NUMERICAL STUDY OF TWO DIMENSIONAL SMART STRUCTURES.

BY

DOMENICO VIGILANTE

A Thesis submitted to the Faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the Degree

Master of Science
in
Engineering Mechanics

Dr. Edmund G. Henneke

Dr. Romesh Batra

Dr. Don H.Morris

April 30, 2002
Blacksburg, Virginia

Keywords: Finite Element Method, Non Conforming Element, Piezoelectric Transducers, Piezo-electromechanical Coupling, Smart Structures, Vibrations Control.

NUMERICAL STUDY OF TWO DIMENSIONAL SMART STRUCTURES.

DOMENICO VIGILANTE

(ABSTRACT)

In this thesis we use a new numerical code, based upon a mixed FEM-Runge-Kutta method, for the analysis and the design of plane 2-dimensional smart structures. We applied the developed code to the study of arbitrarily shaped piezo-electromechanical (PEM) plates. This code is based on a weak formulation of their governing equations as found in [18]. The optimal parameters needed to synthesize the appropriate electric networks are computed, and the overall performances of such plates are investigated. In particular, two examples are studied: firstly, a simple case is used to test the main features of the code; secondly, a more complex PEM plate is designed and analyzed by means of the proposed numerical approach.

To my family

Contents

List of Figures **vii**

Chapter

1 Introduction **1**

Background and Motivations 1

Outline 4

I Galërkin formulation and software development **5**

2 Definition of smart structures and their mathematical model **6**

3 Galerkin formulation **10**

4 Element and shape functions **16**

5 Software description **21**

Data input 22

FEM analysis 24

Time evolution 25

Data post processing and output 26

The control settings 27

II An application of the FEM code: the piezo electromechanical plate. 29

6 Description of the system 30

7 Mathematical model 37

8 Numerical results 52

Test case: the simply supported square PEM plate 53

Uncoupled systems 54

Energy exchange in coupled system: the undamped case 55

Tuned coupled system: damping mechanical vibrations by means of a dissipative network 58

Second case: a complex geometry 60

Uncoupled systems 61

Energy exchange in coupled system: the undamped case 64

Coupled system: damping mechanical vibrations by means of a dissipative network 66

9 Concluding remarks 73

APPENDICES

A. The finite element method 75

Integral or *weak* statements equivalent to the differential equations 75

The Galerkin method of approximation 78

B. Some remarks on fourth order elements 81

C. Why did I use *Mathematica*? 83

D. Enhancing the code's capability 84

E. The main program 86

List of Figures

2.1 General representation of a 2-dimensional problem 7

4.1 Local area coordinates 18

5.1 Program structure 22

5.2 First block: data input 23

5.3 FEM analysis in the space domain 24

5.4 Third block: discrete time evolution 26

5.5 Results presentation 27

6.1 Electromechanical 2-dimensional continuum 33

6.2 2-dimensional electric network 34

6.3 Lumped approximation of an electromechanical continuum 35

8.1 Schematic of the system 52

8.2 Device parameters 53

8.3 Electric modal shapes 54

8.4 Mechanical modal shapes 55

8.5 Graphical representation of the electric and mechanical eigenfrequencies 56

8.6 Electric and mechanical eigenfrequencies: comparisons with the analytical results 57

8.7 Modal coupling table: black means maximum coupling, white means vanishing coupling 58

8.8 Energy time evolution for intial conditions on the a) first mechanical mode; b) second mechanical mode 59

8.9 Time evolution of the electric and mechanical displacements of a node of the mesh; initial conditions on the first mechanical mode. 60

8.10 Evolution from purely mechanical initial conditions (first mode) of the damped system: 1° row) optimal resistance value; 2° row) low resistance value. 61

8.11 Shape of the device 62

8.12 Frequency responses of the electric and mechanical uncoupled systems: 1° row) electric system tuned to the first mechanical eigenfrequency; 2° row) electrioc system tuned to the fourth mechanical eigenfrequencies. 63

8.13 Table of eigenfrequencies 64

8.14 Mechanical modal shapes 65

8.15 Electric modal shapes 66

8.16 Table of modal coupling 67

8.17 Time evolution from different uni-modal initial conditions. 68

8.18 Time evolution from impulsive initial conditions. 69

8.19 Time evolution from uni-modal initial conditions; fourth mode tuned. 70

8.20 Time evolution from impulsive initial conditions; fourth mode tuned. 70

8.21 Time evolution from first mode initial conditions; three different damping ratios. 71

8.22 Time evolution from uni-modal initial conditions: damped case . . 72

8.23 Time evolution from impulsive initial conditions: damped case . . 72

9.1 Domain description 76

Chapter 1

Introduction

Background and Motivations

A growing interest in the possibilities allowed in industrial applications by the conception of smart structures has arised in the past several years, due to the innovations introduced by material engineering in control techniques. There is now the possibility to design integrated systems in which the control subsystem is no longer based upon the paradigm of sensing-evaluating-actuating, but rather is embedded in the system: self-controlled structures, such as the piezo-electromechanical (PEM) beams and plates proposed in ([15], [16], [17], [18]), seem to show very interesting performances, as suggested by theoretical analysis. The design of this class of smart structures is directed to the passive, purely-electrical damping of mechanically excited structural vibrations. The main feature of the considered structures is that their evolution is governed by a system of PDEs where the unknown fields describe the electrical and mechanical state. The electrical state is not driven by an active control system and therefore no external "intelligence" is required in order to operate the mechanical vibration damping.

Obviously, the complexity of PEM structures does not allow for the determina-

tion of analytical or semi-analytical solutions of the governing PDEs in the general case. Hence, in order to supply a designing tool for experimental and industrial applications, a powerful numerical method for the analysis of such systems has to be developed.

In this thesis a mixed space-Finite-Element and time-Runge-Kutta numerical code realized by the author is described and applied to the analysis of the vibrational behavior of 2-dimensional PEM plates.

As the electrical and mechanical states of considered PEM plates evolve governed by second and fourth order space differential operators, respectively, the main difficulty in the code development consisted in the determination of a set of shape-functions suitable to represent both the aforementioned states. This difficulty has been confronted following the ideas described in [1] for the generic fourth order partial differential equation. The main trick is to build the Finite Elements starting from a *non-conforming* triangular element with three corner nodes. The resulting Finite Element approximation presents the following properties: i) as it passes the *patch test* it converges monotonically in the energy norm (convergence which needs to be proven case by case for non-conforming elements) ii) as it includes only three corner nodes it is immediately adaptable to represent approximate solutions of second order space PDEs.

The novelty of the proposed Finite Element method consists in the simultaneous solution of the studied coupled PDEs by means of the same mesh of elements.

The subsequent time integration presents great numerical difficulties. Indeed, using the Finite Elements projection, the governing system of PDEs is reduced to a system of time ODEs, the dimension of which can be huge. Therefore, in order to make the calculation possible or to save computational time, attention must be paid to the implementation of the Runge-Kutta algorithm. In order to reduce the dimension of the time ODEs system, its modal representation has been used and truncated at a proper number of electric and mechanical modes.

The numerical code provided is then used to study the electromechanical behavior of a specific PEM plate with complex geometry. In particular the design of such a plate is performed by means of an optimal parameters evaluation procedure using the developed code as a subroutine.

More precisely once the mechanical properties of the considered PEM plates are fixed, the proposed code allows for the determination of net inductances and net resistances which optimizes electrical damping of the mechanical energy of a specific mechanical mode. This is done by tuning the electrical network to the desired mechanical frequency (choosing the net inductance) and subsequently minimizing the mechanical damping ratio (choosing the net resistance).

It is finally shown that the specific designed device is able to damp mechanically induced vibrations in a specific range of frequencies.

Outline

This framework can be divided into three main parts:

1. A description of the software structure and features is provided and the weak formulation of a general input problem is presented, as well as all the separate stages of the numerical procedure.

2. The complete numerical analysis of two piezo-electric-mechanical systems is performed, and their parameters are optimized. In this way a first look upon the use of the numerical tool, both in the analysis and in the project of smart structures, can be given. The weak formulation of such systems is provided, and all the data input needed by the program are carried out; then two practical applications are considered: firstly, a simply supported square PEM plate (whose analytical solution has been provided in [18]) is used as a test bench for the computational algorithm; secondly, a more complex geometry (with no available analytical solutions) is deeply investigated, and some project parameters are obtained;

3. A complete print out of the main routine of the code is provided, with some short descriptions of the different block functions.

Part I

Galërkin formulation and software development

Chapter 2

Definition of smart structures and their mathematical model

Many definitions have been given to the concept of *smart structures*; indeed, the recent and fast growth of this technological area does not allows for an accurate and definitive characterization of its limits. Nevertheless, we need to establish some precise objects to which the numerical procedure we are going to develop will be applied. Therefore we will derive a general mathematical model which represents all the systems we want to study, starting from the following definition, which is one of the most pertinent to our analysis:

Definition 1 *A smart structure is a material system with intelligence and life features integrated in its micro-strucure to reduce mass and energy and produce adaptive functionality.*

From this definition, we conclude that the smart strucures we are going to consider are constituted by the continuous coupling (at the micro level) of two or more systems. Therefore a continuum model can be applied, with proper micro-macro identification procedures, to these structures. Finally, the mathematical

model for such smart structures will be given, in a strong formulation, as a set of coupled differential equations. More precisely, we will assume a certain number of systems governed by fourth order PDEs and second order PDEs coupled in different ways. Since we want to apply a numerical approximation procedure to our problem, we must state it in an integral form. Therefore, here we provide the general weak formulation which describes the behavior of the considered smart structures.

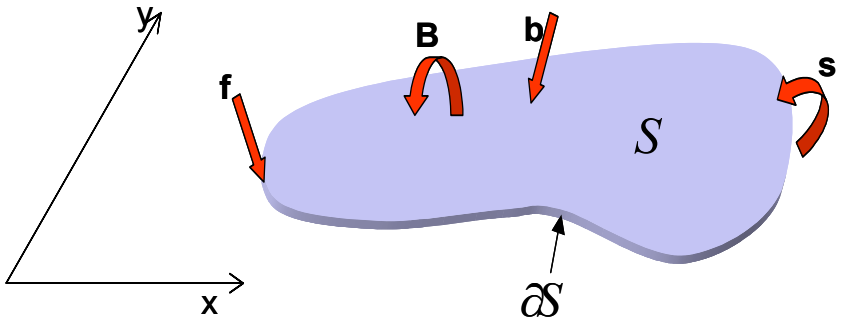

Figure 2.1: General representation of a 2-dimensional problem

Let $\mathcal{S}$ be a 2-dimensional domain, and let $\partial\mathcal{S}$ be its boundary. Referring to Fig. (**??**), we can consider the following 2-dimensional mathematical model, based upon a virtual work principle:

Virtual work principle (must hold for any compatible test field $(\mathbf{u}^t, \boldsymbol{\varepsilon}^t)$ *):*

$$\left\langle \mathbf{b},\, \dot{\mathbf{u}}^t \right\rangle_{\mathcal{S}} + \left\langle \mathbf{B}_1, \dot{\mathbf{u}}^t_{,1} \right\rangle_{\mathcal{S}} + \left\langle \mathbf{B}_2, \dot{\mathbf{u}}^t_{,2} \right\rangle +$$

$$\left\langle \mathbf{f}, \dot{\mathbf{u}}^t \right\rangle_{\partial\mathcal{S}} + \left\langle \mathbf{s}_1, \dot{\mathbf{u}}^t_{,1} \right\rangle_{\partial\mathcal{S}} + \left\langle \mathbf{s}_2, \dot{\mathbf{u}}^t_{,2} \right\rangle_{\partial\mathcal{S}} = \left\langle \boldsymbol{\sigma}, \dot{\boldsymbol{\varepsilon}}^t \right\rangle_{\mathcal{S}}$$

Constitutive relations and external-inertia forces

$$\begin{aligned}
&\mathbf{b} = \mathbf{G}\,\ddot{\mathbf{u}} + \mathbf{S}\,\dot{\mathbf{u}} + \mathbf{T}\,\mathbf{u} + \mathbf{V}\,\dot{\boldsymbol{\varepsilon}} + \mathbf{b}_0 \\
&\mathbf{B}_1 = \mathbf{G}_B^1\,\ddot{\mathbf{u}}_{,1} + \mathbf{B}_1^0\,, \qquad \mathbf{B}_2 = \mathbf{G}_B^2\,\ddot{\mathbf{u}}_{,2} + \mathbf{B}_2^0 \\
&\mathbf{f} = \mathbf{f}_0 \\
&\mathbf{s}_1 = \mathbf{s}_1^0\,, \qquad \mathbf{s}_2 = \mathbf{s}_2^0 \\
&\boldsymbol{\sigma} = \mathbf{E}\,\boldsymbol{\varepsilon} + \mathbf{C}\,\dot{\mathbf{u}} + \mathbf{R}\,\mathbf{u} + \boldsymbol{\sigma}_0
\end{aligned} \tag{2.1}$$

Kinematical compatibility

$$\begin{aligned}
\boldsymbol{\varepsilon} &= \mathbf{H}_0\mathbf{u} + \mathbf{H}_1\mathbf{u}_{,1} + \mathbf{H}_2\mathbf{u}_{,2} + \mathbf{H}_3\mathbf{u}_{,11} + \\
&\quad \mathbf{H}_4\mathbf{u}_{,22} + \mathbf{H}_5\mathbf{u}_{,12} \\
\boldsymbol{\varepsilon}^t &= \mathbf{H}_0^t\mathbf{u}^t + \mathbf{H}_1^t\mathbf{u}_{,1}^t + \mathbf{H}_2^t\mathbf{u}_{,2}^t + \mathbf{H}_3^t\mathbf{u}_{,11}^t + \\
&\quad \mathbf{H}_4^t\mathbf{u}_{,22}^t + \mathbf{H}_5^t\mathbf{u}_{,12}^t
\end{aligned}$$

Here the notation $\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{S}}$ holds for $\int_{\mathcal{S}} \mathbf{x}^T\,\mathbf{y}\,d\mathcal{S}$, $\mathbf{u}$ is the vector of kinematical descriptors of the system, $\boldsymbol{\varepsilon}$ the deformation field, $\mathbf{b}$, $\mathbf{B}_1$, $\mathbf{B}_2$, $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{f}$ the generalized body and surface forces, $\boldsymbol{\sigma}$ is the internal stress field. Moreover, $\dot{x}$ holds for the time derivative of x, while $x_{,i}$ holds for the space derivative of x in the ith-direction on the plane. The suffix t indicates that we deal with a trial field. The domain $\mathcal{S}$ is obviously a 2-dimensional region.

We observe that with this model, joined with proper boundary and initial conditions, we can describe in a weak manner all the 2-dimensional systems gov-

erned by second and fourth order differential equations, and that we can have any number of them coupled in different ways.

From now on, we will assume this model to represent all the cases of interest for our analysis.

Chapter 3

Galerkin formulation

Now we show how to obtain an ordinary time differential system of equations from the proposed integral model.

We begin by substituting the constitutive relations in the virtual work principle:

$$\left\langle \mathbf{G}\,\ddot{\mathbf{u}} + \mathbf{S}\,\dot{\mathbf{u}} + \mathbf{T}\,\mathbf{u} + \mathbf{V}\,\dot{\boldsymbol{\varepsilon}} + \mathbf{b}_0,\ \dot{\mathbf{u}}^t \right\rangle_{\mathcal{S}} + \left\langle \mathbf{G}_B^1\,\ddot{\mathbf{u}}_{,1} + \mathbf{B}_1^0, \dot{\mathbf{u}}_{,1}^t \right\rangle_{\mathcal{S}} + \left\langle \mathbf{G}_B^2\,\ddot{\mathbf{u}}_{,2} + \mathbf{B}_2^0, \dot{\mathbf{u}}_{,2}^t \right\rangle +$$

$$\left\langle \mathbf{f}_0, \dot{\mathbf{u}}^t \right\rangle_{\partial\mathcal{S}} + \left\langle \mathbf{s}_1^0, \dot{\mathbf{u}}_{,1}^t \right\rangle_{\partial\mathcal{S}} + \left\langle \mathbf{s}_2^0, \dot{\mathbf{u}}_{,2}^t \right\rangle_{\partial\mathcal{S}} - \left\langle \mathbf{E}\,\boldsymbol{\varepsilon} + \mathbf{C}\,\dot{\mathbf{u}} + \mathbf{R}\,\mathbf{u} + \boldsymbol{\sigma}_0, \dot{\boldsymbol{\varepsilon}}^t \right\rangle_{\mathcal{S}} = 0\,.$$

Following the steps of a standard Finite Elements procedure, we subdivide the domain $\mathcal{S}$ into subdomains $\mathcal{D}^e$ that cover $\mathcal{S}$ without superpositions (i.e. $\mathcal{D}^1 \cup \mathcal{D}^2 \cup \ldots \cup \mathcal{D}^{ne} = \mathcal{S}$ and $\mathcal{D}^i \cap \mathcal{D}^j = \emptyset$ for $i \neq j$). Hence the integrals in the previous equation can be expressed as summations of integrals computed on each element $\mathcal{D}^e$.

$$\sum_e \left\langle \mathbf{G}\,\ddot{\mathbf{u}} + \mathbf{S}\,\dot{\mathbf{u}} + \mathbf{T}\,\mathbf{u} + \mathbf{V}\,\dot{\boldsymbol{\varepsilon}} + \mathbf{b}_0,\ \dot{\mathbf{u}}^t \right\rangle_{\mathcal{D}^e} + \sum_e \left\langle \mathbf{G}_B^1\,\ddot{\mathbf{u}}_{,1} + \mathbf{B}_1^0, \dot{\mathbf{u}}_{,1}^t \right\rangle_{\mathcal{D}^e} +$$

$$\sum_e \left\langle \mathbf{G}_B^2\,\ddot{\mathbf{u}}_{,2} + \mathbf{B}_2^0, \dot{\mathbf{u}}_{,2}^t \right\rangle + \sum_e \left\langle \mathbf{f}_0, \dot{\mathbf{u}}^t \right\rangle_{\partial\mathcal{D}^e \cap \partial\mathcal{S}} + \sum_e \left\langle \mathbf{s}_1^0, \dot{\mathbf{u}}_{,1}^t \right\rangle_{\partial\mathcal{D}^e \cap \partial\mathcal{S}} +$$

$$\sum_e \left\langle \mathbf{s}_2^0, \dot{\mathbf{u}}_{,2}^t \right\rangle_{\partial\mathcal{D}^e \cap \partial\mathcal{S}} - \sum_e \left\langle \mathbf{E}\,\boldsymbol{\varepsilon} + \mathbf{C}\,\dot{\mathbf{u}} + \mathbf{R}\,\mathbf{u} + \boldsymbol{\sigma}_0, \dot{\boldsymbol{\varepsilon}}^t \right\rangle_{\mathcal{D}^e} = 0\,.$$

Next we consider a single element. The displacement vector can be locally expressed in terms of bases functions defined only in the element:

$$\mathbf{u}^e(\mathbf{r},t) = \mathbf{N}^e(\mathbf{r})\,\mathbf{q}^e(t) \tag{3.1}$$

where $\mathbf{N}^e(\mathbf{r})$ is the shape functions matrix, while $\mathbf{q}(t)$ is the vector of the nodal values of $\mathbf{u}^e(\mathbf{r},t)$. To gain a more compact expression, we define

$$\boldsymbol{\varepsilon}^e = \mathbf{N}^e_\varepsilon\,\mathbf{q}^e\,, \tag{3.2}$$

$$\boldsymbol{\varepsilon}^{et} = \mathbf{N}^{et}_\varepsilon\,\mathbf{q}^{et}\,, \tag{3.3}$$

where

$$\mathbf{N}^e_\varepsilon = \mathbf{H}_0\mathbf{N}^e + \mathbf{H}_1\mathbf{N}^e_{,1} + \mathbf{H}_2\mathbf{N}^e_{,2} + \mathbf{H}_3\mathbf{N}^e_{,11} + \mathbf{H}_4\mathbf{N}^e_{,22} + \mathbf{H}_5\mathbf{N}^e_{,12} \tag{3.4}$$

$$\mathbf{N}^{et}_\varepsilon = \mathbf{H}^t_0\mathbf{N}^e + \mathbf{H}^t_1\mathbf{N}^e_{,1} + \mathbf{H}^t_2\mathbf{N}^e_{,2} + \mathbf{H}^t_3\mathbf{N}^e_{,11} + \mathbf{H}^t_4\mathbf{N}^e_{,22} + \mathbf{H}^t_5\mathbf{N}^e_{,12} \tag{3.5}$$

