular what we have tried to do in this work.

As we have just mentioned, there is still a lot to dig around this subject and this work represents only a first stone. This work can be improved but, in our opinion, it opens up many areas of research that the team can continue to explore internally but also with other local or external partners.

Appendix A

Algebraic versus differential formulations in scientific computing. Which are suitable ?

This appendix proposes some arguments to underline the interest and relevance of using global rather than local quantities for the numeric computation of fields. The arguments presented are mainly based on the reflections of E. Ferretti. These arguments seemed very convincing, which is why we have decided to include them in this appendix. Rather than paraphrasing E. Ferretti's arguments, which are very clearly explained in her articles and books, we thought it more appropriate to reproduce the most interesting parts of them in the following paragraphs. These arguments stress the advantages of an algebraic rather than a differential approach in scientific computing. The following quote is based on an argument proposed by [Ferretti, 2014]:

> In order to explain why the algebraic approach of the cell method (CM) is a winning strategy, if compared to that of the *classical* differential formulation, let's start with a brief excursus on the foundation of the differential calculus. Particular attention is devoted to the computation of limits, by highlighting how the numerical techniques used for performing limits may imply a loss of information. The equations using derivatives are called *differential equations* and a numerical formulation involving differential equations is called *differential formulation*. Meanwhile, the derivative is defined as the limit of a function. We can use the classical (ϵ, δ)-definition of limit. Let f be a function defined on an open interval containing c (except possibly at c) and let L be a real number. The statement $\lim_{x \to c} f(x) = L$, means that, for every real $\epsilon > 0$, there exists a real $\delta > 0$ such that, for all real x, if $0 < |x - c| < \delta$, then $|f(x) - L| < \epsilon$:
>
> $$\forall \epsilon > 0 \; \exists \; \delta > 0 : \forall x (0 < |x - c| < \delta \Rightarrow |f(x) - L| < \epsilon) \tag{A.1}$$
>
> The derivative $f'(x)$ of continuous function $f(x)$ can be defined as
>
> $$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}, \; h > 0 \tag{A.2}$$

The ratio in equation (A.2) is not continuous function at $h = 0$, because the limit has the indeterminate form $\{\frac{0}{0}\}$ as $h \to 0$. To avoid this we can use 1) an approximative way with subsequent iterations by reducing the error or 2) an exact way using the *cancellation rule* for limits.

As a reminder, the *cancellation rule* for limits states that, if the numerator and the denominator of a rational function share a common factor, then the new function obtained by algebraically canceling the common factor has limits identical to those of the original function. Here we give a simple example:

$$\lim_{x\to 3} \frac{x^2-9}{x-3} = \lim_{x\to 3} \frac{(x-3)(x+3)}{x-3} = \lim_{x\to 3} \frac{(x+3)}{1} = \frac{(3+3)}{1} = 6$$

In our case, we can write

$$\begin{aligned} &\frac{f(x+h)-f(x)}{h} = g(x), \quad \text{then} \\ &[f(x+h)-f(x)]|_{x=\overline{x}} = h \cdot g(h) \end{aligned} \tag{A.3}$$

In the perspective of a computational analysis using the differential formulation, it is obvious that the choice falls on the exact, rather than the approximated, computation of limits. In doing so, one can obtain an exact solution of the physical phenomenon under consideration only in few elementary cases, with simple geometric shapes of the domain and under particular boundary conditions. Anyway, the most important aspect is not that the exact numerical solution is hardly ever attained in real cases, but rather, that the choice itself of the term "exact" for the limit promised by the cancellation rule is *not entirely appropriate*. Actually, in order to provide the solution of the limit directly, the cancellation rule for limits reduces the order of zero both in the numerator and the denominator by one. Under the numerical point of view, this reduction is made by canceling a quantity with the order of a length, both in the numerator and in the denominator. Under the topological point of view, we could say that the reduction degrades the solution, in the sense that, being deprived of one length scale, the solution given by the cancellation rule provides us with a *lower degree of detail in describing the physical phenomenon under consideration*. In other words, we pay the direct solution of the cancellation rule by losing some kind of information on the solution itself. This is why we can say that the solution provided by the cancellation rule is not exact in a narrow sense, but only in a broad sens [Ferretti, 2014].

But what kind of information we actually lose? To answer this question we should recall a $\epsilon - \delta$ definition of the limit (A.2). This definition implies choosing an open interval, containing a **point** in which we can estimate a function. In other words, the limit on the left side in (A.3) is strictly bonded to the idea of interval of a point and cannot be separated from it. The result of the limit is the value to which the function output appears to approach as the computation

point approaches the estimation point. For evaluating this result, we must choose the computation points with care, in order to derive the trend of the output to a specific degree of approximation. But actually, if we see the right-side of (A.3), we can see a new function $g(x)$ that actually is computed at a point $h = 0$, without any need of evaluating its trend on an interval. The consequence is that the result we obtain is exact (in a broad sense), and we do not need to prefix any desired accuracy for the result itself. This is very useful from the numerical point of view, but, from the topological point of view, we lose information on what happens approaching the evaluation point. It is the same type of information we lose in passing from the description of a phenomenon in a space to the description of the same phenomenon in the tangent space at the evaluation point. The idea underlying in Cell-Method is that the cancellation rule can actually be employed only in those cases where the specific phenomenon uniquely depends on what happens at the point under consideration [Ferretti, 2014].

Studying the physical phenomenon as a function of all the points contained in a neighborhood means that we are using the left-hand side of (A.3), with h approaching zero but never equal to zero. This leads to point-wise variables in any cases, the line, surface, and volume densities. In the first case, we are facing a differential formulation, while, in the second case, we are facing an algebraic formulation where we can associated the global variables to **points**, **lines**, **surfaces** and **volumes** in a direct way. In fact, in classical approach we start from local variables and go to global. In the CM, we start directly from global [Ferretti, 2014].