Therefore, if we consider the contribution of a single element to the integral formulation we get:

$$\begin{aligned}
&\ddot{\mathbf{q}}^{eT}\,\langle\mathbf{G}\,\mathbf{N}^e,\,\mathbf{N}^e\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} + \dot{\mathbf{q}}^{eT}\,\langle\mathbf{S}\,\mathbf{N}^e,\,\mathbf{N}^e\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} + \mathbf{q}^{eT}\,\langle\mathbf{T}\,\mathbf{N}^e,\mathbf{N}^e\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} + \\
&\dot{\mathbf{q}}^{eT}\,\langle\mathbf{V}\,\mathbf{N}^e_\varepsilon\,,\,\mathbf{N}^e\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} + \langle\mathbf{b}_0,\mathbf{N}^{\mathbf{e}}\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} + \ddot{\mathbf{q}}^{eT}\,\left\langle\mathbf{G}^1_B\,\mathbf{N}^e_{,1}\,,\,\mathbf{N}^e_{,1}\right\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} + \\
&\ddot{\mathbf{q}}^{eT}\,\left\langle\mathbf{G}^2_B\,\mathbf{N}^e_{,2}\,,\,\mathbf{N}^e_{,2}\right\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} + \left\langle\mathbf{B}^0_1,\mathbf{N}^e_{,1}\right\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} + \left\langle\mathbf{B}^0_2,\,\mathbf{N}^e_{,2}\right\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} + \\
&\langle\mathbf{f}_0,\,\mathbf{N}^e\rangle_{\partial\mathcal{D}^e\cap\partial\mathcal{S}}\,\dot{\mathbf{q}}^{et} + \left\langle\mathbf{s}^0_1,\,\mathbf{N}^e_{,1}\right\rangle_{\partial\mathcal{D}^e\cap\partial\mathcal{S}}\,\dot{\mathbf{q}}^{et} + \left\langle\mathbf{s}^0_2,\,\mathbf{N}^e_{,2}\right\rangle_{\partial\mathcal{D}^e\cap\partial\mathcal{S}}\,\dot{\mathbf{q}}^{et} + \\
&-\mathbf{q}^{eT}\left\langle\mathbf{E}\,\mathbf{N}^e_\varepsilon\,,\mathbf{N}^{et}_\varepsilon\right\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} - \dot{\mathbf{q}}^{eT}\left\langle\mathbf{C}\,\mathbf{N}^e,\mathbf{N}^{et}_\varepsilon\right\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} - \\
&\mathbf{q}^{eT}\left\langle\mathbf{R}\,\mathbf{N}^e,\mathbf{N}^{et}_\varepsilon\right\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et} - \left\langle\boldsymbol{\sigma}_0,\mathbf{N}^{et}_\varepsilon\right\rangle_{\mathcal{D}^e}\,\dot{\mathbf{q}}^{et}
\end{aligned}$$

where the suffix T holds for $Transpose$. Grouping the terms we get:

$$\ddot{\mathbf{q}}^{eT}(\langle \mathbf{G}\,\mathbf{N}^e,\,\mathbf{N}^e\rangle_{\mathcal{D}^e} + \left\langle \mathbf{G}_B^1\,\mathbf{N}_{,1}^e\,,\,\mathbf{N}_{,1}^e\right\rangle_{\mathcal{D}^e} + \left\langle \mathbf{G}_B^2\,\mathbf{N}_{,2}^e\,,\,\mathbf{N}_{,2}^e\right\rangle_{\mathcal{D}^e})\dot{\mathbf{q}}^{et} +$$

$$\dot{\mathbf{q}}^{eT}(\langle \mathbf{S}\,\mathbf{N}^e,\,\mathbf{N}^e\rangle_{\mathcal{D}^e} + \langle \mathbf{V}\,\mathbf{N}_\varepsilon^e\,,\,\mathbf{N}^e\rangle_{\mathcal{D}^e} - \left\langle \mathbf{C}\,\mathbf{N}^e,\mathbf{N}_\varepsilon^{et}\right\rangle_{\mathcal{D}^e})\dot{\mathbf{q}}^{et} +$$

$$\mathbf{q}^{eT}(\langle \mathbf{T}\,\mathbf{N}^e,\mathbf{N}^e\rangle_{\mathcal{D}^e} - \left\langle \mathbf{E}\,\mathbf{N}_\varepsilon^e\,,\mathbf{N}_\varepsilon^{et}\right\rangle_{\mathcal{D}^e} - \left\langle \mathbf{R}\,\mathbf{N}^e,\mathbf{N}_\varepsilon^{et}\right\rangle_{\mathcal{D}^e})\dot{\mathbf{q}}^{et} +$$

$$(\langle \mathbf{b}_0,\mathbf{N}^{\mathbf{e}}\rangle_{\mathcal{D}^e} + \left\langle \mathbf{B}_1^0,\mathbf{N}_{,1}^e\right\rangle_{\mathcal{D}^e} + \left\langle \mathbf{B}_2^0,\,\mathbf{N}_{,2}^e\right\rangle_{\mathcal{D}^e} + \langle \mathbf{f}_0,\,\mathbf{N}^e\rangle_{\partial\mathcal{D}^e\cap\partial\mathcal{S}} +$$

$$\left\langle \mathbf{s}_1^0,\,\mathbf{N}_{,1}^e\right\rangle_{\partial\mathcal{D}^e\cap\partial\mathcal{S}} + \left\langle \mathbf{s}_2^0,\,\mathbf{N}_{,2}^e\right\rangle_{\partial\mathcal{D}^e\cap\partial\mathcal{S}} - \left\langle \boldsymbol{\sigma}_0,\mathbf{N}_\varepsilon^{et}\right\rangle_{\mathcal{D}^e})\dot{\mathbf{q}}^{et}$$

Hence we can define:

Local mass matrix	$\mathbf{K}_0^e = \langle \mathbf{G}\,\mathbf{N}^e,\,\mathbf{N}^e\rangle_{\mathcal{D}^e} + \left\langle \mathbf{G}_B^1\,\mathbf{N}_{,1}^e\,,\,\mathbf{N}_{,1}^e\right\rangle_{\mathcal{D}^e} + \left\langle \mathbf{G}_B^2\,\mathbf{N}_{,2}^e\,,\,\mathbf{N}_{,2}^e\right\rangle_{\mathcal{D}^e}$
Local damping/coupling matrix	$\mathbf{K}_1^e = \langle \mathbf{S}\,\mathbf{N}^e,\,\mathbf{N}^e\rangle_{\mathcal{D}^e} + \langle \mathbf{V}\,\mathbf{N}_\varepsilon^e\,,\,\mathbf{N}^e\rangle_{\mathcal{D}^e} - \left\langle \mathbf{C}\,\mathbf{N}^e,\mathbf{N}_\varepsilon^{et}\right\rangle_{\mathcal{D}^e}$
Local stiffness matrix	$\mathbf{K}_2^e = \langle \mathbf{T}\,\mathbf{N}^e,\mathbf{N}^e\rangle_{\mathcal{D}^e} - \left\langle \mathbf{E}\,\mathbf{N}_\varepsilon^e\,,\mathbf{N}_\varepsilon^{et}\right\rangle_{\mathcal{D}^e} - \left\langle \mathbf{R}\,\mathbf{N}^e,\mathbf{N}_\varepsilon^{et}\right\rangle_{\mathcal{D}^e}$
Local load vector	$\mathbf{F}^e = -\langle \mathbf{b}_0,\mathbf{N}^{\mathbf{e}}\rangle_{\mathcal{D}^e} - \left\langle \mathbf{B}_1^0,\mathbf{N}_{,1}^e\right\rangle_{\mathcal{D}^e} - \left\langle \mathbf{B}_2^0,\,\mathbf{N}_{,2}^e\right\rangle_{\mathcal{D}^e} - \langle \mathbf{f}_0,\,\mathbf{N}^e\rangle_{\partial\mathcal{D}^e\cap\partial\mathcal{S}} - \left\langle \mathbf{s}_1^0,\,\mathbf{N}_{,1}^e\right\rangle_{\partial\mathcal{D}^e\cap\partial\mathcal{S}} - \left\langle \mathbf{s}_2^0,\,\mathbf{N}_{,2}^e\right\rangle_{\partial\mathcal{D}^e\cap\partial\mathcal{S}} + \left\langle \boldsymbol{\sigma}_0,\mathbf{N}_\varepsilon^{et}\right\rangle_{\mathcal{D}^e}$

(3.6)

From a computational point of view, this is probably the heaviest stage, since we have to carry out all these integrals for each element of the mesh; as a consequence, we pay great attention to the optimization of the calculus algorithm for

this phase.

The next step is to assemble the local matrices and vector to get the final system. Following standard finite element procedures, based upon the correspondence between local and global node numbering, we finally obtain:

$$\mathbf{K}_2\,\ddot{\mathbf{q}}(t) + \mathbf{K}_1\,\dot{\mathbf{q}}(t) + \mathbf{K}_0\,\mathbf{q}(t) = \mathbf{F}(t) \tag{3.7}$$

where $\mathbf{K}_2$, $\mathbf{K}_1$, $\mathbf{K}_0$ are, respectively, the global mass, damping/coupling and stiffness matrices, $\mathbf{F}$ is the global load vector and $\mathbf{q}$ is the vector of nodal generalized displacements.

Note that the dimensions of this system of equations can be very large; a quite refined mesh of the domain can easily reach thousands of nodes, so that a numerical analysis of such a system would be too heavy. Therefore a different approach is applied. First we perform a modal analysis of the system, assuming $\mathbf{K}_1$ to be small compared to $\mathbf{K}_0$ and $\mathbf{K}_2$; then we project all the matrices involved over a base of a suitable number of eigenvectors. In this way we get a reduced system of much smaller dimensions. Obviously, also the initial conditions have to be projected on the chosen base, losing something in terms of accuracy in reproducing Dirac-delta like functions, but gaining much more in terms of computational time. This procedure can be illustrated as follows. We define

$$\mathbf{q}(t) = \mathbf{a}\,e^{j\omega t},$$

so that the homogeneous problem reads as follows:

$$-\omega^2 \mathbf{K}_2 \mathbf{a} + \mathbf{K}_0 \mathbf{a} = \mathbf{0},$$

which, with some computation, becomes:

$$\mathbf{K}_0^{-1} \mathbf{K}_2 \mathbf{a} = \frac{1}{\omega^2} \mathbf{a}$$

Hence the eigensystem of $\mathbf{K}_0^{-1} \mathbf{K}_2$ will give us eigenvectors and eigenfrequency of the uncoupled/undamped system. Hence, we define $\mathbf{a}_i$ to be the i-th eigenvector, and choose a number n of them to build up an orthonormal base. Now we define:

$$\mathbf{T} = \begin{bmatrix} \mathbf{a}_{11} & \mathbf{a}_{12} & \dots & \mathbf{a}_{1m} \\ \mathbf{a}_{21} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \mathbf{a}_{n1} & \mathbf{a}_{n2} & \dots & \mathbf{a}_{nm} \end{bmatrix},$$

where m is the dimension of the initial system of equations. We then consider the projection of the nodal displacement vector $\mathbf{q}$ on this base:

$$\begin{aligned} \mathbf{z} &= \mathbf{T}\,\mathbf{q}, \\ \mathbf{q} &= \mathbf{T}^T \mathbf{z}. \end{aligned} \tag{3.8}$$

Therefore, substituting this expression in eq. (3.7) and multiplying on the left side by $\mathbf{T}$, we get:

$$\mathbf{K}_{2red}\, \ddot{\mathbf{z}}(t) + \mathbf{K}_{1red}\, \dot{\mathbf{z}}(t) + \mathbf{K}_{0red}\, \mathbf{z}(t) = \mathbf{F}_{red}(t)$$

where

$$\mathbf{K}_{2red} = \mathbf{T}\,\mathbf{K}_2\,\mathbf{T}^T, \quad \mathbf{K}_{1red} = \mathbf{T}\,\mathbf{K}_1\,\mathbf{T}^T, \quad \mathbf{K}_{0red} = \mathbf{T}\,\mathbf{K}_0\,\mathbf{T}^T, \quad \mathbf{F}_{red} = \mathbf{T}\,\mathbf{F}.$$

Now we can solve the reduced system with the Runge - Kutta algorithm, and obtain the time evolution of $\mathbf{z}$ as an interpolating function of some discrete values at a number of instants. The values of the displacement vector $\mathbf{q}(t)$ can be easily obtained from eq. (3.8).

At this stage the results can be post - processed to find the required answers. For example, if we are interested in vibrations damping, a significative parameter to compute is the time needed to reduce the vibration amplitude of a certain quantity, starting with different initial conditions. Another aspect of particular interest in many applications is the energetic behavior of the whole system, partitioned in all the subsystems that compose it. In this way we can observe the energy interchange between the coupled systems. An easy way to compute the energy of all the subsystems is the following. Let $\mathbf{q}^s$ be a subvector extracted from $\mathbf{q}$ and referred to a particular subsystem, and let $\mathbf{K}_2^s$ and $\mathbf{K}_0^s$ be the corresponding submatrices. We can now define:

$$\begin{aligned}
\mathcal{E}_p^s &= \frac{1}{2}\mathbf{q}^{sT}\,\mathbf{K}_0^s\,\mathbf{q}^s, \\
\mathcal{E}_c^s &= \frac{1}{2}\dot{\mathbf{q}}^{sT}\,\mathbf{K}_2^s\,\dot{\mathbf{q}}^s, \\
\mathcal{E}_t^s &= \mathcal{E}_p^s + \mathcal{E}_c^s,
\end{aligned}$$

where $\mathcal{E}_p^s$, $\mathcal{E}_c^s$ and $\mathcal{E}_t^s$ are the generalized potential, kinetic and total energy.

Chapter 4

Element and shape functions

Since we are trying to build up a code which should allow the solution for a wide range of problems, an important feature will be the automatic mesh generation. Now, the simplest way to achieve this goal makes use of some triangulation algorithms (like Delauney's one), which easily allows to generate a triangular mesh for a wide class of 2-dimensional domains. This is the main reason which leads us to consider triangular elements as our first choice. So we investigated the huge literature about triangular finite elements suitable for our requirements.

As we said when formulating the mathematical model, we deal with 2-dimensional systems governed by second and fourth order partial differential equations. Therefore, the continuity requirement for the shape functions are different in each case. We need in fact C^0 shape functions for second order problems and C^1 shape functions for fourth order problems. Now, while the first case presents no difficulties, since linear shape functions with three corner nodes, each one with one degree of freedom, satisfies it, the second one presents some obstacles. In fact, in order to satisfy C^1 continuity, we observe that the complexity of the element (i.e. the degree of polynomial expansion and the total number of degrees of freedom) in-

creases enormously. This problem suggested that we consider the possibility of using *non-conforming* elements. That is, elements which do not satisfy the C^1 continuity requirement but have been proven to converge. In fact, even if for non-conforming elements we don't have the theoretical convergence insurance, there is another way to prove it, the *patch test.* This is a means of assessing convergence for problems in which the shape functions violate continuity requirements. This test is applicable to all finite elements and, if properly extended and interpreted, it can provide a *necessary* and *sufficient* condition for convergence, and also an assessment of the convergence rate. Therefore we decided to abandon conformity.

Now we describe the element that we found to be convenient for our work. It is a modification, proposed by Specht, of one of the first non-conforming elements (derived by Bazeley, Cheung, Irons and Zienkiewicz). To avoid asymmetry and gain a general formulation, we introduce the area coordinates, which are indeed a natural choice for triangles, viz. Fig.(4.1)

We shall use polynomial expansion terms, which are given in area coordinates. For instance,

$$\alpha_1 L_1 + \alpha_2 L_2 + \alpha_3 L_3$$

gives the three terms of a complete linear polynomial and

$$\alpha_1 L_1 L_2 + \alpha_2 L_2 L_3 + \alpha_3 L_3 L_1 + \alpha_4 L_1^2 + \alpha_5 L_2^2 + \alpha_6 L_3^2$$

gives all six terms of a quadratic, and so on.

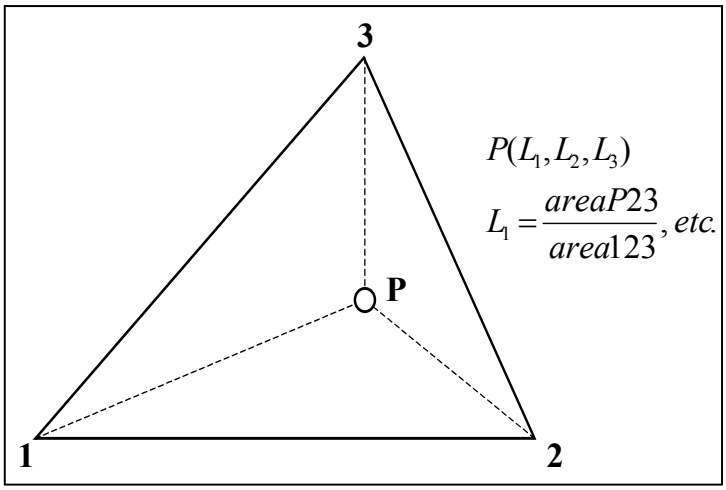

Figure 4.1: Local area coordinates

For a nine degrees of freedom element (three corner nodes, each one with three degrees of freedom: the displacement w and the slopes θ_x and θ_y) any of the above terms can be used in a suitable combination, remembering, however, that only nine independent functions are needed. We will ensure that all six quadratic terms are present. In addition, we select three fourth-order terms. The particular form of these is so designed that the patch test criterion is identically satisfied. We write

$$\begin{aligned} w &= [L_1, L_2, L_3, L_1L_2, L_2L_3, L_3L_1, \\ &\quad L_1^2L_2 + \frac{1}{2}L_1L_2L_3\{3(1-\mu_3)L_1 - (1+3\mu_3)L_2 + (1+3\mu_3)L_3\}, \\ &\quad L_2^2L_3 + \frac{1}{2}L_1L_2L_3\{3(1-\mu_1)L_2 - (1+3\mu_1)L_3 + (1+3\mu_1)L_1\}, \\ &\quad L_3^2L_1 + \frac{1}{2}L_1L_2L_3\{3(1-\mu_2)L_3 - (1+3\mu_2)L_1 + (1+3\mu_2)L_2\}]\boldsymbol{\alpha} \\ &= \mathbf{P}\boldsymbol{\alpha} \end{aligned}$$

where

$$\mu_1 = \frac{l_3^2 - l_2^2}{l_1^2} \qquad \mu_2 = \frac{l_1^2 - l_3^2}{l_2^2} \qquad \mu_3 = \frac{l_2^2 - l_1^2}{l_3^2}$$

and l_1, l_2, l_3 are lengths of the triangle sides.

On identification of nodal values and inversion, the shape functions can be written explicitly in terms of the component of the vector $\mathbf{P}$ as

$$\mathbf{N}_i^T = \begin{bmatrix} \mathbf{P}_i - \mathbf{P}_{i+3} + \mathbf{P}_{k+3} + 2(\mathbf{P}_{i+6} - \mathbf{P}_{k+6}) \\ -b_j(\mathbf{P}_{k+6} - \mathbf{P}_{k+3}) - b_k\mathbf{P}_{i+6} \\ -c_j(\mathbf{P}_{k+6} - \mathbf{P}_{k+3}) - c_k\mathbf{P}_{i+6} \end{bmatrix}$$

where i, j, k are the cyclic permutation of 1,2,3 and

$$\begin{aligned} b_1 &= y_2 - y_3 \\ c_1 &= x_3 - x_2 \end{aligned}$$

where $\{x_i, y_i\}$ are the coordinates of node i.

The element now derived passes all the patch tests and performs excellently.

If we deal with a certain number of coupled systems, some of second order and some of fourth order, we can easily build the global shape functions matrix simply by alternating, in a proper way, those shape functions (for fourth order problems) with linear shape functions (for second order problem). Obviously, on each corner node of the triangular element will be defined three independent variables for each fourth order function and one independent variable for each second order function. Calling n_4 and n_2 the numbers of fourth order and second

order functions respectively, we have that the total number of degrees of freedom per element is

$$DOF = 3(3n_4 + n_2).$$

Chapter 5

Software description

In the previous section we have shown how to formulate the problem following the Galerkin - weighted residual approach. Now we are going to illustrate the main structure and the principal features of the program. When writing finite element programs, one has to choose between developing an integral program with very few subroutines or producing a modular program in which different operations are carried out in individual subroutines. As seen in the previous section, the basic theoretical step of the finite element process can be separated into several distinct phases. Therefore it is logical to write finite element programs in which each theoretical operation is performed in a separate subroutine. Such an approach is adopted here. The structure of the program developed is illustrated in Fig. (5.1).

We observe that we divided the whole program structure into four main parts: the data input, the spatial finite element analysis, the time evolution solution and the data output. A separate description will be given for the control part of the code.

Now we briefly describe each one of these program blocks.

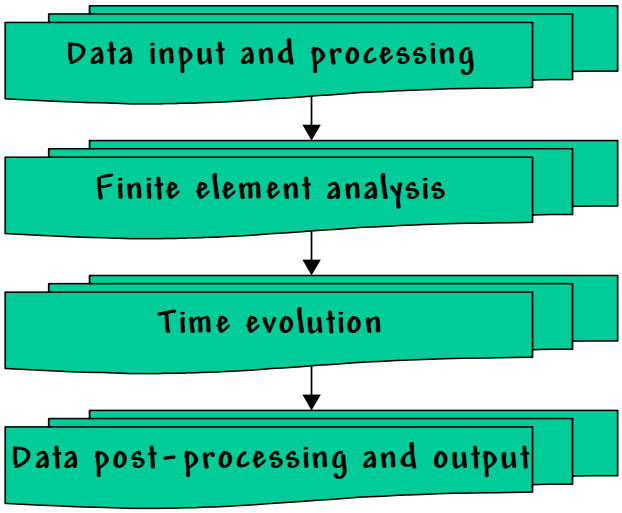

Figure 5.1: Program structure

Data input

This could seem an obvious aspect of the procedure, but, from a practical engineering point of view, it is one of the most important. Therefore some considerations should be made.

First of all, we must somehow input all the matrices involved in the table (2.1). However, once the system under analysis has been defined, it should be indicated to design a particular subroutine that automatically generates those matrices once some necessary parameters have been entered. Obviously this subroutine cannot be general but has to be programmed for each class of problem under consideration. It is however an easy procedure which gives good results in terms of time savings. Sometimes one may need to choose some parameters in a particular way;

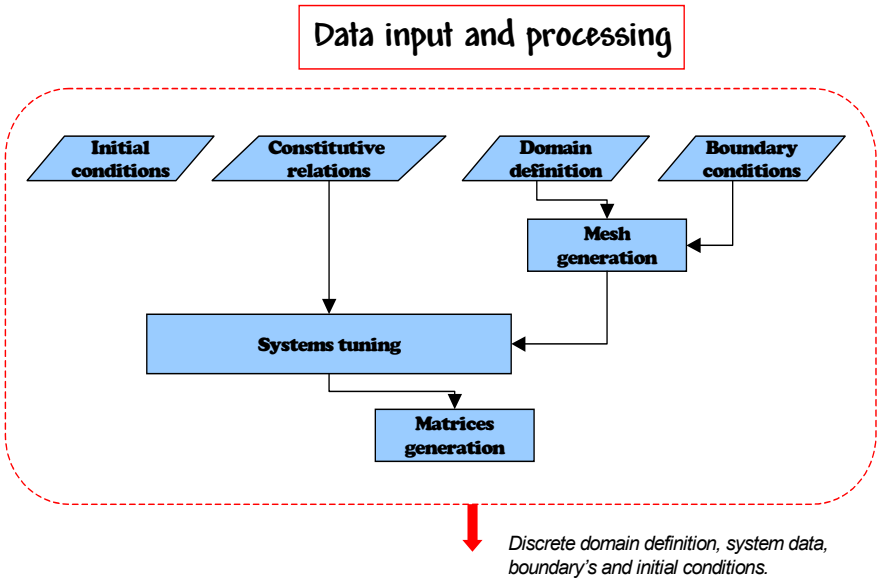

Figure 5.2: First block: data input

that is, maybe it is necessary to tune a certain subsystem on some eigenfrequency. This is an important aspect to consider in the realization of coupled structures; if it is necessary to obtain some internal resonance phenomena, all the subsystems have to be tuned on the same frequency. So we decided to add this important feature to the code. Using the FEM subroutines separately for each subsystem, the program computes the eigenfrequencies and corrects the parameters till reaching the desired results.

Another necessary information deals with the shape of the domain. The code needs the boundaries of the domain to be defined as polygonals; in practice one assigns a certain number of points which approximate the real aspect of the domain. For each part of the boundary one must give also the boundary conditions.

All those data are treated by a subroutine, the mesh generator, that subdivides the domain into a finite number of triangular elements, and gives three arrays of data: a nodes list, in which the node coordinates are written; an elements list, in which the number of the three corner nodes of each element are given; and a segments list, which gives the nodes on the boundaries.