Appendix B

Glossary of the Algebraic Topology

Topology is a branch of mathematics that deals with sets with a notion of neighborhood around each point, and is called topological spaces, as well as with the continuous applications between these spaces, which preserve this notion. Topology studies the properties of geometric figures that are preserved under continuous deformations, including stretching and bending. Since these kind of deformations are homeomorphisms, topology studies the properties of geometric figures that are invariant under homeomorphisms.

A topological space is a set of points, along with a set of neighborhoods for each point, that satisfy a set of axioms relating points and neighborhoods. The definition of a topological space relies only upon set theory. Manifolds and metric spaces are specializations of topological spaces with extra structures or constraints.

Definition 1. *In Euclidean space, an object is* ***convex*** *if, for every pair of points within the object, every point on the straight line segment that joins them is also within the object.*

Visually, the difference between convex and non-convex is presented on the figure B.1.

(a) A convex set

(b) A non-convex set, with a line-segment outside the set.

Figure B.1: Difference between convex and non-convex set.

As we use a sort of discretization and so we consider the a geometrical instance as a set of points, we need to introduce the following terms.

Definition 2. *The **convex hull** of a set X of points in the Euclidean plane or in a Euclidean space (or, more generally, in an affine space over the reals) is the smallest convex set that contains X. For instance, when X is a bounded subset of the plane, the convex hull may be visualized as the shape enclosed by a rubber band stretched around X [Andrew, 1979].*

In more formal terms, we can define a **convex hull** of a set X of points as:

1. the (unique) minimal convex set containing X,
2. the intersection of all convex sets containing X,
3. the set of all convex combinations of points in X, or the union of all simplices with vertices in X.

The example of a convex hull of the set of point is shown in the figure B.2.

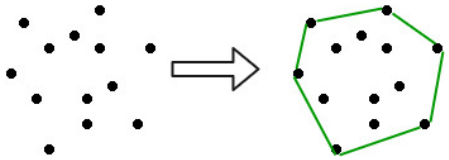


Figure B.2: Example of a convex hull (green shape).

Definition 3. ***Simplex** is the smallest convex set containing the given vertices.*

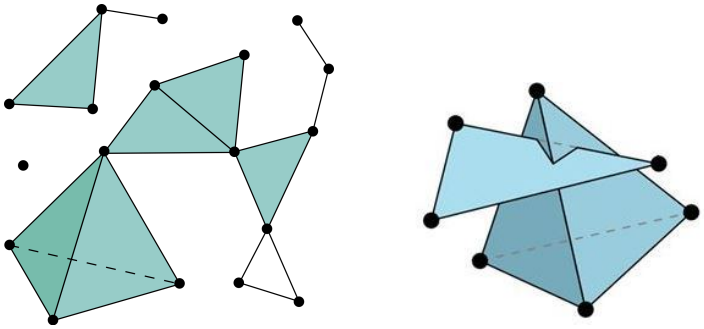
In other words, a **k-simplex** is a k-dimensional geometrical instance that is the convex hull of its $k+1$ vertices. For example, a 1-simplex is a straight line, a 2-simplex is a triangle and a 3-simplex is a tetrahedron.

Definition 4. *A **simplicial complex** $\mathcal{K}$ is a set of simplices that satisfies the following conditions:*

1. *Every face of a simplex from $\mathcal{K}$ is also in $\mathcal{K}$;*
2. *The non-empty intersection of any two simplices $\sigma_1, \sigma_2 \in \mathcal{K}$ is the face of both σ_1 and σ_2.*

In more simple way we can say that these spaces are built from simplices glued together in a combinatorial fashion. The example of mathematicly correct simplical complex is represented in the figure B.3(a) and invalid in the figure B.3(b) (improper intersection). But we need to add a restriction to the classical definition because we need to apply the definition of simplical complex to the mesh that we need in the simulation purposes. In our work, we are interested in only simplicial complex that have the following properties:

1. to be also a manifold;
2. we also require the simplicial complex to be *strongly regular*. This means that simplices must not have identifications on their boundaries. For example, edges are not allowed to begin and end in the same vertex;



(a) A simplical 3-complex.

(b) Invalid simplical complex

Figure B.3: Difference between valid and invalid representation of simplicial complex.

3. Embedding: each $k-1$-dimensional geometrical instance is belong to at least 2 k-dimensional geometrical instance (and at least 1 if border).

If we apply this additional restriction, we can see that the simplicial complex in the figure B.3(a) is no more a simplicial complex we desired.

In fact, it is rather numerically difficult to build a simplicial complex from set of point. For doing this we have used an approach proposed by Nathan Bell and Anil N. Hirani (see section 3.4 and [Bell and Hirani, 2012]). For the aim of simulation we will build our simplical complex from the initial mesh.

Definition 5. *The convex hull of any nonempty subset of the $n+1$ points that define an n-simplex is called a **face** of the simplex.*

Faces are simplices themselves. In particular, the convex hull of a subset of size $m+1$ (of the $n+1$ defining points) is an m-simplex, called an m-face of the n-simplex. The 0-faces (i.e., the defining points themselves as sets of size 1) are called the **vertices** (singular: vertex), the 1-faces are called the **edges**, the $(n-1)$-faces are called the **facets**. The set of faces of a geometric shape includes the geometric shape itself and the empty set, which for consistency may be defined as having dimension -1. Here we can see some examples:

1. The facets of a line segment are its 0-faces, or vertices;
2. The facets of a polygon are its 1-faces, or edges;
3. The facets of a polyhedron, or plane tiling, are its 2-faces;
4. The facets of a 4-polytope, or 3-honeycomb, are its 3-faces;
5. The facets of a 5-polytope, or 4-honeycomb, are its 4-faces.

Definition 6. *A simplex A is a **coface** of a simplex B if B is a face of A.*

The number of 1-faces (edges) of the n-simplex is the n-th triangle number, the number of 2-faces of the n-simplex is the (n-1)th tetrahedron number, the number of 3-faces of the n-simplex is the (n-2)th 5-cell number, and so on. Excluding from

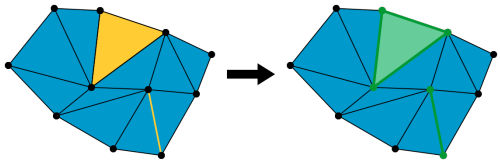
the list of the m-faces the empty set, the total number of faces of a k-simplex, $F(k)$, is always a power of two minus one: $F(k) = 2^{k+1} - 1$.