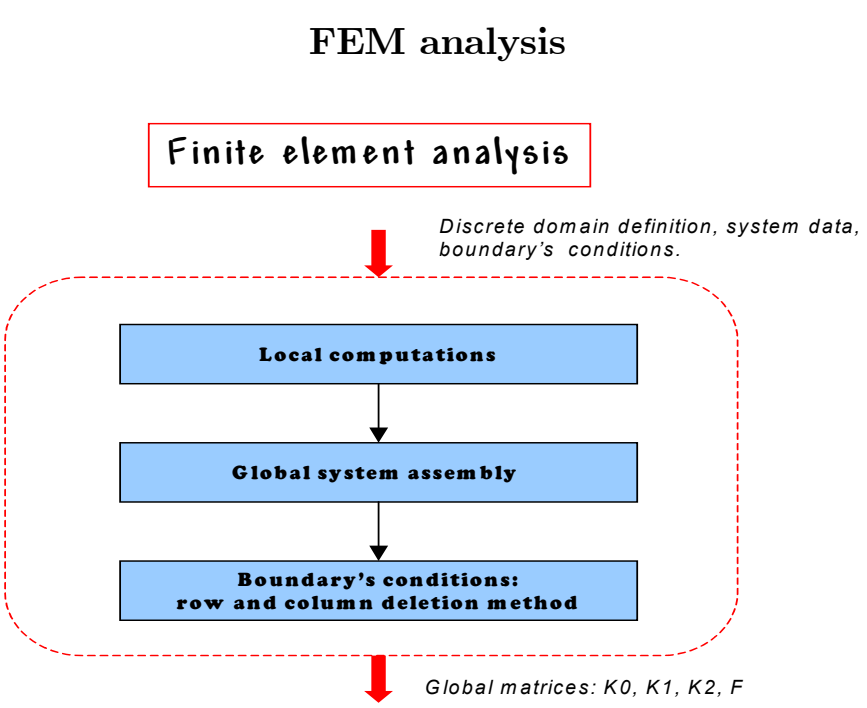

Figure 5.3: FEM analysis in the space domain

From a computational point of view, this is the phase which requires the greatest part of CPU time. In order to compute the local matrices of table (3.6), we have to perform an huge number of integrals of polynomials, and this could take a significant amount of time. To speed up the process we must use numerical

integration, based upon the Gauss quadrature technique.

The global matrices are built with the standard FEM approach, which is based upon the correspondence between local and global nodal numbering. Then we must impose the *essential* boundary conditions, and we can follow two different procedures: if those conditions are homogeneous (i.e. the values of some boundary nodal functions are compelled to be zero) we can delete the rows and columns of the final system which corresponds to zeroed variables; otherwise, we have to use the *penalty method*. Obviously, in the first case the final system has a smaller dimension than in the second case, so when it is possible we will prefer it.

If the problem is well posed, at the end of the process we will get non-singular mass and stiffness matrices.

Time evolution

The global system of differential equations derived in the previous stage is probably too big to be treated by a time-step algorithm like the Runge-Kutta one. Therefore we have to reduce the system. We can do this by projecting all the matrices involved on a certain base of eigenvectors; for instance, we can choose a certain number of eigenvectors for each uncoupled system to build up a base of the desired dimension. The initial conditions have to be projected as well, so that we have a new reduced system which can be successfully solved with the Runge-Kutta subroutine. At this stage it is important to have the possibility to control

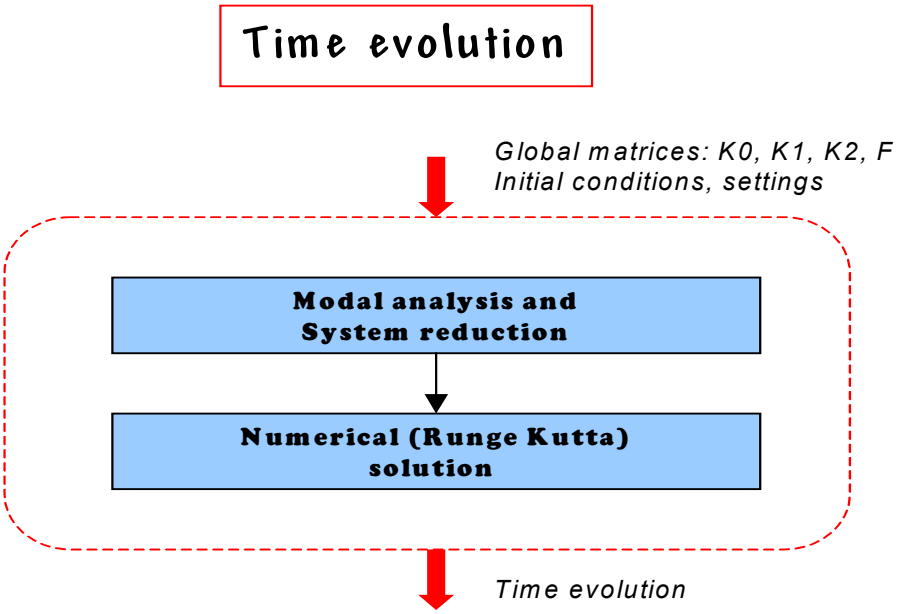

Figure 5.4: Third block: discrete time evolution

some aspect of this subroutine (the time interval between two sample points, the accuracy of the solution, the total simulation time). Then the subroutine will refine automatically the time step to reach the required accuracy. The solution of the process will be an array of sample values for each unknown of the system.

Data post processing and output

From an engineering point of view, the project of new devices requires some tools of analysis that give in a clear way an idea of the performances of the system designed. Therefore, this phase of numerical analysis must be oriented to show all the interesting results and comparisons with easy-to-read tables and plots. The array of values, representing the time evolution of the system, have

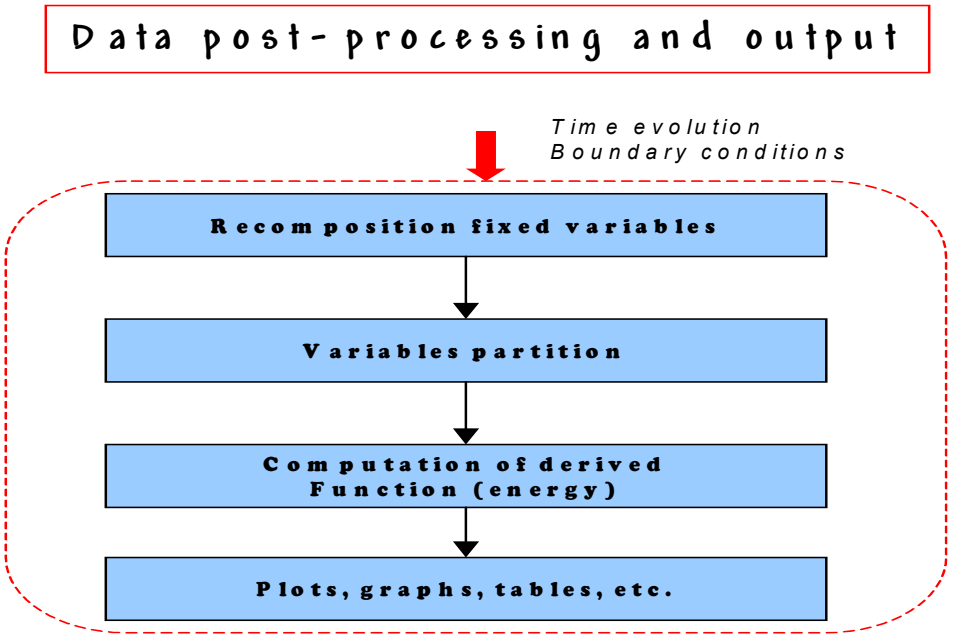

Figure 5.5: Results presentation

to be processed in different ways: first of all, we must join to the solution all the fixed nodes deleted when imposing the boundary conditions; then we have to partition the global system to obtain the behavior of each subsystem; finally we can compute all the derived functions of particular interest (first of all the energy of each subsystem). All those results must be presented with tables, plots, animations and so on.

The control settings

The code setup is a particular set of user's input, which allows one to decide the level of accuracy of each part of the code and, as a consequence, the CPU time needed to solve the problem.

First of all we can establish the mesh refinement, then the dimension of the reduced system's base, and finally the accuracy and the sample period of the Runge-Kutta subroutine. Furthermore, if we are interested in the tuning subroutine, we must choose the modes that have to be tuned and setup the procedure.

Part II

An application of the FEM code:

the piezo electromechanical plate.

Chapter 6

Description of the system

The control of structural vibrations is one of the foremost issues of mechanical engineering. Vibrations are undesirable for reliability, comfort and functionality of mechanical devices. Indeed,they are a cause of material failure by fatigue, they generate noise that is disturbing for humans, they impose a limit to the achievable precision in machinery, and their oscillations cause instability in aerodynamical systems.

Traditional solutions to the problem are to utilize viscoelastic material to add damping to the structure or to design the mass and stiffness distribution to control its dynamical behavior. An alternative device for a narrow band vibration damping is the Dynamical Vibration Absorber (DVA). It consists of a spring-mass-damper to be coupled to the system to obtain a resonant energy exchange and dissipation.

In the last decade many research efforts were devoted to study the applications to this field of the significant electromechanical coupling offered by the new generation of piezoelectric ceramics such as lead zirconate titanium (PZT). In this context active and passive techniques have been developed. Active controls

use PZT materials as sensor and actuators to apply a feedback control to the structure.

These achieve good performance,but they present the disadvantages of high power requirement, stability problems, and the need of a complex central unity for the implementation of the control law. Passive solutions suggest to utilize the two-ways piezo-electric-mechanical coupling to transfer mechanical energy into electrical form and then dissipate it in resistances by the Joule effect. It has been shown that resonant systems are more effective to this aim. In analogy to DVAs, piezoelectric transducers shunted on an inductance and a resistance can be bonded to a structure to form a coupled, highly dissipative system. Adjusting the electrical parameters, the RLC circuit can be tuned to a given mechanical mode replacing it with a pair of strongly damped electromechanical modes. This solution with respect to the active ones presents the advantages of constructive simplicity, intrinsic stability, and total independence from the environment.

Indeed, in principle, passive devices not only do not require power to work, but they can even be used to produce small amounts of energy. In practice the high inductance required to lower the electrical frequencies to the range of mechanical ones are frequently synthesized with electronic circuits that require a small amount of power to work.

Resonant Shunted Piezoelectrics (RSP) are strongly preferable to the mechanical analog DVAs because of their low cost, low weight, low space requirements

and flexibility. In fact, the electrical parameters can be easily adjusted to match the characteristics of the structural mode to be damped.

Exploiting this aspect, semi-active systems in which the values of the electrical parameters are chosen by a real time control unit also have been proposed. Industrial applications of the RRSP are now available. The greatest disadvantage of these devices is that they are effective only in a narrow band of frequencies because of their one-degree-of-freedom resonant nature.

An innovative idea presented by dell'Isola and Vidoli consists in establishing a distributed piezoelectric media to form a "smart structure" capable to be adjusted for an optimal broadband damping. They proposed to think about controlling a continuous mechanical structure with its electrical analog to enhance a complete communication coupling between the two systems.

In ([15],[16],[17],[18]) the problem of controlling the vibrations in beams and plates by distributing on them a set of PZT actuators (suitably electrically interconnetted) has been addressed. The new concept of 2-dimensional electric continuum was introduced, the evolution of which parallels that valid for the plate.

The main idea behind this concept is the following: to synthesize an electric circuit (coupled with the mechanical structure through the piezo-actuators), which has a behaviour analogous to the mechanical system. Exploiting this concept, it has been proposed to control plate-like structures by means of an internal

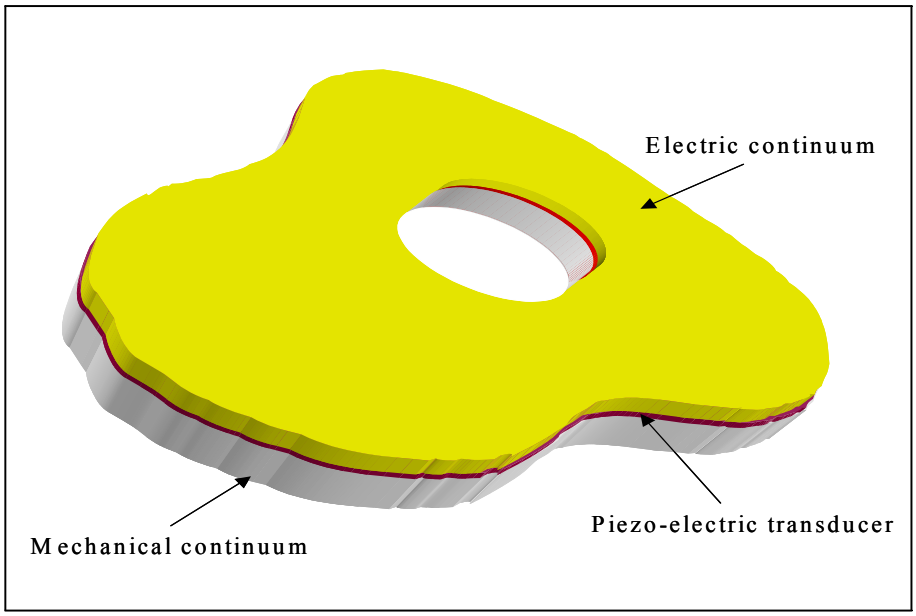

Figure 6.1: Electromechanical 2-dimensional continuum

resonance phenomenon between mechanical modes and electric modes.

Fig. (**??**) represents a scheme of such an electric network. Actually, we use discrete components, which connect a finite number of piezoelectric actuators which become TTN belonging to a modular electric network. Nevertheless, according to the hypothesis of large wavelengths compared with the components and modules dimensions (this is a plausible hypothesis, considering the mechanical wavelengths of lower modes), it is possible to study the electric network by means of a continuum (homogeneized) model. Therefore we must consider a 2-dimensional generalization of the transmission line. Moreover, using the well-known synthesis tecniques of discrete electric networks, it is possible to simulate every differential equation (for a detailed discussion of this point see [21]).

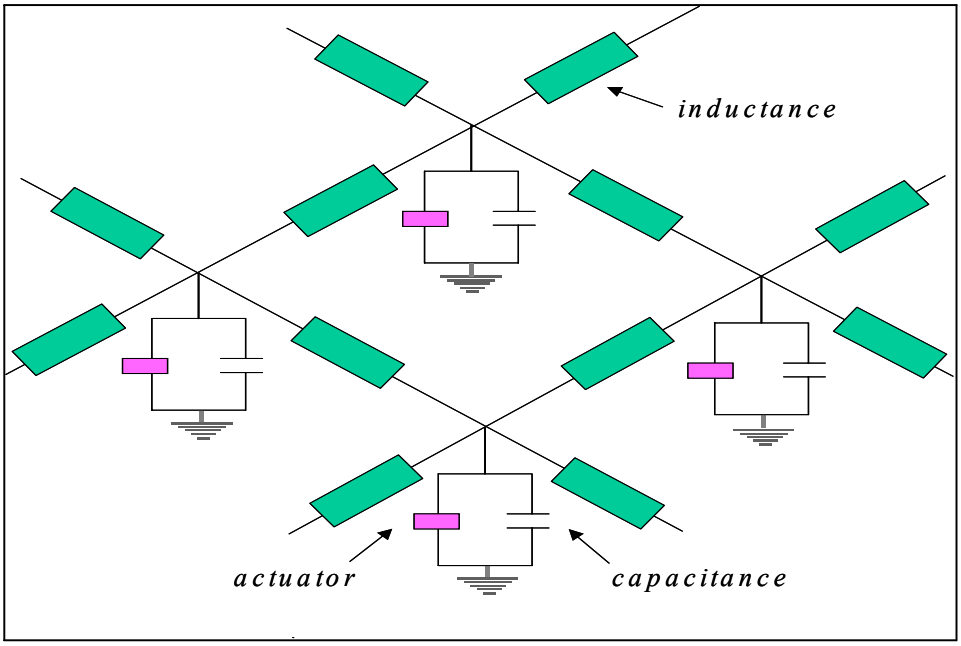

Figure 6.2: 2-dimensional electric network

In practice, in order to obtain the maximum coupling on every vibration mode, we want the dispersive relations that relate group velocity and frequency to be the same in the electric and mechanical sub-systems (see [17]). To satisfy this requirement the electric networks must *exhibit* the same governing differential equations that rule the mechanical sub-system. In other words we must choose the electric components in order to slow down the group velocity of the electric wave so as to make it equal to the mechanical one. In this way every electromechanical signal which propagates through the structure has a substantial electric content, which can easily be managed by the control system.

We remark explicitly that we aim to dissipate in a purely electrical way the

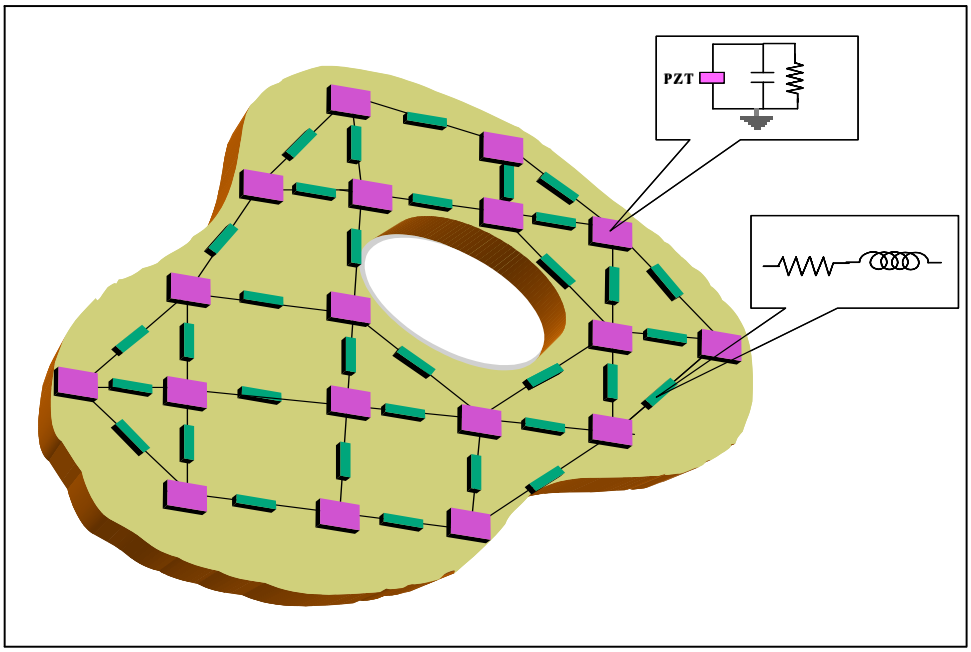

Figure 6.3: Lumped approximation of an electromechanical continuum

mechanical vibrational energy: this can be done by means of electrical resistors that, in the chosen topology (Fig. (**??**)), can be in series to the inductances or in parallel to the PZT.

Obviously, the electric network must be adapted depending on the mechanical phenomena we wish to control: for example, the longitudinal and torsional vibrations of a bar, governed by second order differential equations, or the flexural vibrations, governed by fourth order differential equations, are to be damped with dedicated electric networks.

Till now, only some elementary structures have been investigated with semi-analytical methods, as plates or beams with simple shapes, developing a suitable

electric network interconnecting the actuators. In particular, for flexural waves, the synthesis of an analogous circuit governed by fourth order equations is not straightforward. ([19], [21] and [20]).

Nevertheless some very interesting results can be obtained controlling the structure with a second order electric network. In this way we can maximize the coupling of only one vibration mode, but the circuit we have to realize is very simple. Anyway, this control tecnique has proven to have better performances and lower costs than other PZT based control tecniques ([22],[23]).

As found in [18], the PEM plates obtained interconnecting the electric terminals of the PZT transducers with a second order electric network are governed by the following system of PDEs

$$\begin{cases} \triangle\triangle w + \alpha\ddot{w} - \gamma\dot{\varphi} = 0 \\ \triangle\varphi - \beta\ddot{\varphi} + \gamma\triangle\dot{w} = 0 \end{cases} \tag{6.1}$$

to be completed by suitable boundary and initial conditions. Here w is the mechanical vertical displacement, φ is a time integral of the electric tension and γ is the coupling factor. Moreover, $\triangle$ denotes the 2-dimensional space laplacian operator while $\dot{x}$ denotes the the time derivative of x.

Chapter 7

Mathematical model

In this section we discuss the mathematical model for the systems under investigation. A weak formulation of the equations (6.1) is derived, and the parameters needed by the numerical code are computed by means of a standard identification procedure. Indeed we manage to determine the constitutive relations for introduced PEM plates (i.e. at our macro-level) in terms of the mechanical and electrical parameters characterizing the constituting mechanical and electrical elements at a micro-level: the identification in expended power simply generalizes the standard theory of Kirchhoff-Love plate (as done in [18]).

We remark that, in order to describe the main features of the behavior of the investigated electromechanical system, a continuum model was introduced (cfr. [18]). Since the physical realization of the electric part of PEM plates is obtained by lumped elements, this model allows for an accurate description of vibrations only when the involved wavelengths are not too small compared with the dimensions of the single element and the circuit module. By formulating the balance of power for the considered PEM plate, we obtain a weak formulation of the problem (6.1), which allows for a numerical analysis by means of the finite

element method.

We deal with a plate body $\mathcal{B}$ occupying the region $\mathcal{C} = \mathcal{S} \times \mathcal{I}$, where $\mathcal{S}$ is a plane surface, and $\mathcal{I}$ the real interval $[-h, h]$. As usual the thickness $2h$ is supposed to be small compared with the diameter of $\mathcal{S}$. The electric network is assumed to be two-dimensional. The following balance of power

$$\langle \mathbf{b}, \dot{\mathbf{u}} \rangle_{\mathcal{C}} + \langle \mathbf{f}, \dot{\mathbf{u}} \rangle_{\partial\mathcal{C}} = \langle \boldsymbol{\sigma}, \dot{\boldsymbol{\varepsilon}} \rangle_{\mathcal{C}} \tag{7.1}$$

must hold for every compatible test field $(\dot{\mathbf{u}}, \dot{\boldsymbol{\varepsilon}})$. Here $\mathbf{b}$ and $\mathbf{f}$ are the body and surface external forces, $\boldsymbol{\sigma}$, the stress tensor, $\mathbf{u}$ and $\boldsymbol{\varepsilon}$, the displacements and deformations fields respectively. We have also

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_z \\ i \end{bmatrix}, \qquad \mathbf{u} = \begin{bmatrix} u \\ v \\ w \\ \psi \end{bmatrix}, \qquad \mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ f_z \\ \iota \end{bmatrix},$$

where (b_1, b_2, b_z) and (f_1, f_2, f_z) are the components of the mechanical external forces, i and ι are body and surface current densities from ground, (u, v, w) are the components of the mechanical displacements vector, and ψ is the time integral of the electric potential between a point on the surface and the ground.

The stress and deformation field, according to the Cauchy model, are the following:

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_z \\ \tau_{12} \\ \tau_{z1} \\ \tau_{2z} \\ I_1 \\ I_2 \end{bmatrix}, \qquad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_z \\ \gamma_{12} \\ \gamma_{z1} \\ \gamma_{2z} \\ \Delta_1 \\ \Delta_2 \end{bmatrix} = \begin{bmatrix} u_{,1} \\ v_{,2} \\ w_{,z} \\ u_{,2} + v_{,1} \\ v_{,z} + w_{,2} \\ w_{,1} + u_{,z} \\ \psi_{,1} \\ \psi_{,2} \end{bmatrix},$$

where (I_1, I_2) are the components of the surface current density.

According to the geometry of the body, the position vector is decomposed as

$$\mathbf{x} = \mathbf{r} + z\,\mathbf{e}, \tag{7.2}$$

where $\mathbf{r}$ is the position vector in $\mathcal{S}$, $z \in \mathcal{I}$ and $\mathbf{e}$ is the unit vector perpendicular to $\mathcal{S}$. In order to deduce the behavior of a bending plate from the 3-dimensional Cauchy model of $\mathcal{B}$, we use the Kirchhoff-Love compatible identification procedure based on the following kinematical reduction map for mechanical displacements:

$$\mathbf{u}_m(\mathbf{r},\zeta) = \begin{bmatrix} -\dfrac{\partial w(\mathbf{r})}{\partial r_1} z \\ -\dfrac{\partial w(\mathbf{r})}{\partial r_2} z \\ w(\mathbf{r}) \end{bmatrix}.$$

Hence the function $w(\mathbf{r})$ models the transverse displacements of the plate's points. As already stated, the electric system is assumed to be 2-dimensional, and the

electric potential drop $\dot{\psi}$ depends on $\mathbf{r}$ only. Therefore

$$\mathbf{u}(\mathbf{r}) = \begin{bmatrix} \mathbf{u}_m(\mathbf{r}) \\ \phi(\mathbf{r}) \end{bmatrix} = \begin{bmatrix} -\dfrac{\partial w(\mathbf{r})}{\partial r_1} z \\ -\dfrac{\partial w(\mathbf{r})}{\partial r_2} z \\ w(\mathbf{r}) \\ \phi(\mathbf{r}) \end{bmatrix}. \tag{7.3}$$

Substituting in the eqn. (7.1) the reduction map (7.3) we obtain:

$$\left\langle \hat{\mathbf{b}},\, \hat{\mathbf{u}} \right\rangle_{\mathcal{S}} + \left\langle \mathbf{B}_1, \hat{\mathbf{u}}_{,1} \right\rangle_{\mathcal{S}} + \left\langle \mathbf{B}_2, \hat{\mathbf{u}}_{,2} \right\rangle + \left\langle \hat{\mathbf{f}}, \hat{\mathbf{u}} \right\rangle_{\partial\mathcal{S}} + \left\langle \mathbf{s}_1, \hat{\mathbf{u}}_{,1} \right\rangle_{\partial\mathcal{S}} + \left\langle \mathbf{s}_2, \hat{\mathbf{u}}_{,2} \right\rangle_{\partial\mathcal{S}} = \langle \hat{\boldsymbol{\sigma}}, \hat{\boldsymbol{\varepsilon}} \rangle_{\mathcal{S}} \,, \tag{7.4}$$

where

$$\hat{\mathbf{u}} = \begin{bmatrix} w \\ \psi \end{bmatrix} \qquad \hat{\mathbf{b}} = \begin{bmatrix} g \\ i \end{bmatrix}, \qquad \hat{\mathbf{f}} = \begin{bmatrix} q \\ \iota \end{bmatrix},$$

$$\mathbf{B}_1 = \begin{bmatrix} B_1 \\ 0 \end{bmatrix}, \qquad \mathbf{B}_2 = \begin{bmatrix} B_2 \\ 0 \end{bmatrix}, \qquad \mathbf{s}_1 = \begin{bmatrix} s_1 \\ 0 \end{bmatrix},$$

$$\mathbf{s}_2 = \begin{bmatrix} s_2 \\ 0 \end{bmatrix}, \qquad \hat{\boldsymbol{\sigma}} = \begin{bmatrix} m_1 \\ m_2 \\ m_{12} \\ I_1 \\ I_2 \end{bmatrix}, \qquad \hat{\boldsymbol{\varepsilon}} = \begin{bmatrix} \chi_1 \\ \chi_2 \\ \chi_{12} \\ \psi_{,1} \\ \psi_{,2} \end{bmatrix},$$

are the dynamical action and the kinematical fields in the reduced plate model, and where

$$
\begin{aligned}
g(\mathbf{r}) &= \int_{-h}^{h} b_z(\mathbf{r})\, dz, \quad B_1(\mathbf{r}) = \int_{-h}^{h} b_1(\mathbf{r})\, z\, dz, \quad B_2(\mathbf{r}) = \int_{-h}^{h} b_2(\mathbf{r})\, z\, dz, \quad (7.5)\\
q(\mathbf{r}) &= \int_{-h}^{h} f_z(\mathbf{r})\, dz, \quad s_1(\mathbf{r}) = \int_{-h}^{h} f_1(\mathbf{r})\, z\, dz, \quad s_2(\mathbf{r}) = \int_{-h}^{h} f_2(\mathbf{r})\, z\, dz,\\
m_1(\mathbf{r}) &= \int_{-h}^{h} \sigma_1(\mathbf{r})\, z\, dz, \quad m_2(\mathbf{r}) = \int_{-h}^{h} \sigma_2(\mathbf{r})\, z\, dz, \quad m_{12}(\mathbf{r}) = \int_{-h}^{h} \tau_{12}(\mathbf{r})\, z\, dz,\\
\chi_1 &= -w_{,11} \qquad \chi_2 = -w_{,22} \qquad \chi_{12} = -2w_{,12}
\end{aligned}
$$

We can express the infinitesimal deformation field in the following way:

$$
\hat{\boldsymbol{\varepsilon}} = \begin{bmatrix} \chi_1 \\ \chi_2 \\ \chi_{12} \\ \psi_{,1} \\ \psi_{,2} \end{bmatrix} = \mathbf{H}_0\hat{\mathbf{u}} + \mathbf{H}_1\hat{\mathbf{u}}_{,1} + \mathbf{H}_2\hat{\mathbf{u}}_{,2} + \mathbf{H}_3\hat{\mathbf{u}}_{,11} + \mathbf{H}_4\hat{\mathbf{u}}_{,22} + \mathbf{H}_5\hat{\mathbf{u}}_{,12}
$$

hence obtaining the first matrices for the numerical code.

To complete the model we need to consider the constitutive relations. For the mechanical part, assuming that the body $\mathcal{B}$ is linear orthotropic, we get:

$$
\begin{bmatrix} \boldsymbol{\sigma}_1 \\ \boldsymbol{\sigma}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{E}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{\varepsilon}_1 \\ \boldsymbol{\varepsilon}_2 \end{bmatrix},
$$

where $\boldsymbol{\sigma}_1^T = \{\sigma_1,\, \sigma_2,\, \sigma_z\}$ and $\boldsymbol{\varepsilon}_1^T = \{\varepsilon_1,\, \varepsilon_2,\, \varepsilon_z\}$. In the reduced model we have

$$
\mathbf{m} = \mathbf{E}_\chi \boldsymbol{\chi}
$$

where

$$
\mathbf{E}_\chi = \frac{2\,h^3}{3}\mathbf{E}_1.
$$

Moreover, considering the inertia forces, from (7.5) we get:

$$g = -2\,h\,\rho\,\ddot{w} + g_0\,,$$
$$B_1 = -\frac{2\,h^3}{3}\,\rho\,\ddot{w}_{,1} + B_1^0, \qquad B_2 = -\frac{2\,h^3}{3}\,\rho\,\ddot{w}_{,2} + B_2^0\,.$$

Assuming the network to be linear dissipative, the purely electric constitutive relations are:

$$\begin{aligned} i_e &= K_c\ddot{\psi} + G_N\dot{\psi} + i_0 \\ -\dot{\psi}_{,1} &= L_N\,\frac{\partial I_1}{\partial t} + R_N\,I_1 \\ -\dot{\psi}_{,2} &= L_N\,\frac{\partial I_2}{\partial t} + R_N\,I_2 \end{aligned}$$

where K_c and G_N are respectively the capacitance and the conductance from the surface to the ground, and L_N and R_N are respectively the net-inductance and net-resistance. We can invert the last two relations, getting:

$$\mathbf{I} = \mathbf{C}_0 e^{-\frac{R_N}{L_N}t} - \frac{1}{L_N}\int_{t_0}^{t} e^{-\frac{R_N}{L_N}(t-\tau)}(grad\,\dot{\psi})\,d\tau$$

Now let's assume the current on the net to be zero for $t = t_0$, so that $\mathbf{C}_0 = \emptyset$. We can define the following integral operator

$$Y_L(.) = -\frac{1}{L_N}\int_{t_0}^{t} e^{-\frac{R_N}{L_N}(t-\tau)}(.)\,d\tau\,,$$

and note that, assuming that time and space operators commute,

$$\mathbf{I} = Y_L(grad\,\dot{\psi}) = grad\,Y_L(\dot{\psi})\,.$$

In addition we specify the constitutive relation of the single PZT actuator:

$$\begin{bmatrix} m_1 \\ m_2 \\ m_{12} \\ \frac{Q}{d^2} \end{bmatrix} = \begin{bmatrix} g_{mm1} & 0 & 0 & -g_{me1} \\ 0 & g_{mm2} & 0 & -g_{me2} \\ 0 & 0 & g_{mm12} & -g_{me12} \\ g_{me1} & g_{me2} & g_{me12} & g_{ee} \end{bmatrix} \begin{bmatrix} \chi_1 \\ \chi_2 \\ \chi_{12} \\ \dot{\psi} \end{bmatrix}$$

which can be rearranged as follows:

$$\begin{bmatrix} \mathbf{m} \\ \frac{Q}{d^2} \end{bmatrix} = \begin{bmatrix} \mathbf{E}_\chi^p & -\mathbf{E}_c^T \\ \mathbf{E}_c & g_{ee} \end{bmatrix} \begin{bmatrix} \boldsymbol{\chi} \\ \dot{\psi} \end{bmatrix}$$

where $\mathbf{m}$ and $\boldsymbol{\chi}$ are the bending moments and curvatures, while $\frac{Q}{d^2}$ and $\dot{\psi}$ are the charge per unit area and voltage between the actuators plates, respectively.

Therefore the coupled constitutive relations for $\mathbf{m}$ and i read as follows:

$$\begin{aligned} \mathbf{m} &= \left(\mathbf{E}_\chi + \mathbf{E}_\chi^p\right) \boldsymbol{\chi} - \mathbf{E}_c^T \dot{\psi}\,, \\ i &= i_e + \frac{\dot{Q}}{d^2} = \left(K_c + g_{ee}\right) \ddot{\psi} + G_N \dot{\psi} + \mathbf{E}_c \dot{\boldsymbol{\chi}} + i_0 = \\ & \quad C_N \ddot{\psi} + G_N \dot{\psi} + \mathbf{E}_c \dot{\boldsymbol{\chi}} + i_0 \end{aligned}$$

At this stage we have to face the problem of the time integral operator Y_L. In order to remove it, we have to change the electric kinematical descriptor in the following way. We define

$$Y_L\left(\dot{\psi}\right) := \alpha\,,$$

hence

$$\begin{aligned} L_N\dot{\alpha} + R_N\alpha &= -\dot{\psi}\,, \\ L_N\ddot{\alpha} + R_N\dot{\alpha} &= -\ddot{\psi}\,. \end{aligned} \tag{7.6}$$

So the overall constitutive relations read as follows:

$$\begin{aligned} \mathbf{m} &= \mathbf{E}_{\chi g}\boldsymbol{\chi} + \mathbf{E}_c^T\left(L_N\dot{\alpha} + R_N\alpha\right)\,, \\ i &= -C_N\,\left(L_N\,\ddot{\alpha} + R_N\,\dot{\alpha}\right) - G_N\,\left(L_N\,\dot{\alpha} + R_N\,\alpha\right) + \mathbf{E}_c\,\dot{\boldsymbol{\chi}}\,, \\ \mathbf{I} &= grad\,\boldsymbol{\alpha}\,, \qquad g = -2\,h\,\rho\,\ddot{w} + g_0\,, \\ B_1 &= -\frac{2\,h^3}{3}\,\rho\,\ddot{w}_{,1} + B_1^0, \qquad B_2 = -\frac{2\,h^3}{3}\,\rho\,\ddot{w}_{,2} + B_2^0\,, \\ \mathbf{s} &= \mathbf{s}_0\,, \qquad \hat{\mathbf{f}} = \hat{\mathbf{f}}_0\,. \end{aligned}$$

Now we can put everything in the general form (2.1), posing:

$$\begin{aligned} \mathbf{G} &= \begin{bmatrix} -2h\rho & 0 \\ 0 & -C_N L_N \end{bmatrix}, \qquad \mathbf{S} = \begin{bmatrix} 0 & 0 \\ 0 & -C_N R_N - G_N L_N \end{bmatrix}, \\ \mathbf{G}_B^1 &= \begin{bmatrix} -\frac{2\,h^3}{3}\,\rho & 0 \\ 0 & 0 \end{bmatrix}, \qquad \mathbf{G}_B^2 = \begin{bmatrix} -\frac{2\,h^3}{3}\,\rho & 0 \\ 0 & 0 \end{bmatrix}, \end{aligned}$$

$$\mathbf{T} = \begin{bmatrix} 0 & 0 \\ 0 & -G_N R_N \end{bmatrix}, \qquad \mathbf{V} = \begin{bmatrix} g_{me1} & g_{me2} & g_{me12} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad \mathbf{E} = \begin{bmatrix} \mathbf{E}_{\chi g} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} 0 & L_N\, g_{me1} \\ 0 & L_N\, g_{me2} \\ 0 & L_N\, g_{me12} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \qquad \mathbf{R} = \begin{bmatrix} 0 & R_N\, g_{me1} \\ 0 & R_N\, g_{me2} \\ 0 & R_N\, g_{me12} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

and

$$\mathbf{H}_0 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \qquad \mathbf{H}_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix},$$

$$\mathbf{H}_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \mathbf{H}_3 = \begin{bmatrix} -1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

$$\mathbf{H}_4 = \begin{bmatrix} 0 & 0 \\ -1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \qquad \mathbf{H}_5 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -2 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

Note that the displacement vector is now:

$$\mathbf{u}(\mathbf{r}) = \begin{bmatrix} w(\mathbf{r}) \\ \alpha(\mathbf{r}) \end{bmatrix}$$

We have arranged the problem in order to make it fit the general case that the code is able to manage. The last step, which will provide us an easier understanding of the results, is to give a dimensionless formulation of the whole problem. So let us introduce the dimensionless time and space coordinates:

$$T = \omega t\,, \qquad \mathbf{x} = \frac{\mathbf{r}}{l}\,, \tag{7.7}$$

where l and ω are, respectively, characteristic length and angular frequency, and the dimensionless time and space derivatives:

$$\frac{\partial}{\partial \mathbf{r}} \to \frac{1}{l}\frac{\partial}{\partial \mathbf{x}}\,, \qquad \frac{\partial}{\partial \mathbf{t}} \to \omega \frac{\partial}{\partial \mathbf{T}}\,. \tag{7.8}$$

Hence we get the dimensionless displacements

$$k(\mathbf{r}) = \frac{w(\mathbf{r})}{l}\,, \qquad \phi'(\mathbf{r}) = \frac{\dot{\psi}(\mathbf{r})}{V}\,. \tag{7.9}$$

where V is the characteristic voltage and $'$ holds for dimensionless time derivative. We must derive also a dimensionless form for the variable α we have introduced. So let us consider the relation (7.6), which we rewrite here

$$L_N\,\dot{\alpha} + R_N\,\alpha = -\dot{\psi}\,,$$

that, with equations (7.9), (7.7) and (7.8) becomes:

$$L_N\,\omega\,\alpha' + R_N\,\alpha = -V\,\dot{\boldsymbol{\phi}}$$

Hence we can define the dimensionless variable:

$$\delta = \frac{L_N\omega}{V}\alpha\,,$$

and the relation (7.6) reads as follows:

$$\delta' + \frac{R_N}{L_N\omega}\delta = -\boldsymbol{\phi}'$$

We can give the transformation rules for all the matrices involved in the problem. First of all, note that

$$\mathbf{u} = \begin{bmatrix} w \\ \\ \psi \end{bmatrix} = \begin{bmatrix} l & 0 \\ \\ 0 & \frac{V}{L_N\omega} \end{bmatrix} \begin{bmatrix} k \\ \\ \phi \end{bmatrix} = \mathbf{A}\,\mathbf{u}_d\,, \tag{7.10}$$

We observe that:

$$\dot{\mathbf{u}} = \omega\,\mathbf{A}\,\mathbf{u}_d'\,, \tag{7.11}$$

$$\ddot{\mathbf{u}} = \omega^2\,\mathbf{A}\,\mathbf{u}_d''\,. \tag{7.12}$$

Moreover, we have to consider that the change in the kinematical descriptor of the electric part of the system (i. e. α instead of ψ) has another side effect. In fact, even if we give all the constitutive relations in terms of α, the dimensional form of the trial fields in equation (7.1) must be the original one, that is with a virtual ψ and not a virtual α. This fact can be taken into account by considering a different dimension matrix for the test fields:

$$\dot{\mathbf{u}}^t = \begin{bmatrix} \dot{w}^t \\ \dot{\psi}^t \end{bmatrix} = \omega \begin{bmatrix} l & 0 \\ 0 & \frac{V}{\omega} \end{bmatrix} \begin{bmatrix} k^{t\prime} \\ \phi^{t\prime} \end{bmatrix} = \omega\, \hat{\mathbf{A}}\, \mathbf{u}_d^{t\prime}\,. \tag{7.13}$$

With some computation we can find the kinematical compatibility relation between $\boldsymbol{\varepsilon}$ and $\mathbf{u}$

$$\begin{aligned} \boldsymbol{\varepsilon} &= \mathbf{H}_0\mathbf{u} + \mathbf{H}_1\mathbf{u}_{,1} + \mathbf{H}_2\mathbf{u}_{,2} + \mathbf{H}_3\mathbf{u}_{,11} + \mathbf{H}_4\mathbf{u}_{,22} + \mathbf{H}_5\mathbf{u}_{,12} \qquad (7.14)\\ &= \mathbf{H}_0\,\mathbf{A}\,\mathbf{u}_d + \frac{1}{l}\mathbf{H}_1\,\mathbf{A}\,\mathbf{u}_{d,1} + \frac{1}{l}\mathbf{H}_2\,\mathbf{A}\,\mathbf{u}_{d,2} + \frac{1}{l^2}\mathbf{H}_3\,\mathbf{A}\,\mathbf{u}_{d,11} + \frac{1}{l^2}\mathbf{H}_4\,\mathbf{A}\,\mathbf{u}_{d,22} + \frac{1}{l^2}\mathbf{H}_5\,\mathbf{A}\,\mathbf{u}_{d,12}\\ &= \bar{\mathbf{H}}_0\,\mathbf{u}_d + \bar{\mathbf{H}}_1\,\mathbf{u}_{d,1} + \bar{\mathbf{H}}_2\,\mathbf{u}_{d,2} + \bar{\mathbf{H}}_3\,\mathbf{u}_{d,11} + \bar{\mathbf{H}}_4\,\mathbf{u}_{d,22} + \bar{\mathbf{H}}_5\,\mathbf{u}_{d,12} \end{aligned}$$

where

$$\begin{aligned} \bar{\mathbf{H}}_0 &= \mathbf{H}_0\mathbf{A}\,, \qquad \bar{\mathbf{H}}_1 = \frac{1}{l}\mathbf{H}_1\mathbf{A}\,, \qquad \bar{\mathbf{H}}_2 = \frac{1}{l}\mathbf{H}_2\mathbf{A}\,, \qquad (7.15)\\ \bar{\mathbf{H}}_3 &= \frac{1}{l^2}\mathbf{H}_3\mathbf{A}\,, \qquad \bar{\mathbf{H}}_4 = \frac{1}{l^2}\mathbf{H}_4\mathbf{A}\,, \qquad \bar{\mathbf{H}}_5 = \frac{1}{l^2}\mathbf{H}_5\mathbf{A}\,. \end{aligned}$$

With similar reasoning we get:

$$\begin{aligned} \boldsymbol{\varepsilon}^t &= \mathbf{H}_0\mathbf{u}^t + \mathbf{H}_1\mathbf{u}^t_{,1} + \mathbf{H}_2\mathbf{u}^t_{,2} + \mathbf{H}_3\mathbf{u}^t_{,11} + \mathbf{H}_4\mathbf{u}^t_{,22} + \mathbf{H}_5\mathbf{u}^t_{,12} \qquad (7.16)\\ &= \mathbf{H}_0\,\hat{\mathbf{A}}\,\mathbf{u}_d + \frac{1}{l}\mathbf{H}_1\,\hat{\mathbf{A}}\,\mathbf{u}_{d,1} + \frac{1}{l}\mathbf{H}_2\,\hat{\mathbf{A}}\,\mathbf{u}_{d,2} + \frac{1}{l^2}\mathbf{H}_3\,\hat{\mathbf{A}}\,\mathbf{u}_{d,11} + \frac{1}{l^2}\mathbf{H}_4\,\hat{\mathbf{A}}\,\mathbf{u}_{d,22} + \frac{1}{l^2}\mathbf{H}_5\,\hat{\mathbf{A}}\,\mathbf{u}_{d,12}\\ &= \bar{\mathbf{H}}^t_0\,\mathbf{u}_d + \bar{\mathbf{H}}^t_1\,\mathbf{u}_{d,1} + \bar{\mathbf{H}}^t_2\,\mathbf{u}_{d,2} + \bar{\mathbf{H}}^t_3\,\mathbf{u}_{d,11} + \bar{\mathbf{H}}^t_4\,\mathbf{u}_{d,22} + \bar{\mathbf{H}}^t_5\,\mathbf{u}_{d,12} \end{aligned}$$

We can note that all the space derivatives are dimensionless. Now we can substitute the constitutive relations and all the positions from (7.10) to (7.16) in the equation (2.1), which becomes:

$$l^2 \left\langle \mathbf{G}\,\omega^2\,\mathbf{A}\,\mathbf{u}_d''+\mathbf{S}\,\omega\,\mathbf{A}\,\mathbf{u}_d'+\mathbf{T}\,\mathbf{A}\,\mathbf{u}_d+\omega\,\mathbf{V}\,\boldsymbol{\varepsilon}'+\mathbf{b}_0, \omega\,\hat{\mathbf{A}}\,\mathbf{u}_d^{t\prime} \right\rangle_{\mathcal{S}_d} + l^2 \left\langle \mathbf{G}_B^1\,\frac{\omega^2}{l}\mathbf{A}\,\mathbf{u}_{d,1}''+\mathbf{B}_1^0, \frac{\omega}{l}\hat{\mathbf{A}}\,\mathbf{u}_{d,1}^{t\prime} \right\rangle_{\mathcal{S}_d} +$$

$$l^2 \left\langle \mathbf{G}_B^2\,\frac{\omega^2}{l}\mathbf{A}\,\mathbf{u}_{d,2}''+\mathbf{B}_2^0, \frac{\omega}{l}\hat{\mathbf{A}}\,\mathbf{u}_{d,2}^{t\prime} \right\rangle_{\mathcal{S}_d} + l \left\langle \hat{\mathbf{f}}_0, \omega\,\hat{\mathbf{A}}\,\mathbf{u}_d^{t\prime} \right\rangle_{\partial\mathcal{S}_d} + l \left\langle \mathbf{s}_1^0, \frac{\omega}{l}\hat{\mathbf{A}}\,\mathbf{u}_{d,1}^{t\prime} \right\rangle_{\partial\mathcal{S}_d} + l \left\langle \mathbf{s}_2^0, \frac{\omega}{l}\hat{\mathbf{A}}\,\mathbf{u}_{d,2}^{t\prime} \right\rangle_{\partial\mathcal{S}_d} =$$

$$l^2 \left\langle \mathbf{E}\,\boldsymbol{\varepsilon} + \mathbf{C}\,\omega\,\mathbf{A}\,\mathbf{u}_d'+\mathbf{R}\,\mathbf{A}\,\mathbf{u}_d+\boldsymbol{\sigma}_0, \boldsymbol{\varepsilon}^t \right\rangle_{\mathcal{S}_d}$$