To define the last 3 definition, we consider a $\mathcal{K}$ to be a simplicial complex and let S be a collection of simplices in $\mathcal{K}$.

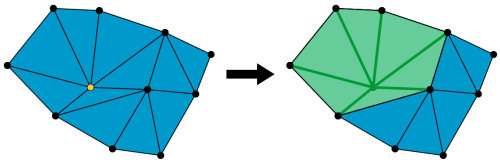
Definition 7. *The **closure** of S, denoted $cl(S)$, is the smallest simplicial complex containing S. The vertex set of the abstract simplicial complex $\mathcal{K}$ is the union of its elements [Attali et al., 2012]. See figure B.4(a).*

Definition 8. *The **star** of a simplex S in $\mathcal{K}$, denoted $St_{\mathcal{K}}(S)$, is the collection of simplices of $\mathcal{K}$ having S as a face [Attali et al., 2012]. See figure B.4(b).*

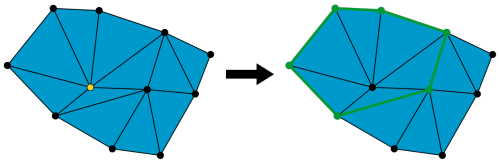
Definition 9. *The **link** of S in $\mathcal{K}$, denoted $Lk_{\mathcal{K}}(S)$, is a simplicial complex that equal the closed star of S minus the stars of all faces of S: $Lk_{\mathcal{K}}(S) = \{\tau \in \mathcal{K} | \tau \cup S \in \mathcal{K}, \tau \cap S = \emptyset\}$ [Attali et al., 2012]. See figure B.4(c).*



(a) Two simplices (yellow) and their closure (green).



(b) A vertex (yellow) and its star (green).



(c) A vertex (yellow) and its link (green).

Figure B.4: Basic topological operations: closure, star, link.

In the definition of the link, we can easily see the following relation:

$$Lk_{\mathcal{K}}(S) = Cl_{\mathcal{K}}(St_{\mathcal{K}}(S)) - St_{\mathcal{K}}(Cl_{\mathcal{K}}(S)) \tag{B.1}$$

Here, the important remark is given by *E. Feretti* in [Ferretti, 2014]:

> The space distribution of the point-wise field variables in the differential formulation, when discretized, gives rise to a discrete distribution of points that can be viewed as the 0-simplices of a simplicial cell-complex. On the contrary, the CM makes use of 3-simplices, or 4-simplices when also time is involved, allowing us to associate global variables with all the dimensions of the 4-cell complex. Using 4-cell complexes instead of 1-cell complex is the topological equivalent of avoiding to apply the *Cancelation Rule* for limits and is the main reason why the CM is able to take into account the length scales in computational Physics – until the third dimension – while the *differential formulation* does **not**.

B.1 Boundaries, Coboundaries and the Incidence Matrices

In this subsection we will define some more definition of algebraic topology for using as a tool for treating global variables. Here we will give only a brief definition. The complete and exact definition of these terms can be found in the homology theory [Dold, 2012].

Definition 10. *In abstract algebra, an abelian group, also called a commutative group, is a group in which the result of applying the group operation to two group elements does not depend on their order (the axiom of commutativity). Abelian groups generalize the arithmetic of the addition of integers.*

The basic idea of homology is to associate vector spaces to objects like graphs [Perrin, 2017]. Assume that $G = (V, E)$ is a graph, where V is a set whose elements are called vertices, and edges E is a set of two-sets (set with two distinct elements: each edge e has its origin $\alpha(e)$ and its end $\omega(e)$) of vertices.

Definition 11. *One defines the* ***boundary operator*** $\partial : \mathbb{Z}(E) \to \mathbb{Z}(V)$ *from the free abelian group on E to the free abelian group on V by $\partial(e) = \omega(e) - \alpha(e)$. (We say that a free abelian group G is the free abelain group of rank r if there is a linearly independent generating set with r elements). The elements of ∂ are called* ***boundaries****.*

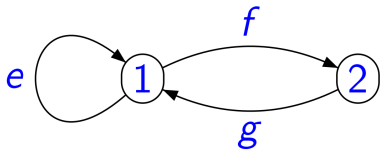


Figure B.5: The example of graph [Perrin, 2017].

For the graph in the figure B.5, $V = 1, 2$ and $E = e, f, g$. We have $\partial(e) = 0$, $\partial(f) = 2 - 1$, $\partial(g) = 1 - 2$. Thus the matrix of ∂ is

$$\begin{bmatrix} 0 & 0 \\ -1 & 1 \\ 1 & -1 \end{bmatrix} \tag{B.2}$$

By duality [Perrin, 2017], we have a coboundary operator δ:

Definition 12. *One defines the* ***coboundary operator*** $\delta = \partial^* = \partial^t : \mathbb{Z}^V \to \mathbb{Z}^E$ *defined for $\phi \in \mathbb{Z}^V$ by $\partial^t\phi(e) = \phi(\omega(e)) - \phi(\alpha(e))$.*

We can note that coboundary operator acts in the reverse way. Indeed, by definition of dual operator we can write

$$\partial^t\phi(e) = \phi(\partial(e)) = \phi(\omega(e) - \alpha(e)) = \phi(\omega(e)) - \phi(\alpha(e)) \tag{B.3}$$

In the same way, the coboundary operator for the graph in the figure B.5 is

$$\begin{bmatrix} 0 & -1 & 1 \\ 0 & 1 & -1 \end{bmatrix} \tag{B.4}$$

As we will choose for V a topological space X (often replaced by a finite family of subsets) and use the continuous dual $C(X, Z)$ instead of Z^V the matrix of dual operator ∂^t is just a transposed matrix of primal ∂.

To tie together previous definitions and boundary definition we can say that the set of faces of a p-cell defines the boundary of the p-cell. When the p-cell is a simplex, the boundary of the p-cell is its link. Also, the set of cofaces of a p-cell defines the coboundary of the p-cell.