With some manipulations we get:

$$\left\langle \omega^3\,\hat{\mathbf{A}}\,\mathbf{G}\,\mathbf{A}\,\mathbf{u}_d'',\, \mathbf{u}_d^{t\prime} \right\rangle_{\mathcal{S}_d} + \left\langle \frac{\omega^3}{l^2}\hat{\mathbf{A}}\,\mathbf{G}_B^1\,\mathbf{A}\,\mathbf{u}_{d,1}'', \mathbf{u}_{d,1}^{t\prime} \right\rangle_{\mathcal{S}_d} + \left\langle \frac{\omega^3}{l^2}\hat{\mathbf{A}}\,\mathbf{G}_B^2\,\mathbf{A}\,\mathbf{u}_{d,2}'', \mathbf{u}_{d,2}^{t\prime} \right\rangle_{\mathcal{S}_d} +$$

$$\left\langle \omega^2\,\hat{\mathbf{A}}\,\mathbf{V}\,\boldsymbol{\varepsilon}',\, \mathbf{u}_d^{t\prime} \right\rangle_{\mathcal{S}_d} + \left\langle \omega\,\hat{\mathbf{A}}\,\mathbf{T}\,\mathbf{A}\,\mathbf{u}_d,\, \mathbf{u}_d^{t\prime} \right\rangle_{\mathcal{S}_d} +$$

$$\left\langle \omega^2\,\hat{\mathbf{A}}\,\mathbf{S}\,\mathbf{A}\,\mathbf{u}_d',\, \mathbf{u}_d^{t\prime} \right\rangle_{\mathcal{S}_d} + \left\langle \frac{\omega}{l}\hat{\mathbf{A}}\,\mathbf{B}_1^0, \mathbf{u}_{d,1}^{t\prime} \right\rangle_{\mathcal{S}_d} + \left\langle \frac{\omega}{l}\hat{\mathbf{A}}\,\mathbf{B}_2^0, \mathbf{u}_{d,2}^{t\prime} \right\rangle_{\mathcal{S}_d} +$$

$$\left\langle \omega\hat{\mathbf{A}}\;\mathbf{b}_0,\, \mathbf{u}_d^{t\prime} \right\rangle_{\mathcal{S}_d} + \left\langle \frac{\omega}{l}\hat{\mathbf{A}}\,\hat{\mathbf{f}}_0, \mathbf{u}_d^{t\prime} \right\rangle_{\partial\mathcal{S}_d} + \left\langle \frac{\omega}{l^2}\hat{\mathbf{A}}\,\mathbf{s}_1^0, \mathbf{u}_{d,1}^{t\prime} \right\rangle_{\partial\mathcal{S}_d} + \left\langle \frac{\omega}{l^2}\hat{\mathbf{A}}\,\mathbf{s}_2^0, \mathbf{u}_{d,2}^{t\prime} \right\rangle_{\partial\mathcal{S}_d} =$$

$$\left\langle \omega\,\mathbf{E}\,\boldsymbol{\varepsilon}+\omega^2\,\mathbf{C}\,\mathbf{A}\,\mathbf{u}_d'+\omega\,\mathbf{R}\,\mathbf{A}\,\mathbf{u}_d+\omega\,\boldsymbol{\sigma}_0, \boldsymbol{\varepsilon}^{t\prime} \right\rangle_{\mathcal{S}_d}$$

Now we can obtain the coefficient of k'' to be of the order of the unity:

$$\begin{aligned} \omega^3\hat{\mathbf{A}}\,\mathbf{G}\,\mathbf{A} &= \omega^3 \begin{bmatrix} -2h\rho & 0 \\ 0 & -C_N\,L_N \end{bmatrix} \begin{bmatrix} l^2 & 0 \\ 0 & \frac{V^2}{L_N\omega^2} \end{bmatrix} = \\ &= \omega^3 2hl\bar{\rho} \begin{bmatrix} -\tilde{\rho} & 0 \\ 0 & -\frac{V^2\bar{C}_N\,\tilde{c}}{2hl^2\bar{\rho}\omega^2} \end{bmatrix} \end{aligned}$$

where we assume that $\rho(\mathbf{r}) = \bar{\rho}\,\tilde{\rho}(\mathbf{r})$ and $C_N(\mathbf{r}) = \bar{C}_N\,\tilde{c}(\mathbf{r})$.

Let us choose

$$V = \sqrt{\frac{2h\bar{\rho}l^2\omega^2}{\bar{C}_N}}$$

so that:

$$\omega^3 \hat{\mathbf{A}}^T \mathbf{G} \mathbf{A} = \omega^3 2hl^2\bar{\rho} \begin{bmatrix} -\tilde{\rho} & 0 \\ 0 & -\tilde{c} \end{bmatrix}$$

Now we can divide the equation by $\omega^3 2hl^2\bar{\rho}$ obtaining the dimensionless formulation for which we were aiming. In table (7.17) we report all the transformation rules needed to get the new matrices that we have to substitute in the equations (2.1).

Inertia	$\bar{\mathbf{G}}=\frac{1}{2hl^2\bar{\rho}}\hat{\mathbf{A}}\,\mathbf{G}\,\mathbf{A}\,,$ $\bar{\mathbf{G}}^1_B=\frac{1}{2hl^4\bar{\rho}}\hat{\mathbf{A}}\,\mathbf{G}^1_B\,\mathbf{A}\,,\qquad \bar{\mathbf{G}}^2_B=\frac{1}{2hl^4\bar{\rho}}\hat{\mathbf{A}}\,\mathbf{G}^2_B\,\mathbf{A}.$
Stiffness	$\tilde{\mathbf{E}}=\frac{1}{\omega 2hl^2\bar{\rho}}\mathbf{E}.$
Kinematical compatibility	$\bar{\mathbf{H}}_0=\mathbf{H}_0\mathbf{A}\,,\quad \bar{\mathbf{H}}_1=\frac{1}{l}\mathbf{H}_1\mathbf{A}\,,\quad \bar{\mathbf{H}}_2=\frac{1}{l}\mathbf{H}_2\mathbf{A}\,,$ $\bar{\mathbf{H}}_3=\frac{1}{l^2}\mathbf{H}_3\mathbf{A}\,,\quad \bar{\mathbf{H}}_4=\frac{1}{l^2}\mathbf{H}_4\mathbf{A}\,,\quad \bar{\mathbf{H}}_5=\frac{1}{l^2}\mathbf{H}_5\mathbf{A}\,.$ $\bar{\mathbf{H}}^t_0=\mathbf{H}_0\hat{\mathbf{A}}\,,\quad \bar{\mathbf{H}}^t_1=\frac{1}{l}\mathbf{H}_1\hat{\mathbf{A}}\,,\quad \bar{\mathbf{H}}^t_2=\frac{1}{l}\mathbf{H}_2\hat{\mathbf{A}}\,,$ $\bar{\mathbf{H}}^t_3=\frac{1}{l^2}\mathbf{H}_3\hat{\mathbf{A}}\,,\quad \bar{\mathbf{H}}^t_4=\frac{1}{l^2}\mathbf{H}_4\hat{\mathbf{A}}\,,\quad \bar{\mathbf{H}}^t_5=\frac{1}{l^2}\mathbf{H}_5\hat{\mathbf{A}}\,.$
Coupling and damping terms	$\bar{\mathbf{R}}=\frac{1}{\omega^2 2hl^2\bar{\rho}}\mathbf{R}\mathbf{A}\,,\quad \bar{\mathbf{C}}=\frac{1}{\omega 2hl^2\bar{\rho}}\mathbf{C}\,\mathbf{A}\,,$ $\bar{\mathbf{S}}=\frac{1}{\omega 2hl^2\bar{\rho}}\hat{\mathbf{A}}^T\mathbf{S}\mathbf{A}\,,\quad \bar{\mathbf{T}}=\frac{1}{\omega^2 2hl^2\bar{\rho}}\hat{\mathbf{A}}^T\mathbf{T}\mathbf{A}\,,$ $\bar{\mathbf{V}}=\frac{1}{\omega 2hl^2\bar{\rho}}\hat{\mathbf{A}}^T\mathbf{V}.$
External forces and initial distortion	$\bar{\mathbf{b}}_0=\frac{1}{\omega^2 2hl^2\bar{\rho}}\hat{\mathbf{A}}^T\mathbf{b}_0\,,\quad \bar{\mathbf{B}}^0_1=\frac{1}{\omega^2 2hl^3\bar{\rho}}\hat{\mathbf{A}}\,\mathbf{B}^0_1\,,$ $\bar{\mathbf{B}}^0_2=\frac{1}{\omega^2 2hl^3\bar{\rho}}\hat{\mathbf{A}}\,\mathbf{B}^0_2\,,\quad \bar{\mathbf{f}}=\frac{1}{l}\frac{1}{\omega^2 2hl^2\bar{\rho}}\hat{\mathbf{A}}^T\mathbf{f}\,,$ $\bar{\mathbf{s}}^0_1=\frac{1}{\omega^2 2hl^4\bar{\rho}}\hat{\mathbf{A}}\,\mathbf{s}^0_1\,,\quad \bar{\mathbf{s}}^0_2=\frac{1}{\omega^2 2hl^4\bar{\rho}}\hat{\mathbf{A}}\,\mathbf{s}^0_2\,,$ $\bar{\boldsymbol{\sigma}}_0=\frac{1}{\omega^2 2hl^2\bar{\rho}}\boldsymbol{\sigma}_0\,.$

(7.17)

Chapter 8

Numerical results

We now present the numerical results obtained in the analysis of PEM structures. We report here two examples of our program's application. Both of them refer to an aluminum plate, coupled by PZT actuators to an electric network, as shown in Fig. (8.1).

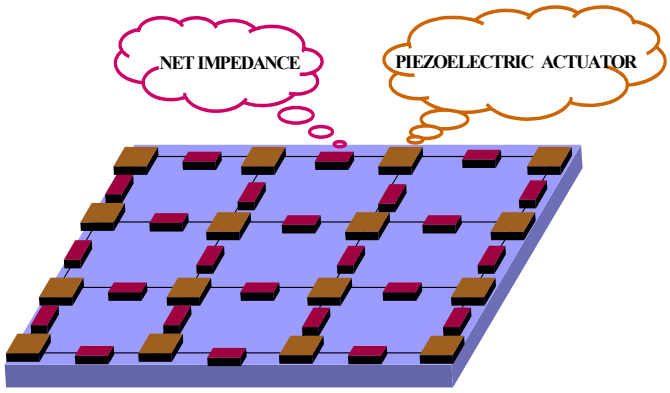

Figure 8.1: Schematic of the system

The values of mechanical and electric parameters are reported in the next table (Fig. (8.2))

The difference between the two example cases is in their geometrical shape. The first analyzed case (the square plate) is actually a test case, which shows

Mechanical parameters	Electric parameters	PZT parameters
Young module (Pascal) $E_y = 70 \cdot 10^9$	To be adjusted to optimize the coupling	Mechanical behavior (Newton) $g_{mn1} = 0.0001$ $g_{mn2} = 0.0001$ $g_{mn12} = 0.00001$
Poisson's ratio (dimensionless) $\nu = 0.33$		Electric behavior (Farad) $g_{ee} = 0.0001$
Mass density (Kg/m^3) $\rho = 2700$		Coupling terms $g_{me1} = 0.001$ $g_{me2} = 0.001$ $g_{me12} = 0$
Plate thickness (m) $t = 0.002$		

Figure 8.2: Device parameters

the accuracy of the numerical code. The second case deals with a more complex geometry that does not allow for an analytical approach.

Test case: the simply supported square PEM plate

We start by considering a simply supported square PEM plate in order to obtain a first test of the performance of the developed code.

The case under consideration has been deeply investigated in ([18]), where semi-analitical solutions have been determined by means of modal analysis. The main feature of this simplified problem consists in the fact that the uncoupled electric and mechanical systems are solved easily because of their simple geometry. Also when we deal with the coupled system, formed by a second order system (the electric network) and a fourth order one (the flexural mechanical wave prop-

agation), a simple series representation of the solution can be provided. We will show how the numerical process converges to the exact solution for both cases and how the coupled system is approximated as well.

Uncoupled systems

The classical solutions for the simply supported square plate, as well as for the second order (membrane-like) equations of the electric network, are compared to the numerical results obtained for a suitably refined mesh. The modal shapes and the eigenfrequencies approximate accurately the exact solution, as shown in Figs (8.3)-(8.6). Figure (8.6) shows that the error in the eigenfrequencies is below 1.2% for the first eigth modes. Also in the modal shapes (Figs (8.3) and (8.4)) we recognize the well known sinusoidal functions that constitute the semi-analitical solution of these systems.

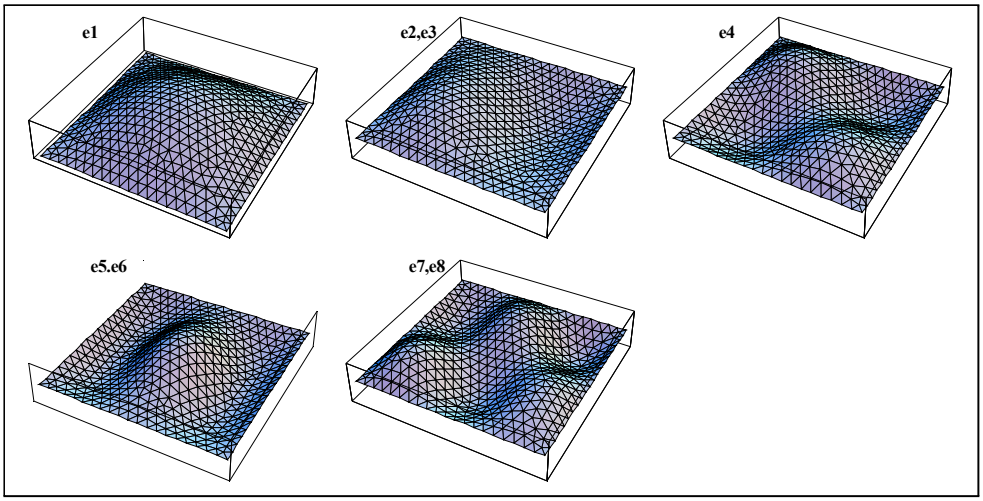

Figure 8.3: Electric modal shapes

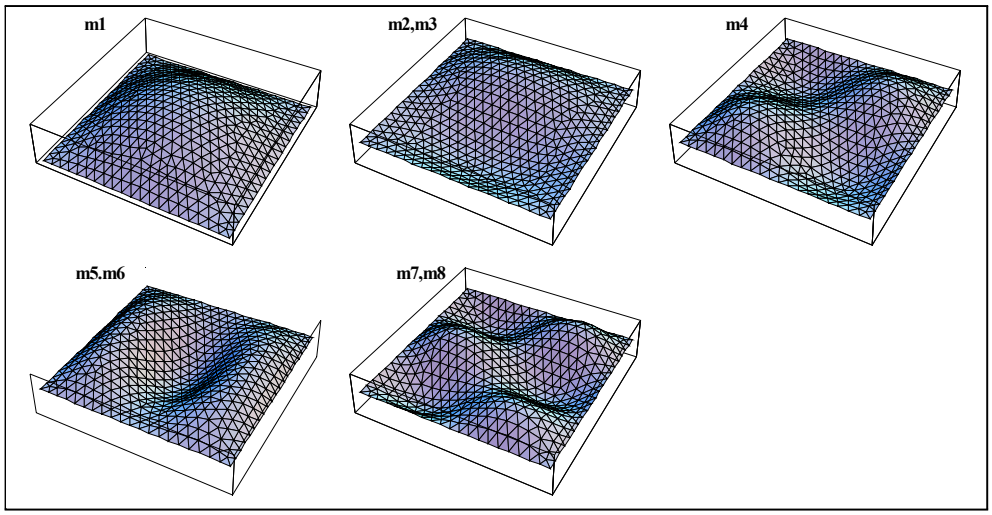

Figure 8.4: Mechanical modal shapes

The first eight modes, for each system, are considered. The electric network has been tuned (choosing a proper value of the net inductance) so that the first electric eigenfrequency matches that of the mechanical system. In this simple case the tuning procedure is trivial, since we dispose of the analytical relations between the net inductance and the electric eigenfrequencies.

Energy exchange in coupled system: the undamped case

When considering the coupled system, the results become more interesting. First of all, Fig. (8.7) shows that the coupling is not negligible only for corresponding modes. The entity of this coupling is due to the spatial inner product of electric and modal shapes. Note that for the case under analisys the mechanical and the electric modal shapes are the same. The orthogonality of these modal

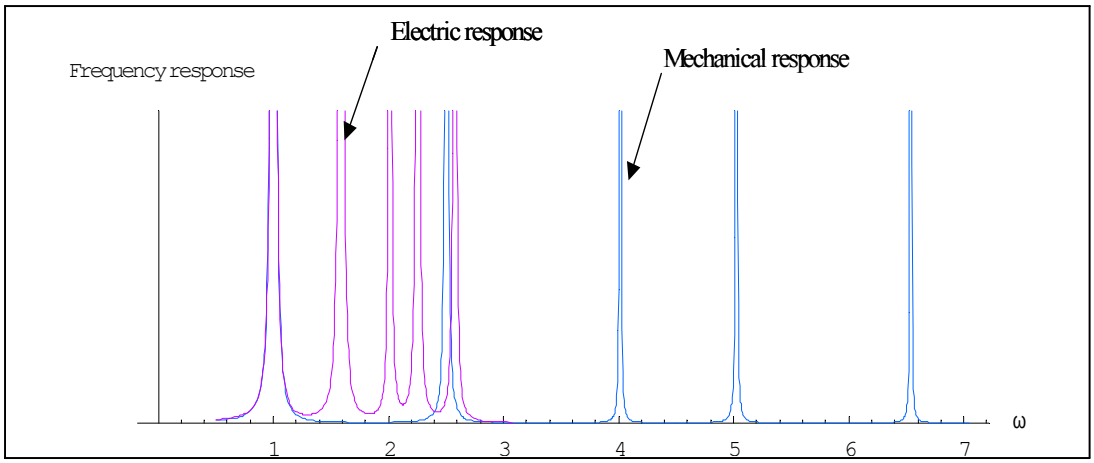

Figure 8.5: Graphical representation of the electric and mechanical eigenfrequencies

shapes implies that the modal coupling table shown in Fig. (8.7) is diagonal (except for the modes whose eigenvalue has an algebraic multiplicity greater than one - this is essentially due to the symmetry of the system).

Now let us give a more expressive interpretation to the introduced concept of coupling. Let us consider a purely mechanical initial condition; for instance, let us assign initial displacements on the first mechanical mode. As expected, due to the transducing effect of the PZT actuators, the initial mechanical energy is converted into an electrical form, then it returns to the mechanical system and so on. This effect involves all the energy initially provided to the system; therefore we say that the coupling is *maximum*. This is due to the fact that the considered mode is tuned, that is it has the same mechanical and electric eigenfrequencies. If we assign initial conditions describing the second mechanical mode (which is

Electric eigenfrequencies			Mechanical eigenfrequencies		
Numerical results	Exact analytical values	Error (percentage)	Numerical results	Exact analytical values	Error (percentage)
1.00258	1.	0.257579	1.00101	1.	0.10068
1.58879	1.58114	0.483655	2.50575	2.5	0.23016
1.58891	1.58114	0.491237	2.50658	2.5	0.263246
2.01443	2	0.72163	4.0151	4.	0.377516
2.25539	2.23607	0.864222	5.0233	5.	0.465981
2.25583	2.23607	0.883647	5.02463	5.	0.492668
2.57761	2.54951	1.10205	6.53656	6.5	0.56244
2.57802	2.54951	1.11842	6.54182	6.5	0.643345

Figure 8.6: Electric and mechanical eigenfrequencies: comparisons with the analytical results

untuned), the amount of energy interchanged is much less than in the previously considered case. Fig. (8.8) shows all these aspects.

Now let's consider the time evolution of the mechanical and electric displacements[1]. Since the corresponding energy is oscillating, we will observe a modulation of the vibrational amplitudes of both systems, with a 90 degree phase shift between them. In the first considered case (initial conditions on the first mechanical mode), this modulation is actually a complete beating effect. The mechanical vibration, starting from maximum amplitude, decreases to a zero value and than grows up again and vice versa for the electric vibration. An example of this phenomenon is given in Fig. (**??**): the time evolution of the electric and mechanical displacements of the central node of the plate is shown when initial conditions on

[1]Let's recall that the electric kinematical descriptor, which we call electric displacement, is a time integral of the voltage.

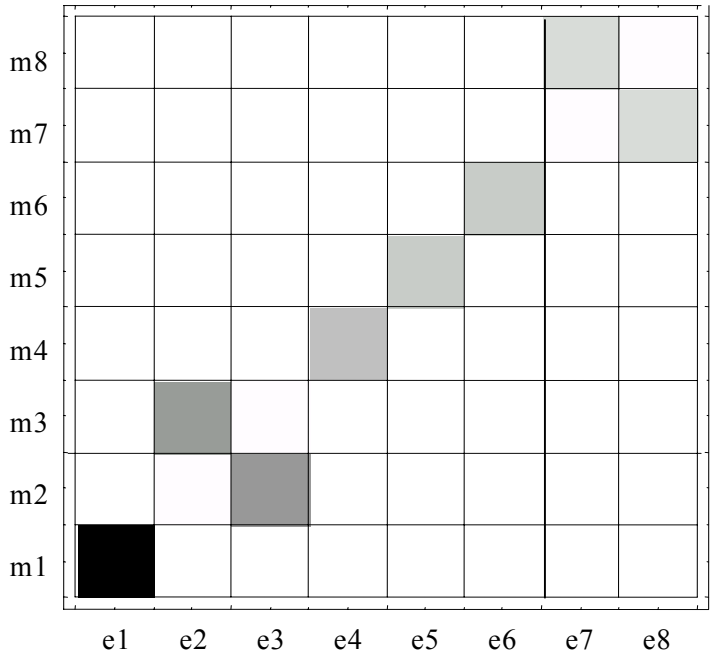

Figure 8.7: Modal coupling table: black means maximum coupling, white means vanishing coupling

the first mechanical mode are assigned.

Tuned coupled system: damping mechanical vibrations by means of a dissipative network

This is probably the main practical application of this distributed control technique. The idea is the following: if the mechanical energy is converted into an electrical form, we could dissipate it with suitable resistors, so that only a fraction of it will return to mechanical form. After some cycles of energy interchange, the electromechanical vibrations damp out. Obviously, an optimum value of the

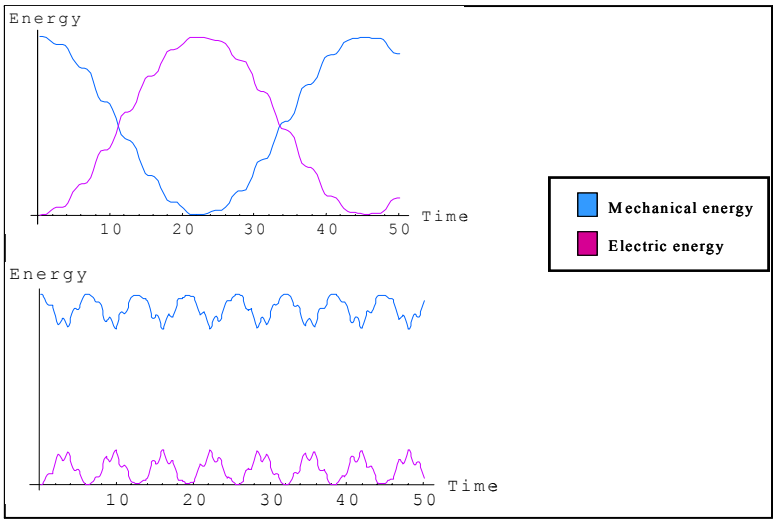

Figure 8.8: Energy time evolution for intial conditions on the a) first mechanical mode; b) second mechanical mode

network resistance can be found, so that the mechanical damping ratio will be maximized; in other words, using the critical value for the resistance, one can obtain that, once the mechanical energy has been converted in the electric form, the damping is such that this energy will not return to the mechanical system. According to ([18]), the critical value for resistance depends on the vibrational mode we want to damp out. In other words, once decided which mode we want to control and damp, both values of network inductance and network resistance are fixed.

Fig. (8.10) shows the time evolution of the energy of the system, tuned for the first eigenfrequency, starting with first mode initial conditions, with the critical damping value suggested in ([18]).

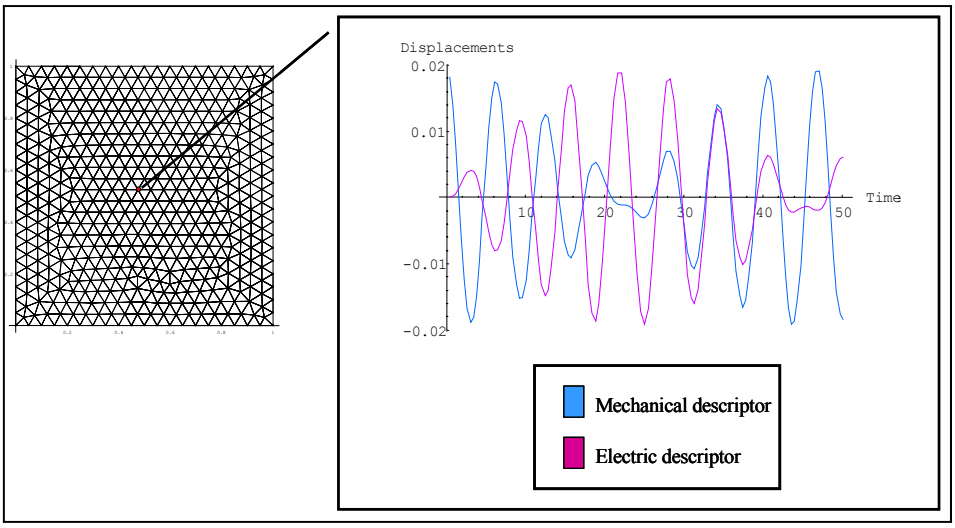

Figure 8.9: Time evolution of the electric and mechanical displacements of a node of the mesh; initial conditions on the first mechanical mode.

Second case: a complex geometry

Following the same steps as the previous section, let us analyze numerically a more complex case. Let us consider an electromechanical 2-dimensional system with the geometry shown in Fig.(**??**), with clamped-grounded edges.

We will consider two different cases of tuning: on the first and on the third mode. In this way we will give an idea of the program features, even if much more general cases could be considered. Let us consider, for instance, that anisotropic inhomogeneous electromechanical continua could be investigated, as well as any other system whose mathematical model can be arranged in the proper weak formulation. Nevertheless, this example will show how the program can be a useful tool for the design of electromechanical devices.

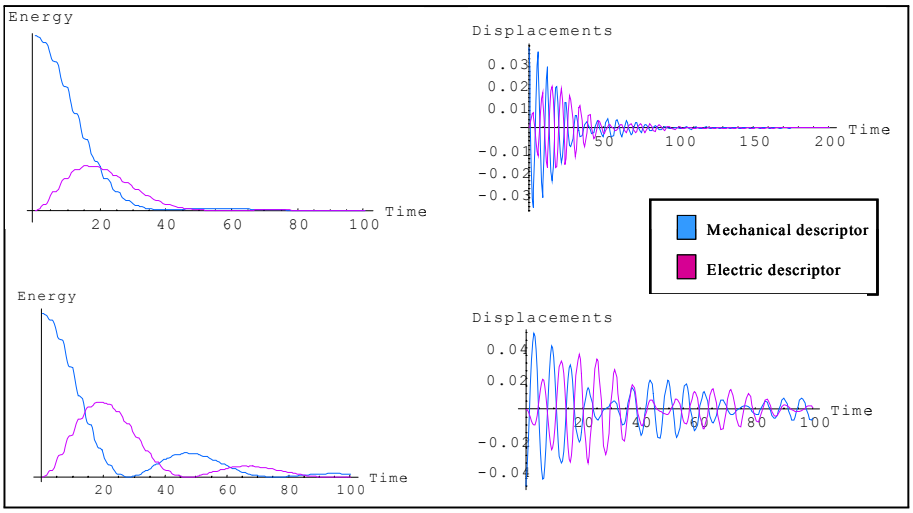

Figure 8.10: Evolution from purely mechanical initial conditions (first mode) of the damped system: 1° row) optimal resistance value; 2° row) low resistance value.

Uncoupled systems

As it has been described in the previous example, the analysis of the electric and mechanical uncoupled systems is the first step of the process. Here we can verify the accuracy of the tuning procedure, as well as check the physical plausibility of the modal shapes. Obviously, in this case we have no analytical solutions to compare the results.

As we can see in Fig. (8.12) or in Fig. (8.13), the tuning procedure was efficient, so that, for instance, the first (or the fourth) eigenfrequency is almost the same for both systems. We remark explicitly that in a similar way it is possible to tune any single pair of electric and mechanical frequency, even if other internal resonances may arise only by chance. Moreover, it is evident that the

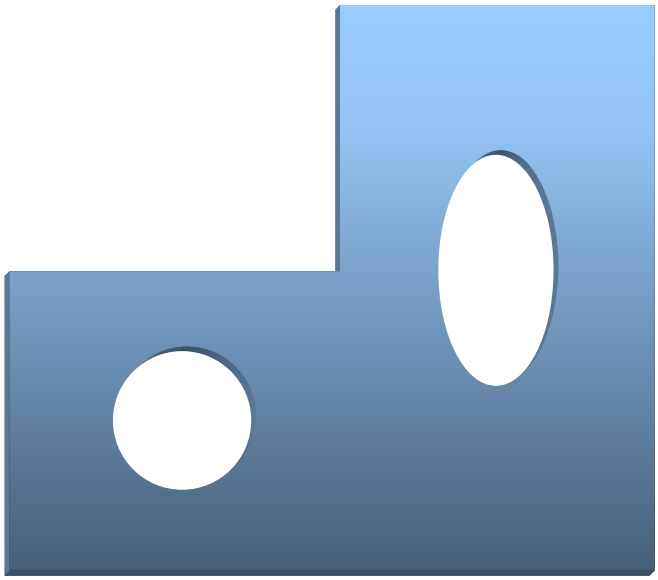

Figure 8.11: Shape of the device

mechanical eigenfrequencies are more closely spaced than in the previous example (the simply supported square plate). This is due to the different geometry and boundary conditions (now the plate is clamped on each edge). As a by-product, this fact has the consequence that also other eigenfrequencies can happen to be almost tuned. Therefore we may expect better coupling performances from this device.

Furthermore, the modal shapes of the electric and mechanical systems are now much more different (because of the clamped edges we cannot impose zero slopes on the boundaries for a second order system such as the electric one), viz. Figs. (8.14) and (8.15).

Hence the coupling will be probably more complicated; that is, we expect each

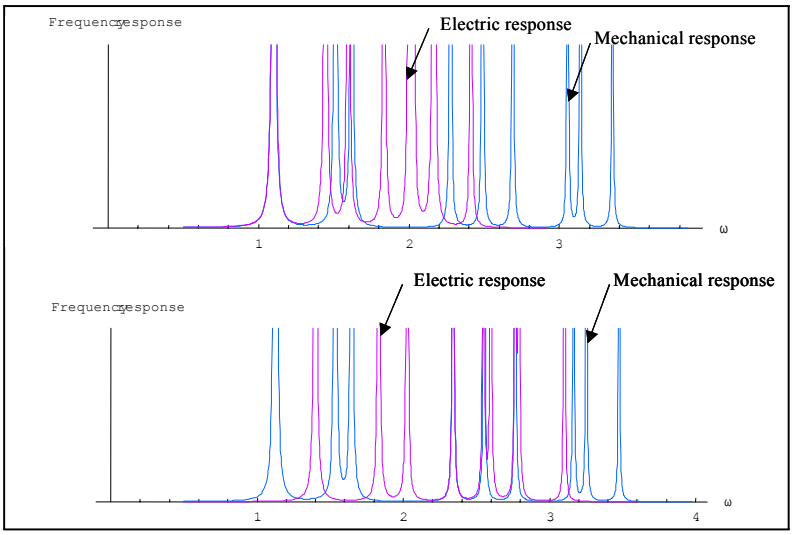

Figure 8.12: Frequency responses of the electric and mechanical uncoupled systems: 1° row) electric system tuned to the first mechanical eigenfrequency; 2° row) electrioc system tuned to the fourth mechanical eigenfrequencies.

electric mode to be coupled with more than one mechanical mode, and viceversa.

In fact, as deduced from Fig. (8.16), we note that there is a distributed modal coupling: the matrix of inner products between electric and mechanical eigenvectors is not anymore diagonal. This fact has important consequences in the dynamics of the whole system. In fact, if we supply energy to a given mechanical mode, this will couple with a large number of electric modes, so that, once the energy returns in the mechanical form, a multi-modal motion appears. However, this effect is limited by the fact that only tuned modes have a complete coupling.

1° MODE TUNED		3° MODE TUNED	
Electric eigenfrequencies	**Mechanical eigenfrequencies**	**Electric eigenfrequencies**	**Mechanical eigenfrequencies**
1.10138	1.10138	1.39896	1.10138
1.44531	1.51307	1.83251	1.51307
1.59817	1.6201	2.02668	1.6201
1.83542	2.27661	2.27662	2.27661
2.00085	2.49051	2.54818	2.49051
2.03117	2.69161	2.59845	2.69161
2.16068	3.05705	2.76599	3.05705
2.1727	3.14112	2.79085	3.14112
2.41381	3.35475	3.10144	3.35475

Figure 8.13: Table of eigenfrequencies

Energy exchange in coupled system: the undamped case

As we expected from the previous analysis, now there is a good coupling in the first few modes, since the first eigenfrequencies are close enough. This fact will have some important consequences in the performances of the system when used as a vibration damper.

Let us firstly consider the case of tuning on the first eigenfrequency (viz. Figs. (8.17) and (8.18)). Again, as a by-product, we find that also the third eigenfrequency comes out to be almost tuned, so that we have a very good energy interchange when exciting the third mechanical mode. The second mode, for the same reason, presents an appreciable coupling yet, but starting from the fourth mode the coupling performances decrease rapidly since the eigenfrequencies of the electric and mechanical sub-systems become very different.

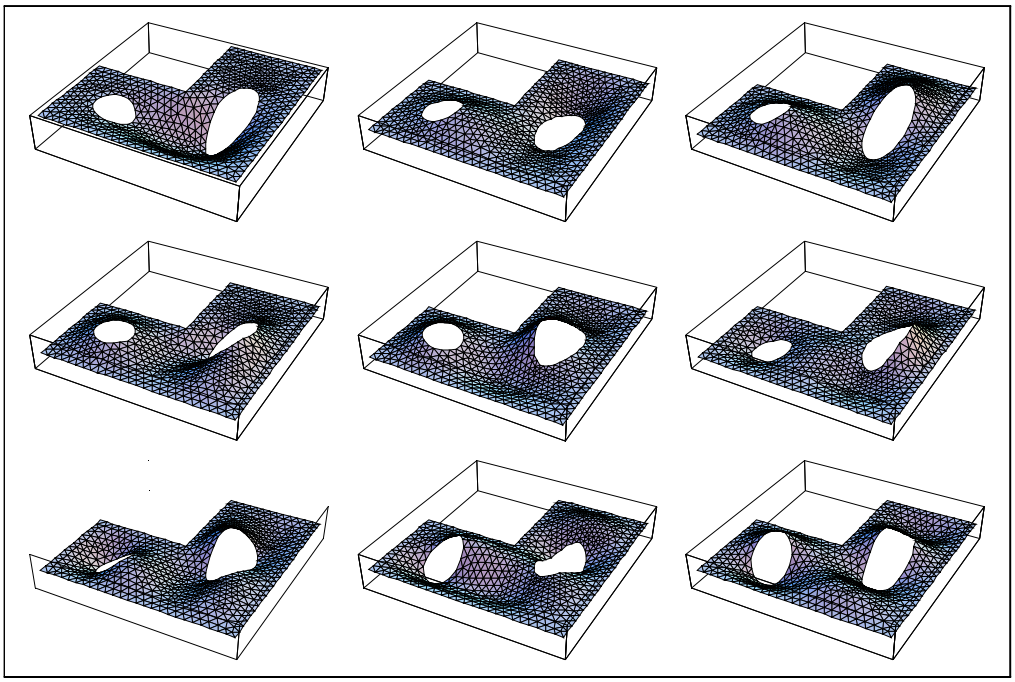

Figure 8.14: Mechanical modal shapes

Now we consider an impact displacement on a point of the structure. Projecting on the chosen base of uncoupled modes, we get a superposition of all the considered modes. The coefficients of such superposition depend upon the point of application of the impulse. Therefore, also the coupling performances are a function of the impulse placement.

Fig. (8.18) shows this concept. In the first case, the impulse is localized in a central area of the plate, so that we excite mostly the first mechanical mode, which is efficiently coupled. Hence we get a good ratio of energy interchange. Instead, if the impulse is applied in a peripheral area of the plate, as shown in the second column of Fig. (8.18), we excite higher modes which are not tuned and therefore

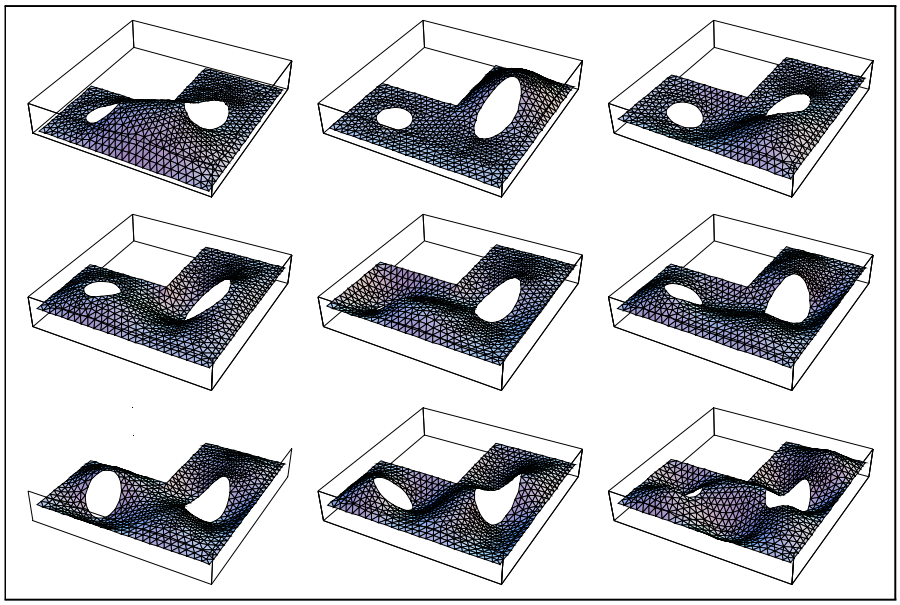

Figure 8.15: Electric modal shapes

not efficiently coupled. This fact implies that a smaller part of mechanical energy is converted into the electric form. However, the performances are still acceptable.

Analogous reasoning can be developed for a different tuning of the systems. Figs. (8.19) and (8.20) show, in the case of tuning on the fourth mode, the behaviour of the considered PEM plate when vibration starts from mechanical uni-modal initial conditions or impulsive initial conditions, respectively.

Coupled system: damping mechanical vibrations by means of a dissipative network

If we add some damping to the electric network, the whole system will not be conservative anymore. Hence this could be a way to dissipate the mechanical

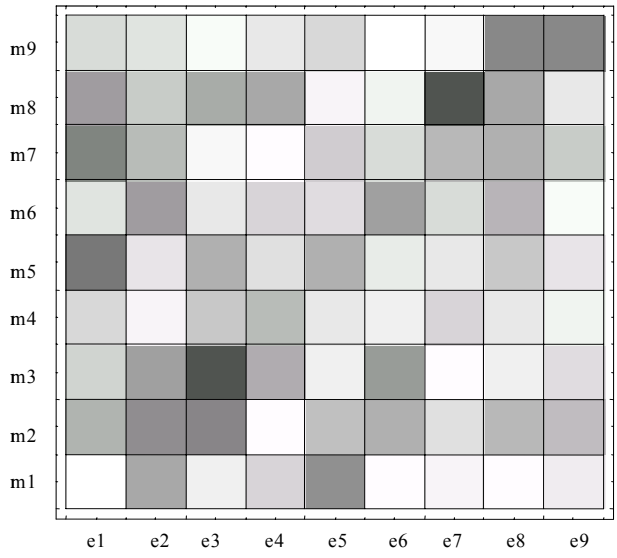

Figure 8.16: Table of modal coupling

energy once converted into the electric form. However, as we noticed in the square plate case, the damping ratio must be maximized by choosing the optimum value for the network resistance, otherwise the damping performances of the system would be unacceptable. Nevertheless, while the analytical studies in ([18]) provides that optimum value for the simply supported square plate, in general cases we have to numerically explore the system to get an approximate value for the optimal resistance. This is exactly what is done here. For this analysis we consider the case of tuning on the first mode, which seems to have better performances. Performing numerical experiments, we can estimate in an iterative fashion the maximum mechanical damping ratio, as shown in Fig. (8.21) where we present the cases of sub-critical damping (first line), critical damping (second line) and

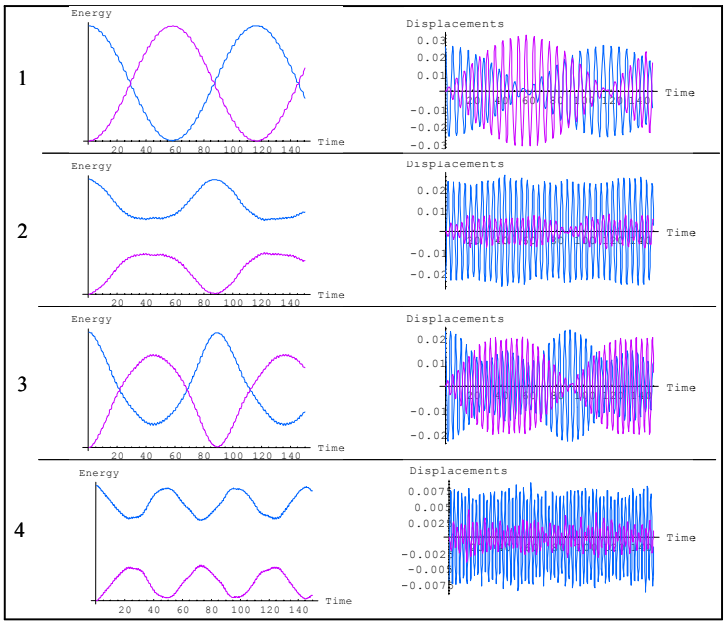

Figure 8.17: Time evolution from different uni-modal initial conditions.

super-critical damping value (third line). Let us note that, if the numerical value of the resistance is too large, the coupling vanishes and, while the electric vibrations decrease rapidly, the mechanical ones present a very low damping ratio. Once the optimum value for the network resistance has been determined, we can analyze the overall damping performances with the usual set of initial conditions: uni-modal (first four modes) and impulsive (two different impulse placements).

The numerical study of this device leads to some useful results from an engineering point of view. We can summarize them in the following two main areas:

1. Firstly, we obtained the optimum network parameters, which are necessary in designing such devices;

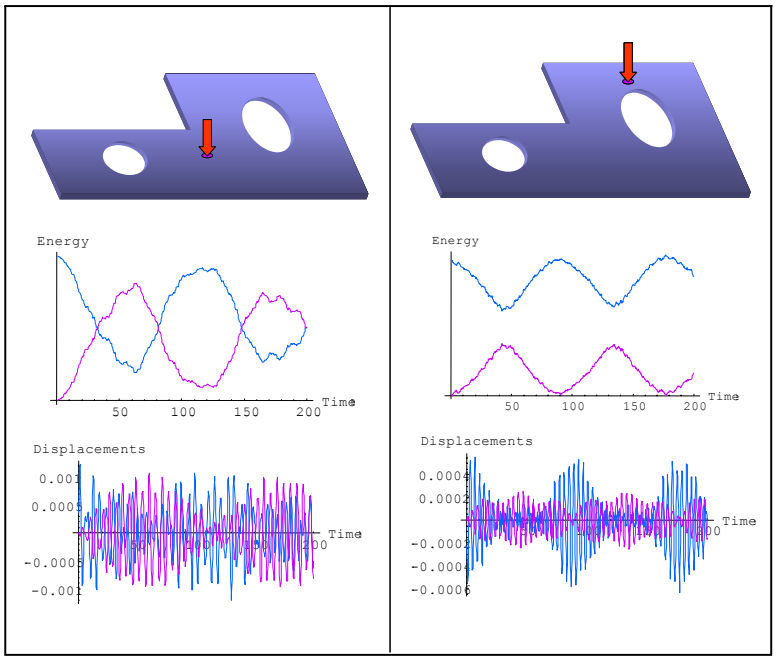

Figure 8.18: Time evolution from impulsive initial conditions.

2. secondly, we got an accurate analysis of the system dynamics, which allows us to forecast its behavior and its performance.