B.2 Chains and Cochains Complexes, Boundary and Coboundary Processes

Chain complex and cochain complex are algebraic means of representing the relationships between the cycles and boundaries in various dimensions of a topological space.

In mathematics, a chain complex is an algebraic structure that consists of a sequence of abelian groups (or modules) and a sequence of homomorphisms between consecutive groups such that the image of each homomorphism is included in the kernel of the next. Associated to a chain complex is its homology, which describes how the images are included in the kernels.

Bibliography

[ARP, 2014] (2014). Arp-4754a, efficient avionics systems engineering with arp-4754a objective using scade system. Technical report, Hoboken, New Jersey.

[Alotto et al., 2011] Alotto, P., Desideri, D., Freschi, F., Maschio, A., and Repetto, M. (2011). Dual-PEEC modeling of a two-port TEM cell for VHF applications. *IEEE Transactions on Magnetics*, 47(5):1486–1489.

[Alotto et al., 2010] Alotto, P., Freschi, F., and Repetto, M. (2010). Multiphysics problems via the cell method: The role of tonti diagrams. *IEEE Transactions on Magnetics*, 46(8):2959–2962.

[Alotto et al., 2013] Alotto, P., Freschi, F., Repetto, M., and Rosso, C. (2013). *The Cell Method for Electrical Engineering and Multiphysics Problems: An Introduction*. Lecture Notes in Electrical Engineering. Springer Berlin Heidelberg.

[Andrew, 1979] Andrew, A. M. (1979). Another efficient algorithm for convex hulls in two dimensions. *Information Processing Letters*, 9(5):216–219.

[Arnold et al., 2010] Arnold, D., Falk, R., and Winther, R. (2010). Finite element exterior calculus: from hodge theory to numerical stability. *Bulletin of the American mathematical society*, 47(2):281–354.

[Arnold and Chen, 2017] Arnold, D. N. and Chen, H. (2017). Finite element exterior calculus for parabolic problems. *ESAIM: Mathematical Modelling and Numerical Analysis*, 51(1):17–34.

[Arnol'd, 1997] Arnol'd, V. (1997). *Mathematical Methods of Classical Mechanics*. Graduate Texts in Mathematics. Springer New York.

[Attali et al., 2012] Attali, D., Lieutier, A., and Salinas, D. (2012). Efficient data structure for representing and simplifying simplicial complexes in high dimensions. *International Journal of Computational Geometry & Applications*, 22(04):279–303.

[Awad and Khanna, 2015] Awad, M. and Khanna, R. (2015). *Efficient learning machines: theories, concepts, and applications for engineers and system designers*. Apress.

[Bamberg and Sternberg, 1991] Bamberg, P. and Sternberg, S. (1991). *A Course in Mathematics for Students of Physics:*. A Course in Mathematics for Students of Physics. Cambridge University Press.

[Bar-Sinai et al., 2019] Bar-Sinai, Y., Hoyer, S., Hickey, J., and Brenner, M. P. (2019). Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349.

[Beihoff et al., 2014] Beihoff, B., Friedenthal, S., Kemp, D., Nichols, D., Oster, C., Peredis, C., Stoewer, H., and Wade, J. (2014). A World In Motion: Systems Engineering Vision 2025. Technical report.

[Bell and Hirani, 2012] Bell, N. and Hirani, A. N. (2012). Pydec: software and algorithms for discretization of exterior calculus. *ACM Transactions on Mathematical Software (TOMS)*, 39(1):3.

[Ben-Chen, 2012] Ben-Chen, M. (2012). Courses cs468: Geometry processing algorithms of stanford university.

[Berge, 1973] Berge, C. (1973). *Graphs and hypergraphs*, volume 6 of *North-Holland mathematical library*. Elsevier.

[Beucher, 1994] Beucher, S. (1994). Watershed, hierarchical segmentation and waterfall algorithm. In *Mathematical morphology and its applications to image processing*, pages 69–76. Springer.

[Biedinger and Lemoine, 1997] Biedinger, J.-M. and Lemoine, D. (1997). Shape sensitivity analysis of magnetic forces. *Magnetics, IEEE Transaction on*, 33(3):2309–2316.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer Science+ Business Media.

[Boffi et al., 2013] Boffi, D., Brezzi, F., Fortin, M., et al. (2013). *Mixed finite element methods and applications*, volume 44. Springer.

[Bossavit, 1988] Bossavit, A. (1988). Whitney forms: A class of finite elements for three-dimensional computations in electromagnetism. *IEE Proceedings A-Physical Science, Measurement and Instrumentation, Management and Education-Reviews*, 135(8):493–500.

[Bossavit, 2012] Bossavit, A. (2012). The premetric approach to electromagnetism in the 'waves are not vectors' debate. *Advanced Electromagnetics*, 1(1).

[Branin, 1966] Branin, F. H. (1966). The algebraic topological basis for network analogies and the vector calculus. In *Symposium on Generalized Networks*.

[Brisset, 2007] Brisset, S. (2007). Démarches et outils pour la conception optimale des machines electriques (rapport d'habilitation à diriger des recherches). resreport, Université des sciences et techniques de Lille.

[Burke, 1985] Burke, W. (1985). *Applied Differential Geometry*. Cambridge University Press.

[Callen, 1960] Callen, H. B. (1960). *Thermodynamics*. John Wiley & Sons, Inc., New York, N.Y.

[Cannon, 2012] Cannon, R. H. J. (2012). *Dynamics of Physical Systems*. Dover Civil and Mechanical Engineering. Dover Publications.

[Ceron and Splendore, 2018] Ceron, A. and Splendore, S. (2018). From contents to comments: Social tv and perceived pluralism in political talk shows. *New Media & Society*, 20(2):659–675.

[Chio and Freeman, 2018] Chio, C. and Freeman, D. (2018). *Machine Learning and Security: Protecting Systems with Data and Algorithms*. " O'Reilly Media, Inc.".

[Chua and Lin, 1975] Chua, L. O. and Lin, P. M. (1975). *Computer-aided analysis of electronic circuits: algorithms and computational techniques*. Prentice-Hall series in electrical and computer engineering. Prentice-Hall.

[Cignoni et al., 2008] Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., and Ranzuglia, G. (2008). MeshLab: an Open-Source Mesh Processing Tool. In Scarano, V., Chiara, R. D., and Erra, U., editors, *Eurographics Italian Chapter Conference*. The Eurographics Association.

[Community of Wikipedia, 2018] Community of Wikipedia (2018). Delaunay triangulation — Wikipedia, the free encyclopedia. [Online; accessed 15 January 2019].

[Cook et al., 1989] Cook, R., Malkus, D., and Plesha, M. (1989). *Concepts and applications of finite element analysis, 3rd ed.* John Wiley & Sons.

[Cosmi, 2001] Cosmi, F. (2001). Numerical solution of plane elasticity problems with the cell method. *Computer Modeling in Engineering and sciences (CMES)*, 2(3):365–372.

[Crandall et al., 1968] Crandall, S., Karnopp, D. C., Kurtz, E. F. J., and Pridmore-Brown, D. C. (1968). *Dynamics of mechanical and electromechanical systems*. McGraw-Hill.

[Daumé III, 2012] Daumé III, H. (2012). A course in machine learning. *Publisher, ciml. info*, 5:69.

[Delaunay, 1934] Delaunay, B. (1934). Sur la sphere vide.

[Delprete et al., 2013] Delprete, C., Freschi, F., and Repetto, M. andRosso, C. (2013). A proposal of non-linear formulation of cell method for thermolastostatic problems. *Computer Modeling in Engineering and sciences (CMES)*, 94(5):397–420.

[Delprete et al., 2010] Delprete, C., Freschi, F., Repetto, M., and Rosso, C. (2010). Experimental validation of a numerical multiphysics technique for electro-thermo-mechanical problem. *COMPEL-The international journal for computation and mathematics in electrical and electronic engineering*, 29(6):1642–1652.

[Demkowicz et al., 2010] Demkowicz, L., Bramwell, J., and Qiu, W. (2010). Solution of dual-mixed elasticity equations using arnold-falk-winther element and discontinuous petrov-galerkin method, a comparison. In *Book of Abstracts*, page 6. Citeseer.

[Desbrun et al., 2005] Desbrun, M., Hirani, A. N., Leok, M., and Marsden, J. E. (2005). Discrete exterior calculus. *arXiv preprint math/0508341*.

[Desoer and Kuh, 1969a] Desoer, C. A. and Kuh, E. S. (1969a). *Basic Circuit Theory*. McGraw-Hill.

[Desoer and Kuh, 1969b] Desoer, C. A. and Kuh, E. S. (1969b). *Basic Circuit Theory*. McGraw-Hill.

[Dodziuk, 1976] Dodziuk, J. (1976). Finite-difference approach to the hodge theory of harmonic forms. *American Journal of Mathematics*, 98(1):79–104.

[Dold, 2012] Dold, A. (2012). *Lectures on algebraic topology*. Springer Science & Business Media.

[Dominik Moritz, 2018] Dominik Moritz (2018). Own work — Wikipedia, the free encyclopedia. [Online; accessed 6 October 2018].

[Du et al., 1999] Du, Q., Faber, V., and Gunzburger, M. (1999). Centroidal voronoi tessellations: Applications and algorithms. *SIAM review*, 41(4):637–676.

[Dular et al., 1998] Dular, P., Geuzaine, C., Henrotte, F., and Legros, W. (1998). A general environment for the treatment of discrete problems and its application to the finite element method. *IEEE Transactions on Magnetics*, 34(5):3395–3398.

[Ekman, 2003] Ekman, J. (2003). *Electromagnetic modeling using the partial element equivalent circuit method*. PhD thesis, Luleå tekniska universitet.

[Elcott and Schroder, 2006] Elcott, S. and Schroder, P. (2006). Building your own dec at home. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, pages 55–59, New York, NY, USA. ACM.

[Elcott et al., 2007] Elcott, S., Tong, Y., Kanso, E., Schröder, P., and Desbrun, M. (2007). Stable, circulation-preserving, simplicial fluids. *ACM Transactions on Graphics (TOG)*, 26(1):4.

[Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231.

[Ferretti, 2014] Ferretti, E. (2014). *The cell method: A purely algebraic computational method in physics and engineering*. Momentum Press.

[Feynman et al., 1965] Feynman, R. P., Leighton, R. B., and Sands, M. (1965). Lectures on physics, vol. 2. *Addison-Wesley Publishing Company, ODD*, 125:v2.

[Firestone, 1932] Firestone, F. A. (1932). A new analogy between mechanical and electrical systems. *Journal of the Acoustical Society of America*, vol. 4:pp. 249–267.

[Flanders, 1989] Flanders, H. (1989). *Differential Forms with Applications to the Physical Sciences*. Dover books on advanced mathematics. Dover Publications.

[Freschi et al., 2008] Freschi, F., Giaccone, L., and Repetto, M. (2008). Educational value of the algebraic numerical methods in electromagnetism. *COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, 27(NOVEMBER):1343–1357.

[Freschi et al., 2006] Freschi, F., Gruosso, G., and Repetto, M. (2006). Unstructured peec formulation by dual discretization. *IEEE Microwave and Wireless Components Letters*, 16(10):531–533.

[Freschi and Repetto, 2008] Freschi, F. and Repetto, M. (2008). A general framework for mixed structured/unstructured peec modelling. *Applied Computational Electromagnetics Society Journal*, 23(3):200–206.

[Fritzson, 2011] Fritzson, P. A. (2011). *Introduction to Modeling and Simulation of Technical and Physical Systems with Modelica*. IEEE Press, John Wiley & Sons, Inc., Hoboken, New Jersey.

[G. Dhatt, 1984] G. Dhatt, G. T. (1984). *Une Présentation de la méthode des éléments finis*. Editeur Maloine S.A., 2nd edition.