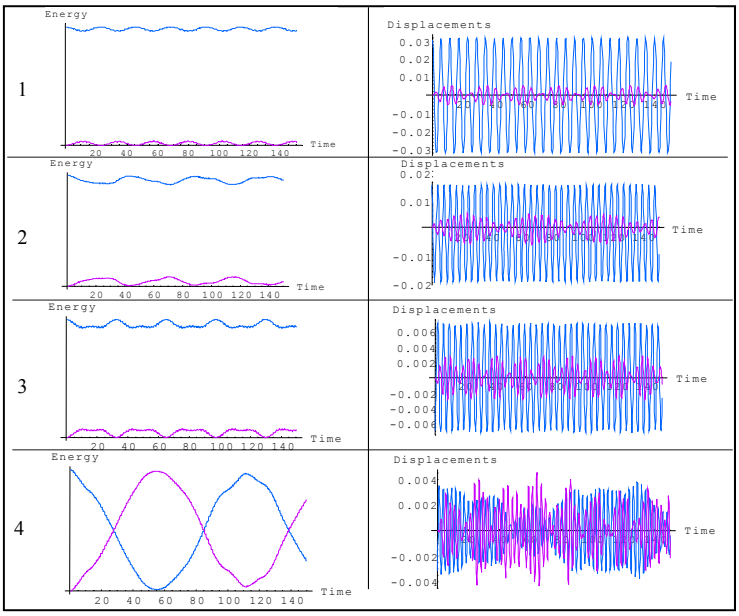

Figure 8.19: Time evolution from uni-modal initial conditions; fourth mode tuned.

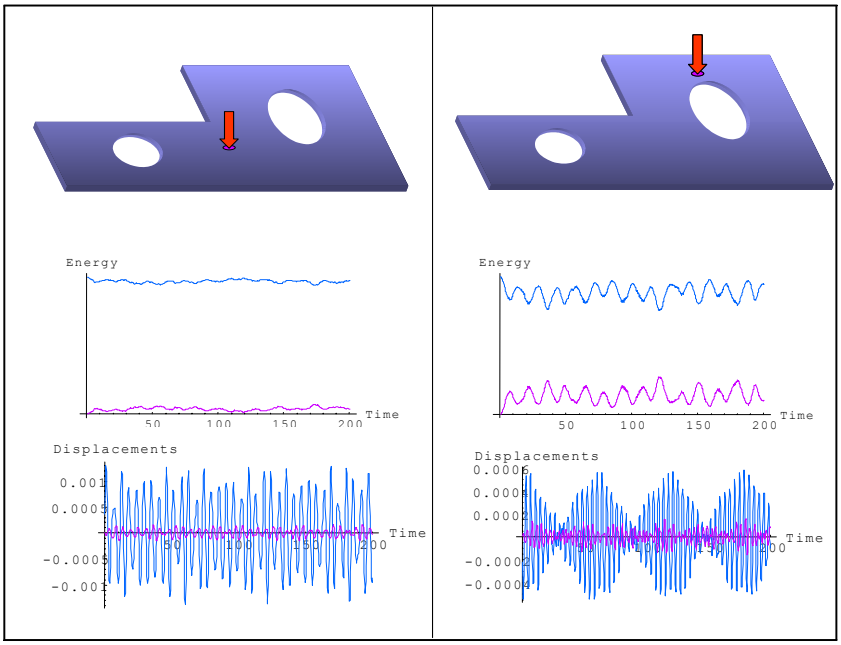

Figure 8.20: Time evolution from impulsive initial conditions; fourth mode tuned.

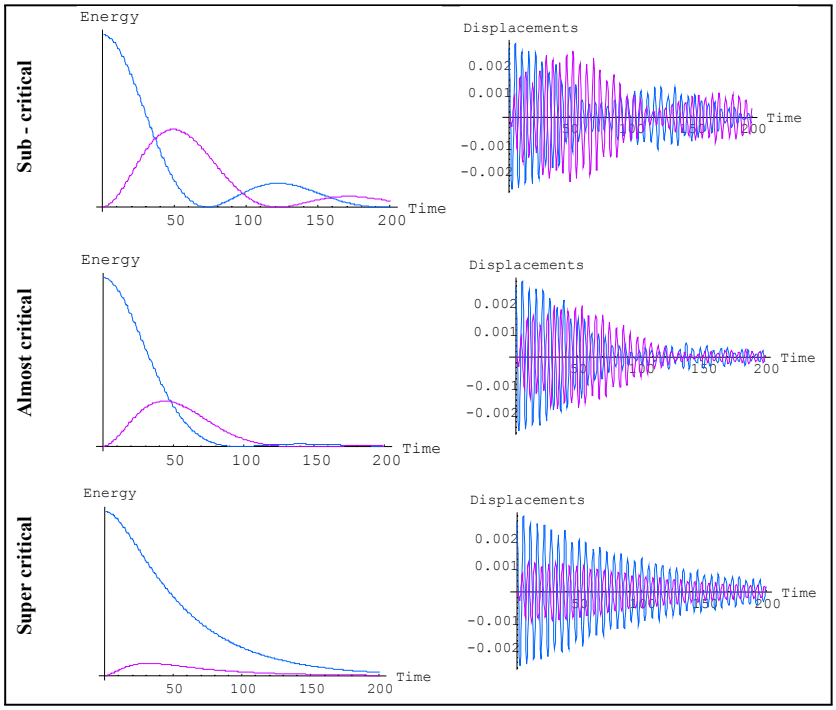

Figure 8.21: Time evolution from first mode initial conditions; three different damping ratios.

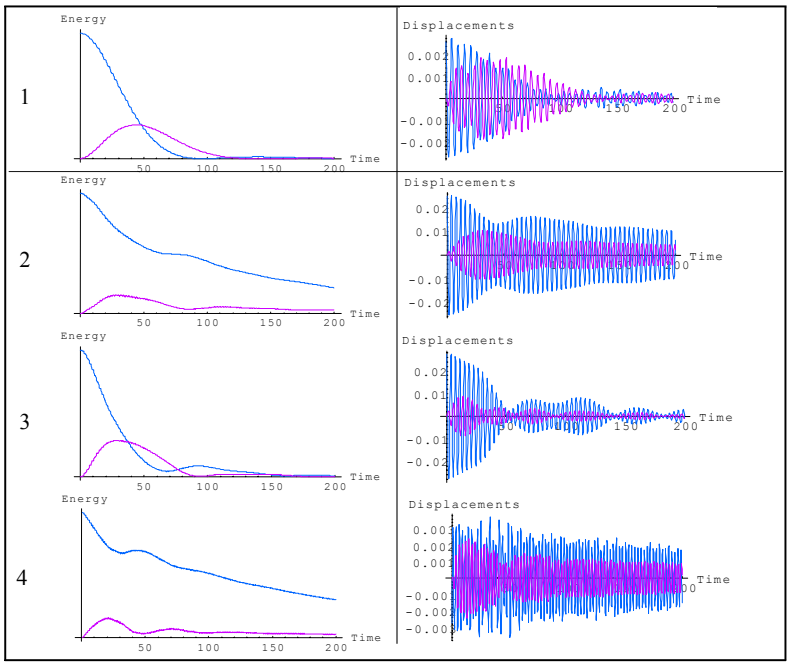

Figure 8.22: Time evolution from uni-modal initial conditions: damped case

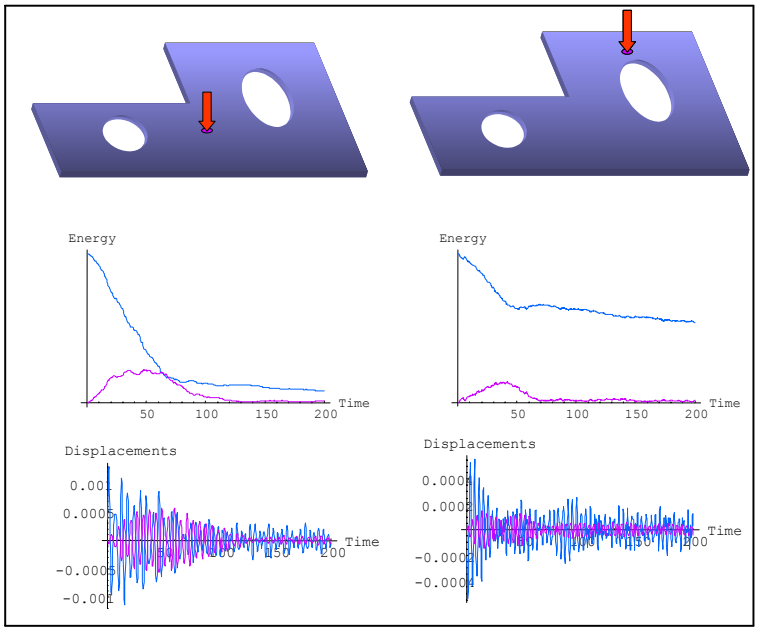

Figure 8.23: Time evolution from impulsive initial conditions: damped case

Chapter 9

Concluding remarks

A new finite element code has been written. Its wide range of applications is probably its best feature, as it allows for the analysis of coupled systems of second order and fourth order PDEs, with dissipative terms. The usage of the code is simple and user-oriented, with few control parameters to be set to control the accuracy and its main aspects. As many procedures as possible have been made completely automatic, so that changing some input parameter is easy and fast. All these characteristics have a price, which is the CPU time needed to perform the computations. Obviously, a dedicated software, realized to solve only a class of particular problems, can easily be optimized and it can reach a higher speed requiring less hardware power. Nevertheless systems involving two or more PDEs of different nature coupled in various fashions require a very flexible computational method.

The program has been applied to the study of PEM plates [18] in which vibration control is obtained by means of piezoelectric actuation. A weak formulation of the evolution equation of such systems has been derived and the data input needed by the code have been computed. A test case, i.e., a simply supported

square PEM plate, has been used to prove the algorithm correctness and the accuracy of the results. Subsequently a more general case has been investigated, and the overall performances of the system have been shown. The code has proven to be useful in the design of such electromechanical systems, since it allowed us to compute the electric parameters needed to optimize the mechanical vibration control and damping.

The obtained results i) show that in a fixed frequency range piezoelectric actuators, interconnected by means of simple passive electric networks can control mechanical vibrations and ii) seem to indicate that more sophisticated electronic circuits could be similarly efficient in a broader band of frequencies.

Appendices

A. The finite element method

Integral or *weak* statements equivalent to the differential equations

Posing the problem to be solved in most general terms, we find that we seek an unknown function $\mathbf{u}$ such that it satisfies a certain differential equation set:

$$\mathbf{A}(\mathbf{u}) = \left\{ \begin{array}{c} A_1(\mathbf{u}) \\ A_2(\mathbf{u}) \\ \vdots \end{array} \right\} = \mathbf{0} \tag{A.1}$$

in a *domain* (volume, area, etc.) Ω, together with certain boundary conditions

$$\mathbf{B}(\mathbf{u}) = \left\{ \begin{array}{c} B_1(\mathbf{u}) \\ B_2(\mathbf{u}) \\ \vdots \end{array} \right\} = \mathbf{0} \tag{A.2}$$

on the boundaries Γ of the domain.

As the set of differential equations has to be zero at each point of the domain

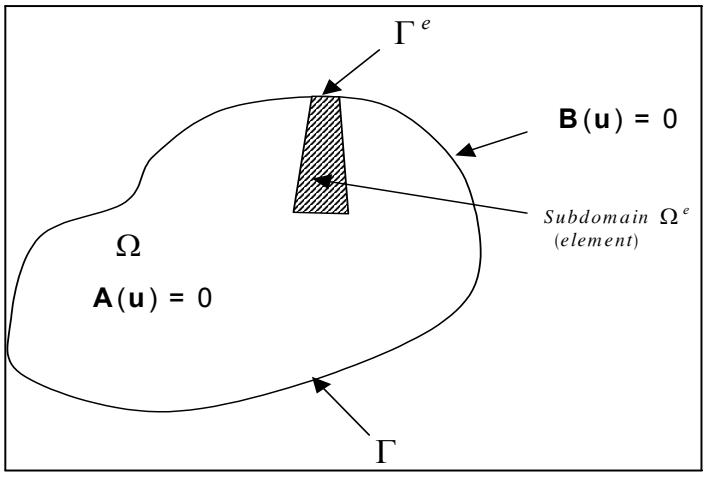

Figure 9.1: Domain description

Ω, it follows that

$$\int \mathbf{v}^T \mathbf{A}(\mathbf{u})\, d\Omega \equiv \int [v_1 A_1(\mathbf{u}) + v_2 A_2(\mathbf{u}) + \cdots]\, d\Omega \equiv 0 \tag{A.3}$$

where

$$\mathbf{v} = \left\{ \begin{array}{c} v_1 \\ v_2 \\ \vdots \end{array} \right\}$$

is a set of arbitrary functions equal in number to the number of equations (or components of $\mathbf{u}$) involved.

The statement is, however, more powerful. We can assert that, if eq (A.3) is satisfied for all $\mathbf{v}$, then the differential equation (A.1) must be satisfied at all points of the domain.

If the boundary conditions are to be simultaneously satisfied,then either we

ensure such satisfaction by the choice of a function $\hat{\mathbf{u}}$ or require that

$$\int_\Gamma \mathbf{v}^T \mathbf{B}(\mathbf{u})\, d\Gamma \equiv \int_\Gamma [v_1\, B_1(\mathbf{u}) + v_2\, B_2(\mathbf{u}) + \cdots]\, d\Gamma \equiv 0$$

for any set of functions $\mathbf{v}$.

Indeed, the integral statement that

$$\int_\Omega \mathbf{v}^T \mathbf{A}(\mathbf{u})\, d\Omega + \int_\Gamma \bar{\mathbf{v}}^T \mathbf{B}(\mathbf{u})\, d\Gamma = 0 \tag{A.4}$$

is satisfied for all $\mathbf{v}$ and $\bar{\mathbf{v}}$ is equivalent to the satisfaction of the differential equations (A.1) and their boundary conditions (A.2).

In the above discussion it was implicitly assumed that integrals such as those in Eq. (A.4) are capable of being evaluated. This places certain restrictions on the possible families to which the functions $\mathbf{v}$ or $\mathbf{u}$ must belong. In general we shall seek to avoid functions which result in any term in the integrals becoming infinite. It is easy to see that if nth-order derivatives occur in any term of A or B then the function has to be such that n-1 derivatives are continuous.

On many occasions it is possible to perform an integration by parts on Eq. (A.4) and replace it by an alternative statement of the form

$$\int_\Omega \mathbf{C}(\mathbf{v})^T \mathbf{D}(\mathbf{u})\, d\Omega + \int_\Gamma \mathbf{E}(\bar{\mathbf{v}})^T \mathbf{F}(\mathbf{u})\, d\Gamma = 0 \tag{A.5}$$

In this the operators $\mathbf{C}$ to $\mathbf{F}$ contain lower order derivatives than those occurring in operators $\mathbf{A}$ and $\mathbf{B}$.

Now a lower order of continuity is required in the choice of the $\mathbf{u}$ function at a price of higher continuity for $\mathbf{v}$ and $\bar{\mathbf{v}}$.

The statement (A.5) is now more "permissive" than the original posed problem and is called a *weak* form of these equations. It is a somewhat surprising fact that often this weak form is more realistic physically than the original differential equation which implied an excessive 'smoothness' of the true solution.

Integral statements of the form of (A.4) and (A.5) form the basis of finite element approximations. We note that the *virtual work principle*, which is often assumed to be so basic as not to merit proof, is simply a weak form of equilibrium equations. From now on, we will assume that the physical system under analysis is described via a virtual work principle.

The Galerkin method of approximation

If the unknown function $\mathbf{u}$ is approximated by the expansion:

$$\mathbf{u} \approx \hat{\mathbf{u}} = \sum \mathbf{N}_i \mathbf{a}_i = \mathbf{N}\mathbf{a}$$

then it is clearly impossible to satisfy both the differential equations and the boundary conditions in a general case. The integral statements (A.4)and (A.5) allow an approximation to be made if, in place of *any function* $\mathbf{v}$, we put a finite set of prescribed functions

$$\mathbf{v} = \mathbf{w}_j \qquad \bar{\mathbf{v}} = \bar{\mathbf{w}}_j \qquad j = 1 \text{ to } n$$

where n is the number of unknown parameters $\mathbf{a}_i$ entering the problem.

Equations (A.4)and (A.5) thus yield a set of ordinary equations from which

parameters $\mathbf{a}$ can be determined, i.e. for Eq. (A.4) we have a set

$$\int_{\Omega} \mathbf{w}_j^T \, \mathbf{A}(\mathbf{Na}) \, d\Omega + \int_{\Gamma} \bar{\mathbf{w}}_j^T \, \mathbf{B}(\mathbf{Na}) \, d\Gamma = 0 \qquad j = 1 \text{ to } n$$

or, from Eq. (A.5),

$$\int_{\Omega} \mathbf{C}(\mathbf{w}_j)^T \, \mathbf{D}(\mathbf{Na}) \, d\Omega + \int_{\Gamma} \mathbf{E}(\bar{\mathbf{w}}_j)^T \, \mathbf{F}(\mathbf{Na}) \, d\Gamma = 0 \qquad j = 1 \text{ to } n$$

Clearly, almost any set of independent functions $\bar{\mathbf{w}}_j$ could be used and, according to the choice of function, a different method can be achieved. In the Galerkin method $\mathbf{w}_j = \mathbf{N}_j$, i.e. simply the original shape functions are used as weighting. This method will be adopted in our finite element work.

In the approximation to the problem of solving the differential equation by an expansion of the standard form, we have assumed that the shape functions $\mathbf{N}$ included in them are all independent coordinates of the problem and that $\mathbf{a}$ was simply a set of constants. The final approximation equations were thus always of an algebraic form, from which a unique set of parameters could be determined.

In some problems it is convenient to proceed differently. Thus, for instance,if the independent variables are x, y and t we could allow the parameters $\mathbf{a}$ to be functions of t and to do approximate expansion only in the domain of x, y, , say Ω. Thus we would have

$$\begin{aligned} \mathbf{u} &= \mathbf{Na} \\ \mathbf{N} &= \mathbf{N}(x, y) \\ \mathbf{a} &= \mathbf{a}(t) \end{aligned}$$

Clearly the derivatives of $\mathbf{a}$ with respect to t will remain in the final discretization and the result will be a set of ordinary differential equations with t as the independent variable. In linear problems such a set will have the appearance

$$\mathbf{K}\,\mathbf{a} + \mathbf{C}\,\dot{\mathbf{a}} + \mathbf{M}\,\ddot{\mathbf{a}} + \cdots + \mathbf{F} = \mathbf{0}$$

where $\dot{\mathbf{a}} \equiv \dfrac{d\,\mathbf{a}}{d\,t}$.

B. Some remarks on fourth order elements

The PEM structures we have analyzed are coupled systems composed by a second order electric continuum (realized by lumped approximation) and a fourth order mechanical continuum. The fourth order differential equations that arise when studying the flexural wave propagation in thin plates have been deeply analyzed, from a numerical point of view, by a large number of researchers. The problems which arise when building a conforming element (i.e. with C^1 shape functions) have been faced in order to find the most convenient solution. A good *trick* has been proposed: why do not we use the Mindlin thick plate model, which leads to second order differential equations, also in the thin limit? The advantages should be considerable in terms of degree of the shape functions needed for the approximation. Unfortunately, the *shear locking* problem does not allow such a simple solution. The plate constitutive relations give us an infinite value for the shear stiffness in the thin limit. This fact can easily be handled from a theoretical point of view leading to the Kirchhoff thin plate model, where shear deformations are assumed to be zero. Nevertheless, numerically we have big troubles. In fact, when the thin limit is approached, the shear stiffness approaches infinity, while the shear deformations, for the numerical approximations, will not be zero. Therefore a large amount of energy will be stored in the shear deformation, and little energy will remain for the vertical displacement. The element will *lock*. Many solutions for this problem have been proposed, based upon different weak formulations

(mixed variational principles) or linked interpolations. However, those solutions, though providing good results both for thin and thick plates are quite complex. Furthermore this method is not convenient for our goals. As a matter of fact, we want a flexible program capable of solving all further systems, while this technique can be applied only to flexural wave propagation. For all these reasons we decided to use the *non-conforming* element described in chapter 4.

C. Why did I use *Mathematica*?

Mathematica is a flexible and intuitive programming language. Its high-level programming constructs let one build sophisticated programs very quickly. Moreover, *Mathematica* programs can mix numerical, symbolic, and graphics operations, with a modular architecture that makes it easy to use as a highly powerful software component. An important feature of *Mathematica*, called *MathLink*, provides a general program-level interface between *Mathematica* and external programs. Those reasons made *Mathematica* my first choice for the main program routine. However, some subroutines are written in different program languages; in particular, the Mesh Generator is a freeware routine realized with *C*, called EasyMesh.

D. Enhancing the code's capability

Since the object of our study is application to smart structures, we can ask ourselves if the numerical analysis we presented can describe with enough accuracy a sufficently large number of practical systems.

Let us recall the initial assumptions about the mathematical model which represents the systems we studied: the code we developed built an approximated solution for structures governed by systems of PDEs, which implies we are assuming for them a continuum model. However in many cases the homogenization procedure that provides us this model from the micro-structure of the systems can cause errors that are not negligible. As a matter of fact, this is what happens in the considered PEM structures when the wave lengths becomes smaller than the network modules constituting the micro-structure of the electric system.

Therefore, sometimes a more refined model is required to describe smart structures. To include those cases, we must improve the code capabilities in order to accept discrete systems.

Furthermore, even if we retain the continuum approach, some improvements can be made in the range of applications which can be considered for the program. In fact, we assumed to have fourth or second order in space PDEs, which can be general enough considering most of the practical applications (we cannot be sure of that), but we assumed also to have only up to second order time derivatives, as in wave propagation problems. It is easy to show that even the PEM structures

we studied with small circuital change will not follow this assumption. In fact, if we replace the net inductance by an impedance constituted by a certain number of inductors and capacitors, we will end up with a system governed by n-th order time derivatives. Hence we should require the numerical procedure to be capable to analyze higher order systems.

Still we are not considering another weak point of the procedure we proposed, that is, we cannot apply it to shells, but only to planar domains. From an industrial point of view, it is therefore crucial to extend the mesh generation and the globalization procedure to include shells as a combination of flat elements.

Finally, many subroutines of the program need to be made faster, because of my lack of experience in algorithmic optimization.

E. The main program

In this appendix the main routine of the program is reported. The code has been divided into a number of short blocks, whose function is described by short remarks. The output of the program is not reported here, since it has been widely presented in the work.