[Garimella et al., 2014] Garimella, R. V., Kim, J., and Berndt, M. (2014). Polyhedral mesh generation and optimization for non-manifold domains. In *Proceedings of the 22nd International Meshing Roundtable*, pages 313–330. Springer.

[George and Borouchaki, 1997] George, P. L. and Borouchaki, H. (1997). *Triangulation de Delaunay et maillage: applications aux éléments finis*. Hermes.

[Geuzaine and Remacle, 2009] Geuzaine, C. and Remacle, J.-F. (2009). Gmsh: a three-dimensional finite element mesh generator with built-in pre-and post-processing facilities. *International Journal for Numerical Methods in Engineering 79(11)*, 0:1309–1331.

[Hammond, 1981] Hammond, P. (1981). *Energy Methods In Electromagnetism*. Monographs in Electrical and Electronic Engineering. Oxford University Press.

[Hatcher, 2002] Hatcher, A. (2002). *Algebraic Topology*. Cambridge University Press.

[Hirani, 2003] Hirani, A. N. (2003). *Discrete Exterior Calculus*. PhD thesis, California Institute of Technology, Pasadena, California.

[Holmes et al., 2012] Holmes, P., Lumley, J. L., Berkooz, G., and Rowley, C. W. (2012). *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge university press.

[Hubert, 2002] Hubert, A. (2002). Modeling, simulation and design of a legged micro-robot. resreport, I.M.T., Technische Universität Braunschweig.

[Hughes, 2012] Hughes, T. J. R. (2012). *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Civil and Mechanical Engineering. Dover Publications.

[Jonathan Richard Shewchuk, 2018] Jonathan Richard Shewchuk (2018). Constrained delaunay tetrahedralizations bistellar flips, and provably good boundary recovery - support of course. [Online; accessed 15 January 2019].

[Karhunen, 1947] Karhunen, K. (1947). *Über lineare Methoden in der Wahrscheinlichkeitsrechnung*, volume 37. Sana.

[Karkulik et al., 2013] Karkulik, M., Of, G., and Praetorius, D. (2013). Convergence of adaptive 3d bem for weakly singular integral equations based on isotropic mesh-refinement. *Numerical methods for partial differential equations*, 29(6):2081–2106.

[Kasper, 2011] Kasper, M. (2011). *Mikrosystementwurf: Entwurf und Simulation von Mikrosystemen*. Springer Berlin Heidelberg.

[Kerschen et al., 2005] Kerschen, G., Golinval, J.-C., VAKAKIS, A., and BERGMAN, L. (2005). The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: An overview. *Nonlinear Dynamics*, 41:147–169.

[Kosambi, 1943] Kosambi, D. (1943). Statistics in function space. *Journal of Indian Mathematical Society 7*, pages 76–88.

[Kotiuga, 1984] Kotiuga, P. R. (1984). *Hodge decompositions and computational electromagnetics*. PhD thesis, McGill University, 845 Rue Sherbrooke Ouest, Montréal, QC H3A 0G4, Canada.

[Kron, 1939] Kron, G. (1939). *Tensor Analysis of Networks*. John Wiley & Sons.

[Kuci et al., 2017] Kuci, E., Henrotte, F., Duysinx, P., and Geuzaine, C. (2017). Design sensitivity analysis for shape optimization based on the lie derivative. *Computer Methods in Applied Mechanics and Engineering*, 317:702 – 722.

[Ladevèze and Pelle, 2005] Ladevèze, P. and Pelle, J.-P. (2005). *Mastering calculations in linear and nonlinear mechanics*, volume 171. Springer.

[Lanczos, 1986] Lanczos, C. (1986). *The Variational Principles of Mechanics*. Dover Books on Physics. Dover Publications.

[Latif et al., 2017] Latif, R., bin Jaafar, M. F., Majlis, B. Y., and Aqil, M. M. (2017). Estimation of thin film stress in buckled mems bridge from pull-in voltage. *2017 IEEE Regional Symposium on Micro and Nanoelectronics (RSM)*, pages 1–4.

[Le Menach et al., 1998] Le Menach, Y., Clénet, S., and Piriou, F. (1998). Determination and utilization of the source field in 3d magnetostatic problems. *IEEE Transactions on Magnetics*, 34(5):2509–2512.

[Lemaire, 1997] Lemaire, C. (1997). *Triangulation de Delaunay et arbres multidimensionnels*. PhD thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne; Université Jean ....

[Lévy and Schwindt, 2018] Lévy, B. and Schwindt, E. L. (2018). Notions of optimal transport theory and how to implement them on a computer. *Computers & Graphics*, 72:135–148.

[Lipnikov et al., 2014] Lipnikov, K., Manzini, G., and Shashkov, M. (2014). Mimetic finite difference method. *Journal of Computational Physics*, 257:1163–1227.

[Lloyd, 1982] Lloyd, S. (1982). Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137.

[Loeve, 1948] Loeve, M. (1948). Fonctions aléatoires du second ordre. appendix to p. lévy. *Processus stochastiques et mouvement Brownien*, pages 299–352.

[Lévy, 2014] Lévy, B. (2014). Restricted voronoi diagrams for (re)-meshing surfaces and volumes. In *Curves and Surfaces*.

[Marmin et al., 2000a] Marmin, F., Clenet, S., Bouillault, F., and Piriou, F. (2000a). Calculation of complementary solutions in 2d finite element method application to error estimation. *IEEE transactions on magnetics*, 36(4):1583–1587.

[Marmin et al., 2000b] Marmin, F., Clenet, S., Bouillault, F., and Piriou, F. (2000b). Calculation of complementary solutions in 2d finite element method application to error estimation. *IEEE Transactions on magnetics*, 36(4):pp 1583–1587.

[Mattiussi, 1997] Mattiussi, C. (1997). An analysis of finite volume, finite element, and finite difference methods using some concepts from algebraic topology. *Journal of Computational Physics*, 133(2):289–309.

[Mattiussi, 2002] Mattiussi, C. (2002). A reference discretization strategy for the numerical solution of physical field problems. *Advances in Imaging and Electron Physics*, 121:143–279.

[Meunier, 2010] Meunier, G. (2010). *The finite element method for electromagnetic modeling*, volume 33. John Wiley & Sons.

[Meyer et al., 2003] Meyer, M., Desbrun, M., Schröder, P., and Barr, A. H. (2003). Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and mathematics III*, pages 35–57. Springer.

[Monk et al., 2003] Monk, P. et al. (2003). *Finite element methods for Maxwell's equations*. Oxford University Press.

[Mullen et al., 2011] Mullen, P., Memari, P., de Goes, F., and Desbrun, M. (2011). Hot: Hodge-optimized triangulations. In *ACM Transactions on Graphics (TOG)*, volume 30, page 103. ACM.

[Munkers, 1984] Munkers, J. (1984). *Elements of Algebraic Topology*, volume 55. Addison–Wesley.

[Nedelec, 1980] Nedelec, J. C. (1980). Mixed finite elements in $\mathbb{R}^3$. *Numerische Mathematik*, 35(3):315–341.

[Nguyen et al., 2017] Nguyen, T.-S., Le Duc, T., Tran, T.-S., Guichon, J.-M., Chadebec, O., and Meunier, G. (2017). Adaptive multipoint model order reduction scheme for large-scale inductive peec circuits. *IEEE Transactions on Electromagnetic Compatibility*, 59(4):1143–1151.

[Nunes et al., 2017] Nunes, A. S., Chabedec, O., Kuo-Peng, P., Dular, P., and Meunier, G. (2017). A coupling between the facet finite element and reluctance network method in 3-d. *IEEE Transactions on Magnetics*, 53(10).

[Obukhov, 1954] Obukhov, A. (1954). Statistical description of continuous fields. *Transactions of the Geophysical International Academy Nauk USSR*, 24(24):3–42.

[Penfield and Haus, 1967] Penfield, P. and Haus, H. A. (1967). *Electrodynamics of Moving Media*. M.I.T. Press.

[Perot and Zusi, 2014] Perot, J. B. and Zusi, C. J. (2014). Differential forms for scientists and engineers. *Journal of Computational Physics*, 257:1373 – 1393. Physics-compatible numerical methods.

[Perrin, 2017] Perrin, D. (2017). *Words and Automata*. http://www-igm.univ-mlv.fr/ perrin/Enseignement/Master2017-18/Cours4/lecture4.pdf.

[Peterson et al., 1998] Peterson, A. F., Ray, S. L., Mittra, R., of Electrical, I., and Engineers, E. (1998). *Computational methods for electromagnetics*, volume 2. IEEE press New York.

[Preparata and Shamos, 1985] Preparata, F. and Shamos, M. (1985). Computational geometry springer-verlag. *New York*.

[Prof. Roberto Tamassia, 1993] Prof. Roberto Tamassia (1993). Cs252 - computational geometry - lecture 13 - support of course. [Online; accessed 15 January 2019].

[project by David Cournapeau, 2007] project by David Cournapeau, C. (2007). User's manual: scikit-learn. overview of clustering methods.

[Pugachev, 1953] Pugachev, V. S. (1953). The general theory of correlation of random functions. *Izvestiya Rossiiskoi Akademii Nauk. Seriya Matematicheskaya*, 17(5):401–420.

[Rajan, 1994] Rajan, V. (1994). Optimality of the delaunay triangulation in $\mathbb{R}^d$. *Discrete & Computational Geometry*, 12(2):189–202.

[Raviart and Thomas, 1977] Raviart, P. A. and Thomas, J. M. (1977). A mixed finite element for 2nd order elliptic problems, mathematical aspects of the finite element method. *Springer Lecture Notes in Mathematics*, 606:315–341.

[Recski, 1989] Recski, A. (1989). *Matroid theory and its applications in electric network theory and in statics*, volume 6 of *Algorithms and combinatorics*. Springer-Verlag.

[Rong and Cangellaris, 2001] Rong, A. and Cangellaris, A. C. (2001). Generalized peec models for three-dimensional interconnect structures and integrated passives of arbitrary shapes. In *IEEE 10th Topical Meeting on Electrical Performance of Electronic Packaging (Cat. No. 01TH8565)*, pages 225–228. IEEE.

[Roth, 1955a] Roth, J. P. (1955a). An application of algebraic topology to numerical analysis: on the existence of a solution to the network problem. *Proc. Nat. Aca. Sci.*, 41:518–521.

[Roth, 1955b] Roth, J. P. (1955b). The validity of kron's method of tearing. *Proc. Nat. Aca. Sci.*, 41:599–600.

[Rowell and Wormley, 1997] Rowell, D. and Wormley, D. N. (1997). *System dynamics: an introduction*. Prentice Hall.

[Ruehli, 1974] Ruehli, A. E. (1974). Equivalent circuit models for three-dimensional multiconductor systems. *IEEE Transactions on Microwave theory and techniques*, 22(3):216–221.

[Ruehli et al., 2003] Ruehli, A. E., Antonini, G., Esch, J., Ekman, J., Mayo, A., and Orlandi, A. (2003). Nonorthogonal peec formulation for time-and frequency-domain em and circuit modeling. *IEEE Transactions on Electromagnetic Compatibility*, 45(2):167–176.

[Sanchez et al., 2014] Sanchez, E. J., Paolini, C. P., and Castillo, J. E. (2014). The mimetic methods toolkit: an object-oriented api for mimetic finite differences. *Journal of Computational and Applied Mathematics*, 270:308–322.

[Sanogo et al., 2014] Sanogo, S., Messine, F., Hénaux, C., and Vilamot, R. (2014). Topology optimization for magnetic circuits dedicated to electric propulsion. *Magnetics, IEEE Transaction on*, 50(12).

[Schilders, 2008] Schilders, W. (2008). *Introduction to Model Order Reduction*, volume 13, pages 3–32.

[Segaran, 2007] Segaran, T. (2007). *Programming collective intelligence: building smart web 2.0 applications*. " O'Reilly Media, Inc.".

[Si, 2013] Si, H. (2013). *TetGen, towards a quality tetrahedral mesh generator*. WIAS.

[Su et al., 2014] Su, Z., Sun, J., Gu, X., Luo, F., and Yau, S.-T. (2014). Optimal mass transport for geometric modeling based on variational principles in convex geometry. *Engineering with Computers*, 30(4):475–486.

[Surazhsky et al., 2003] Surazhsky, V., Alliez, P., and Gotsman, C. (2003). *Isotropic remeshing of surfaces: a local parameterization approach*. PhD thesis, INRIA.

[Sweldens and Schröder, 2000] Sweldens, W. and Schröder, P. (2000). Building your own wavelets at home. In *Wavelets in the Geosciences*, pages 72–107. Springer.

[Taflove and Hagness, 2005] Taflove, A. and Hagness, S. C. (2005). *Computational electrodynamics: the finite-difference time-domain method*. Artech House, Norwood, 3rd edition.

[Tonti, 1976a] Tonti, E. (1976a). The reason for analogies between physical theories. *Applied Mathematical Modelling*, 1(1):37 – 50.

[Tonti, 1976b] Tonti, E. (1976b). The reason for analogies between physical theories. *Applied Mathematical Modelling*, 1(1):37–50.

[Tonti, 1976c] Tonti, E. (1976c). Sulla struttura formale delle teorie fisiche. *Rendiconti del Seminario Matematico e Fisico di Milano*, 46(1):163–257.

[Tonti, 2001a] Tonti, E. (2001a). A direct discrete formulation of field laws: The cell method. *CMES – Computer Modeling in Engineering & Sciences*, 2(2):pp. 237–258.

[Tonti, 2001b] Tonti, E. (2001b). Finite formulation of the electromagnetic field. *Progress in electromagnetics research*, 32:1–44.

[Tonti, 2013] Tonti, E. (2013). *The Mathematical Structure of Classical and Relativistic Physics: A General Classification Diagram*. Modeling and Simulation in Science, Engineering and Technology. Springer New York.

[Tonti and Zarantonello, 2009] Tonti, E. and Zarantonello, F. (2009). Algebraic formulation of elastostatics: the cell method. *Computer Modeling in Engineering and Sciences (CMES)*, 39(3):201.

[Trent, 1955] Trent, H. M. (1955). Isomorphisms between oriented linear graphs and lumped physical systems. *The Journal of the Acoustical Society of America*, vol. 27:pp. 500–527.

[Verite, 1987] Verite, J. (1987). Calculation of multivalued potentials in exterior regions. *IEEE Transactions on Magnetics*, 23:1881–1887.

[Villani, 2008] Villani, C. (2008). *Optimal transport: old and new*, volume 338. Springer Science & Business Media.

[Warnick and Russer, 2014] Warnick, K. F. and Russer, P. (2014). Differential forms and electromagnetic field theory. *Progress In Electromagnetics Research*, 148:3–112.

[Warren et al., 2007] Warren, J., Schaefer, S., Hirani, A. N., and Desbrun, M. (2007). Barycentric coordinates for convex sets. *Advances in Computational Mathematics*, 27(3):319–338.

[Weiland, 1977] Weiland, T. (1977). A discretization model for the solution of maxwell's equations for six-component fields. *Archiv Elektronik und Uebertragungstechnik*, 31:116–120.

[Weisstein, Eric W., 1999] Weisstein, Eric W. (1999). Delaunay triangulation. from mathworld–a wolfram web resource. [Online; accessed 15 January 2019].

[White and Woodson, 1959] White, D. C. and Woodson, H. H. (1959). *Electromechanical Energy Conversion*. John Wiley & Sons.

[Whitney, 1957] Whitney, H. (1957). *Geometric integration theory*. Princeton: Princeton University Press;.

[wikipedia. PEEC, 2018] wikipedia. PEEC (2018). Partial element equivalent circuit. page was last edited on 11 october 2018. [Online; accessed 27 October 2019].

[Yan et al., 2010] Yan, D. M., Wang, W., Lévy, B., and Liu, Y. (2010). Efficient computation of 3D clipped Voronoi diagram. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6130 LNCS:269–282.

[Yee, 1966a] Yee, K. (1966a). Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media. *IEEE Transactions on Antennas and Propagation*, 14:302–307.

[Yee, 1966b] Yee, K. (1966b). Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media. *IEEE Transactions on antennas and propagation*, 14(3):302–307.

Abstract

This PhD thesis deals with modeling and computational methods for the design of multi-physical devices in the context of Model-Based System Engineering (MBSE). The proposed methods are based on geometric and topological tools (differential and algebraic topology) as well as duality theory (energetic and differential forms) and advanced discretization and meshing techniques (Delaunay-Voroinoi mesh and computer graphics tools). Analogies so useful in multi-physics modeling are widely used in this work through Tonti's perspective and diagrams. Several standard approaches to engineering modelling are extended and complemented by machine learning and clustering algorithms to provide innovative methods for modeling, simulation and model order reduction.

The main contribution of this work concerns a new concept of automated generation of lumped parameter (LP) models from field computations. The results of the field calculations are obtained with a cochain method inspired by the Discrete External Calculus and the Tonti's Cell-Method. This thesis proposes an original approach which generates a LP model and applies at the same time a model order reduction. The two objectives, the automatic generation of the LP model and the model reduction, are based on a topological analysis of the physical problem and Artificial Intelligence clustering algorithms.

The final objective of this work is to propose a multi-scale model that can simplify the modeling and simulation of multi-physical devices. This multi-scale model should be suitable for both: solution analysis (computation of fields with the cochain model) and design, optimization, control synthesis (computation of physical quantities in a network of components with the LP model).

Field computations are illustrated on a weakly coupled multi-physical case: Electrokinetic, then thermal, and finally mechanical field are computed. Model order reduction and automatic LP generation are illustrated on a magnetostatic case (based on an electromagnetic device): the magnetic H and B fields are calculated using the cochain method, then a reduced reluctance network is automatically generated from the field results. In this example, the number of degrees of freedom is almost reduced by a factor of one hundred.

KEYWORDS:
Multiscale Modelling, Cochain Method, Lumped Parameters Model, Model Order Reduction, Clustering