Main

```
Off[General::spell1]
```

```
SetDirectory["C:\\documenti\\mat"];
```

Packages

This load the external subroutines

```
<< "Graphics`Graphics`"
<< "Graphics`Graphics3D`"
<< "getmesh.txt"
<< "LinearAlgebra`MatrixManipulation`"
<< "Graphics`Animation`"
<< "freq.txt"
```

Data Input

Constitutive relations

ü Mechanical parameters

Young module (Pascal)

```
Ey = 70*10^9;
```

Poisson's ratio (dimensionless)

```
v = 0.33;
```

Mass density (Kg/m^2, 1/m^2)

```
ρ = 1;
ρ0 = 2700;
```

```
ρt = ρ0*ρ;
```

Plate thickness (m) = 2 h;

```
h = 0.001;
```

ü Piezo parameters

Piezo density (m):

```
d = 0.1;
```

Mechanical behaviour (Newton)

```
gmm1 = 0.001/d^2;
gmm2 = 0.001/d^2;
gmm12 = 0.001/d^2;
```

Electric parameter (Farad/m^2)

```
gee = 1.*^-6/d^2;
```

Coupling terms

```
gme1 = 0.0001/d^2;
gme2 = 0.0001/d^2;
gme12 = 0/d^2;
```

ü Electrical parameters

Electric circuit (Henry/m^2, Ohm/m^2, Farad/m^2, Ohm^-1/m^2)

```
Rn = 0;
Kc = 0;
Gn = 0.00037;
```

Total net capacitance per unit area:

```
Cn = Kc + gee;
```

Domain definition

Please give the exact number of the closed curves which compose the complete boundary:

```
npieces = 3;
```

ü Instructions

Attention:the curve must be closed,but the end point (=first point) must be given only once in the list!
The points have to be numbered anticlockwise for an external boundary of the domain and clockwise for an internal boundary (for example:a hole).

For each boundary segment please define the boundary condition (1=fixed, 0=free):

- 1 = 0000
- 2 = 0001 ground
- 3 = 0010 supported
- 4 = 0011 supported + ground
- 5 = 0100
- 6 = 0101
- 7 = 0110
- 8 = 0111
- 9 = 1000
- 10=1001
- 11=1010
- 12=1011
- 13=1100
- 14=1101
- 15=1110 clamped
- 16=1111 clamped + ground

ü First part of boundary:

```
pts[1] = {{0, 0}, {2, 0}, {2, 2}, {1, 2}, {1, 1}, {0, 1}};
conditions[1] = {16, 16, 16, 16, 16, 16};
raffinment[1] = 0.15;
marker[1] = 1;
```

ü 2nd part of boundary

```
(nphi = 12; )*(dphi = 2*Pi/nphi; )*
  (pts[2] = Table[{0.5 + 0.2*Cos[-dphi*k], 0.5 + 0.2*Sin[-dphi*k]},
     {k, 0, nphi - 1}]; )*(conditions[2] = Table[1, {k, 0, nphi -
1}]; )*
  (raffinment[2] = 0.15; )*(marker[2] = 1; )
```

ü 3rd part of boundary

```
(nphi = 18; )*(dphi = 2*Pi/nphi; )*
  (pts[3] = Table[{1.5 + 0.2*Cos[-dphi*k], 1 + 0.4*Sin[-dphi*k]},
     {k, 0, nphi - 1}]; )*(conditions[3] = Table[1, {k, 0, nphi -
1}]; )*
  (raffinment[3] = 0.15; )*(marker[3] = 1; )
```

Mesh Generator

Initializations

```
createPointsList[pts_List, raff_, marker_] := Module[{length, npts},
   length = Length[pts]; npts = Table[{pPointer++, pts[[k]], raff,
marker},
      {k, 1, length}]; Return[npts]]
```

```
createSegmentsList[newpts_List, bcond_List] :=
  Module[{length, sgts}, length = Length[newpts];
    sgts = Table[{{newpts[[k,1]], newpts[[k + 1,1]]}, bcond[[k]]},
      {k, 1, length - 1}]; sgts = Append[sgts,
      {{newpts[[length,1]], newpts[[1,1]]}, bcond[[length]]}];
Return[sgts]]
```

```
createfile[bpoints_List, bndry_List, filename_String, header_String]
:=
  Module[{npoints, nsegments, real2string, int2string, n, l1, str1,
k,
    outstream}, npoints = Length[bpoints]; nsegments =
Length[bndry];
    l1 = StringLength[header]; str1 = ""; Do[str1 = StringJoin[str1,
"#"],
     {k, 1, l1 + 4}]; str1 = StringJoin[str1, "\n"];
    real2string[n_] := ToString[PaddedForm[n, {4, 2}]];
    int2string[n_] := ToString[PaddedForm[n, 2]];
    outstream = OpenWrite[filename]; WriteString[outstream, str1];
    WriteString[outstream, "# ", header, " #\n"];
WriteString[outstream,
     str1, "\n"]; WriteString[outstream, "# ======== #", "\n"];
    WriteString[outstream, StringJoin["#| ", "POINTS |#\n"]];
    WriteString[outstream, "# ======== #\n\n"];
WriteString[outstream,
     int2string[npoints], "  # Number of points #\n\n"];
    WriteString[outstream, "# Nodes which define the boundary #\n"];
    Do[WriteString[outstream, int2string[bpoints[[k,1]]], ": ",
      real2string[bpoints[[k,2,1]]], " ",
real2string[bpoints[[k,2,2]]], " ",
      real2string[bpoints[[k,3]]], " ", int2string[bpoints[[k,4]]],
"\n"],
     {k, 1, Length[bndry]}]; WriteString[outstream, "\n# ==========
#",
     "\n"]; WriteString[outstream, StringJoin["#| ", "SEGMENTS
|#\n"]];
    WriteString[outstream, "# ========== #\n\n"];
WriteString[outstream,
     ToString[nsegments], "  # Number of segments #\n\n"];
    WriteString[outstream, "# Boundary segments #\n"];
    Do[WriteString[outstream, int2string[k - 1], ": ",
      int2string[bndry[[k,1,1]]], " ", int2string[bndry[[k,1,2]]], "
",
      int2string[bndry[[k,2]]], "\n"], {k, 1, Length[bndry]}];
    Print["The data file '", filename, "' has been created
successfully."];
    Close[outstream]; ]
```

```
assembleboundary := Module[{pointsList, bpiece, k},
   pPointer = 0; Do[pointsList[k] = createPointsList[pts[k],
raffinment[k],
        marker[k]]; bpiece[k] = createSegmentsList[pointsList[k],
        conditions[k]], {k, 1, npieces}];
    bpoints = Flatten[Join[Table[pointsList[k], {k, 1, npieces}]],
1];
    bndry = Flatten[Join[Table[bpiece[k], {k, 1, npieces}]], 1];
    Print["List of points 'bpoints' has been successfully
created."];
    Print["List of segments 'bndry' has been successfully
created."]; ]
```

```
showboundary[bpoints_List, bndry_List] := Module[{sgmntsList, k,
textsList},
   sgmntsList = Table[Line[{bpoints[[bndry[[k,1,1]] + 1,2]],
        bpoints[[bndry[[k,1,2]] + 1,2]]}], {k, 1, Length[bndry]}];
    textsList = Table[Text[bpoints[[k,1]], bpoints[[k,2]]],
      {k, 1, Length[bpoints]}]; Show[Graphics[{sgmntsList,
textsList}],
     AspectRatio -> Automatic]]
```

Some inputs:

Please give the name of outputfile for EasyMesh and a short header (the header will be written as a comment at the beginning of the datafile):

```
filename = "test.d";
header = "test.d";
```

Assembling pieces of boundary

```
assembleboundary
```

Graphic of actual boundary

```
showboundary[bpoints, bndry]
```

Generation of the datafile

```
createfile[bpoints, bndry, filename, header]
```

```
Pause[5];
```

Generation of the mesh

This call the external routine for mesh generation

```
LinkOpen["easymesh.exe test.d"]
```

```
Pause[15]
```

Visualization of Mesh for Control

```
{nodes, elements, segments} = ReadMesh["./test"];
```

```
ne = Length[elements];
nn = Length[nodes];
```

```
gmesh = MeshGraphics[nodes, elements];
Show[{gmesh}, AspectRatio -> Automatic, Axes -> Automatic, PlotRange
-> All];
```

Tuning

Setup of the tuning parameters; choose the mode you want to tune and give two typical lengths of the domain

```
modo = 1;
precision = 0.01;
maxstep = 10;
la = 2;
lb = 2;
```

```
listamodi = {{1, 1}, {1, 2}, {2, 1}, {2, 2}, {1, 3}, {3, 1}, {2, 3},
{3, 2},
    {1, 4}, {4, 1}};
{a, b} = listamodi[[modo]];
fm = Mecfreq[nodes, elements, segments, Ey, v, h, ρt];
obiett = fm[[modo]];
Ln = 1/(obiett^2*Cn)*((a*Pi/la)^2 + (b*Pi/lb)^2);
f = Elfreq[nodes, elements, segments, Ln, Cn];
err = (obiett - f[[modo]])/obiett;
For[i = 1, i < maxstep, i++, If[Abs[err] < precision, Break,
    Ln = Ln*(f[[modo]]/obiett)^2; f = Elfreq[nodes, elements,
segments, Ln,
       Cn]; err = (obiett - f[[modo]])/obiett]];
```

The net inductance is:

```
Ln
```

Computations

Matrices building

□ $b = g\,\ddot{u} + s\,\dot{u} + t\,u + v\,\hat{e} + b_0$

```
gg[x_, y_] = {{-2*h*ρt, 0}, {0, -Cn*Ln}};
ss[x_, y_] = {{0, 0}, {0, -Cn*Rn - Gn*Ln}};
tt[x_, y_] = {{0, 0}, {0, -Gn*Rn}};
vv[x_, y_] = {{0, 0, 0, 0, 0}, {gme1, gme2, gme12, 0, 0}};
b0[x_, y_, t_] = {0, 0};
```

á f = f0

```
f0[x_, y_, t_] = {0, 0};
```

á s1 = s10

```
s10[x_, y_, t_] = {0, 0};
```

á s2 = s20

```
s20[x_, y_, t_] = {0, 0};
```

á B1 = g1 $\sum_x u$ + B10

```
g1[x_, y_] = {{-(2/3)*h^3*ρt, 0}, {0, 0}};
B10[x_, y_, t_] = {0, 0};
```

□ B2 = g2 $\times_x u$ + B20

```
g2[x_, y_] = {{-(2/3)*h^3*ρt, 0}, {0, 0}};
B20[x_, y_, t_] = {0, 0};
```

□ s = ee + c $\acute{\ddot{u}}$ + r u + s_0

```
ee[x_, y_] = {{(8*Ey*h^3)/(12*(1 - v^2)), (8*Ey*h^3*v)/(12*(1 -
v^2)), 0, 0,
     0}, {(8*Ey*h^3*v)/(12*(1 - v^2)), (8*Ey*h^3)/(12*(1 - v^2)), 0,
0, 0},
    {0, 0, (8*Ey*h^3*(1 - v)/2)/(12*(1 - v^2)), 0, 0}, {0, 0, 0, 1,
0},
    {0, 0, 0, 0, 1}};
```

```
cc[x_, y_] = {{0, Ln*gme1}, {0, Ln*gme2}, {0, Ln*gme12}, {0, 0}, {0,
0}};
```

```
rr[x_, y_] = {{0, Rn*gme1}, {0, Rn*gme2}, {0, Rn*gme12}, {0, 0}, {0,
0}};
```

```
σ0[x_, y_] = {0, 0, 0, 0, 0};
```

□ e = $h_0 u + h_1 \times_x u + h_2 \times_y u + h_3 \times_{x,x} u + h_4 \times_{y,y} u + h_5 \times_{x,y} u$

```
h0[x_, y_] = {{0, 0}, {0, 0}, {0, 0}, {0, 0}, {0, 0}};
h1[x_, y_] = {{0, 0}, {0, 0}, {0, 0}, {0, 1}, {0, 0}};
h2[x_, y_] = {{0, 0}, {0, 0}, {0, 0}, {0, 0}, {0, 1}};
h3[x_, y_] = {{-1, 0}, {0, 0}, {0, 0}, {0, 0}, {0, 0}};
h4[x_, y_] = {{0, 0}, {-1, 0}, {0, 0}, {0, 0}, {0, 0}};
h5[x_, y_] = {{0, 0}, {0, 0}, {-2, 0}, {0, 0}, {0, 0}};
```

```
h0v[x_, y_] = {{0, 0}, {0, 0}, {0, 0}, {0, 0}, {0, 0}};
h1v[x_, y_] = {{0, 0}, {0, 0}, {0, 0}, {0, 1}, {0, 0}};
h2v[x_, y_] = {{0, 0}, {0, 0}, {0, 0}, {0, 0}, {0, 1}};
h3v[x_, y_] = {{-1, 0}, {0, 0}, {0, 0}, {0, 0}, {0, 0}};
h4v[x_, y_] = {{0, 0}, {-1, 0}, {0, 0}, {0, 0}, {0, 0}};
h5v[x_, y_] = {{0, 0}, {0, 0}, {-2, 0}, {0, 0}, {0, 0}};
```

á Degrees of freedom

```
dof = 4;
```

Dimensions parameters

ü Space

Caracteristic length (m)

```
l = 1;
```

ü Time

Caracteristic pulsation (rad/sec)

```
ω = 4*(Pi/l)^2*h*Sqrt[Ey/(12*ρ0*(1 - v^2))]
```

ü Voltage

Caracterstic tension

```
Vc = Sqrt[(2*h*ρ0*l^2*ω^2)/Cn]
```

ü Dimension matrix

```
A = {{l, 0}, {0, Vc/(Ln*ω)}};
```

```
A1 = {{l, 0}, {0, Vc/ω}};
```

Dimensionless formulation

```
gg[x_, y_] = 1/(2*h*l^2*ρ0)*Transpose[A1] . gg[l*x, l*y] . A;
g1[x_, y_] = 1/(2*h*l^4*ρ0)*Transpose[A1] . g1[l*x, l*y] . A;
g2[x_, y_] = 1/(2*h*l^4*ρ0)*Transpose[A1] . g2[l*x, l*y] . A;
ss[x_, y_] = 1/(2*ω*h*l^2*ρ0)*Transpose[A1] . ss[l*x, l*y] . A;
tt[x_, y_] = 1/(2*ω^2*h*l^2*ρ0)*Transpose[A1] . tt[l*x, l*y] . A;
vv[x_, y_] = 1/(2*ω*h*l^2*ρ0)*Transpose[A1] . vv[l*x, l*y];
ee[x_, y_] = 1/(2*ω^2*h*l^2*ρ0)*ee[l*x, l*y];
rr[x_, y_] = 1/(2*ω^2*h*l^2*ρ0)*rr[l*x, l*y] . A;
cc[x_, y_] = 1/(2*ω*h*l^2*ρ0)*cc[l*x, l*y] . A;
b0[x_, y_, t_] = 1/(2*ω^2*h*l^2*ρ0)*Transpose[A1] . b0[l*x, l*y,
t/ω];
B10[x_, y_, t_] = 1/(2*ω^2*h*l^3*ρ0)*Transpose[A1] .
     B10[l*x, l*y, t/ω];
B20[x_, y_, t_] = 1/(2*ω^2*h*l^3*ρ0)*Transpose[A1] .
     B20[l*x, l*y, t/ω];
σ0[x_, y_] = 1/(2*ω^2*h*l^2*ρ0)*σ0[l*x, l*y];
f0[x_, y_, t_] = 1/(2*ω^2*h*l^3*ρ0)*Transpose[A1] . f0[l*x, l*y,
t/ω];
s10[x_, y_, t_] = 1/(2*ω^2*h*l^4*ρ0)*Transpose[A1] .
     s10[l*x, l*y, t/ω];
s20[x_, y_, t_] = 1/(2*ω^2*h*l^4*ρ0)*Transpose[A1] .
     s20[l*x, l*y, t/ω];
h1[x_, y_] = 1/l*h1[l*x, l*y] . A;
h2[x_, y_] = 1/l*h2[l*x, l*y] . A;
h3[x_, y_] = 1/l^2*h3[l*x, l*y] . A;
h4[x_, y_] = 1/l^2*h4[l*x, l*y] . A;
h5[x_, y_] = 1/l^2*h5[l*x, l*y] . A;
h1v[x_, y_] = 1/l*h1v[l*x, l*y] . A1;
h2v[x_, y_] = 1/l*h2v[l*x, l*y] . A1;
h3v[x_, y_] = 1/l^2*h3v[l*x, l*y] . A1;
h4v[x_, y_] = 1/l^2*h4v[l*x, l*y] . A1;
h5v[x_, y_] = 1/l^2*h5v[l*x, l*y] . A1;
```

```
{MatrixForm[gg[x, y]], MatrixForm[ss[x, y]], MatrixForm[tt[x, y]],
  MatrixForm[vv[x, y]], MatrixForm[ee[x, y]], MatrixForm[rr[x, y]],
  MatrixForm[cc[x, y]]}
```

Shape functions and geometry

```
xe[e_, i_] := nodes[[elements[[e,i + 1]],2]]
ye[e_, i_] := nodes[[elements[[e,i + 1]],3]]
```

In the triangular Master element domain we observe that:

```
n[r_, s_, 1] = r;
n[r_, s_, 2] = s;
n[r_, s_, 3] = 1 - r - s;
```

Recall that $r + s + t = 1$

Now let's define

```
Table[{l1[e] = Sqrt[(xe[e, 2] - xe[e, 3])^2 + (ye[e, 2] - ye[e,
3])^2],
    l2[e] = Sqrt[(xe[e, 1] - xe[e, 3])^2 + (ye[e, 1] - ye[e, 3])^2],
    l3[e] = Sqrt[(xe[e, 2] - xe[e, 1])^2 + (ye[e, 2] - ye[e, 1])^2],
    μ1[e] = (l3[e]^2 - l2[e]^2)/l1[e]^2,
    μ2[e] = (l1[e]^2 - l3[e]^2)/l2[e]^2,
    μ3[e] = (l2[e]^2 - l1[e]^2)/l3[e]^2, b1[e] = ye[e, 2] - ye[e,
3],
    b2[e] = ye[e, 3] - ye[e, 1], b3[e] = ye[e, 1] - ye[e, 2],
    c1[e] = xe[e, 3] - xe[e, 2], c2[e] = xe[e, 1] - xe[e, 3],
    c3[e] = xe[e, 2] - xe[e, 1]}, {e, 1, ne}];
```

Thus the shape functions matrix can be stated as follows:

```
P[r_, s_, e_] := {n[r, s, 1], n[r, s, 2], n[r, s, 3], n[r, s,
1]*n[r, s, 2],
    n[r, s, 2]*n[r, s, 3], n[r, s, 3]*n[r, s, 1], n[r, s, 1]^2*n[r,
s, 2] +
     1/2*n[r, s, 1]*n[r, s, 2]*n[r, s, 3]*(3*(1 - μ3[e])*n[r, s, 1]
-
       (1 + 3*μ3[e])*n[r, s, 2] + (1 + 3*μ3[e])*n[r, s, 3]),
    n[r, s, 2]^2*n[r, s, 3] + 1/2*n[r, s, 1]*n[r, s, 2]*n[r, s, 3]*
      (3*(1 - μ1[e])*n[r, s, 2] - (1 + 3*μ1[e])*n[r, s, 3] +
       (1 + 3*μ1[e])*n[r, s, 1]), n[r, s, 3]^2*n[r, s, 1] +
     1/2*n[r, s, 1]*n[r, s, 2]*n[r, s, 3]*(3*(1 - μ2[e])*n[r, s, 3]
-
       (1 + 3*μ2[e])*n[r, s, 1] + (1 + 3*μ2[e])*n[r, s, 2])};
```

```
ns[r_, s_, e_] := {{-b2[e]*(P[r, s, e][[9]] - P[r, s, e][[6]]) -
      b3[e]*P[r, s, e][[7]], -c2[e]*(P[r, s, e][[9]] - P[r, s,
e][[6]]) -
      c3[e]*P[r, s, e][[7]], P[r, s, e][[1]] - P[r, s, e][[4]] +
      P[r, s, e][[6]] + 2*(P[r, s, e][[7]] - P[r, s, e][[9]]), 0,
     -b3[e]*(P[r, s, e][[7]] - P[r, s, e][[4]]) - b1[e]*P[r, s,
e][[8]],
     -c3[e]*(P[r, s, e][[7]] - P[r, s, e][[4]]) - c1[e]*P[r, s,
e][[8]],
     P[r, s, e][[2]] - P[r, s, e][[5]] + P[r, s, e][[4]] +
      2*(P[r, s, e][[8]] - P[r, s, e][[7]]), 0,
     -b1[e]*(P[r, s, e][[8]] - P[r, s, e][[5]]) - b2[e]*P[r, s,
e][[9]],
     -c1[e]*(P[r, s, e][[8]] - P[r, s, e][[5]]) - c2[e]*P[r, s,
e][[9]],
     P[r, s, e][[3]] - P[r, s, e][[6]] + P[r, s, e][[5]] +
      2*(P[r, s, e][[9]] - P[r, s, e][[8]]), 0}, {0, 0, 0, n[r, s,
1], 0, 0,
     0, n[r, s, 2], 0, 0, 0, n[r, s, 3]}};
```

Using the linear basis functions for the transformation map between {x,y} and {r,s} domain we get:

```
x[r_, s_, e_] := Sum[xe[e, i]*n[r, s, i], {i, 1, 3}]
y[r_, s_, e_] := Sum[ye[e, i]*n[r, s, i], {i, 1, 3}]
```

Now we can compute the jacobian matrix:

```
jcbmat[e_] := {{D[x[r, s, e], r], D[x[r, s, e], s]},
    {D[y[r, s, e], r], D[y[r, s, e], s]}};
```

```
j[e_] := Det[jcbmat[e]];
```

Cinematic compatibility

```
dx[n_, e_] := 1/j[e]*(D[n, r]*D[y[r, s, e], s] - D[n, s]*D[y[r, s,
e], r])
dy[n_, e_] := 1/j[e]*(-D[n, r]*D[x[r, s, e], s] + D[n, s]*D[x[r, s,
e], r])
d2x[n_, e_] := dx[dx[n, e], e]
d2y[n_, e_] := dy[dy[n, e], e]
d2xy[n_, e_] := dy[dx[n, e], e]
```

```
ε[r_, s_, e_] := h1[x[r, s, e], y[r, s, e]] . dx[ns[r, s, e], e] +
    h2[x[r, s, e], y[r, s, e]] . dy[ns[r, s, e], e] +
    h3[x[r, s, e], y[r, s, e]] . d2x[ns[r, s, e], e] +
    h4[x[r, s, e], y[r, s, e]] . d2y[ns[r, s, e], e] +
    h5[x[r, s, e], y[r, s, e]] . d2xy[ns[r, s, e], e];
```

```
εv[r_, s_, e_] := h1v[x[r, s, e], y[r, s, e]] . dx[ns[r, s, e], e] +
    h2v[x[r, s, e], y[r, s, e]] . dy[ns[r, s, e], e] +
    h3v[x[r, s, e], y[r, s, e]] . d2x[ns[r, s, e], e] +
    h4v[x[r, s, e], y[r, s, e]] . d2y[ns[r, s, e], e] +
    h5v[x[r, s, e], y[r, s, e]] . d2xy[ns[r, s, e], e];
```

```
nx[r_, s_, e_] := dx[ns[r, s, e], e];
ny[r_, s_, e_] := dy[ns[r, s, e], e];
```

Local computations

```
Table[fa[r_, s_] = ε[r, s, e]; fav[r_, s_] = εv[r, s, e];
    fanx[r_, s_] = nx[r, s, e]; fany[r_, s_] = ny[r, s, e];
    ε1 = fa[0.5, 0.5]; εv1 = fav[0.5, 0.5]; ε2 = fa[0, 0.5];
    εv2 = fav[0, 0.5]; ε3 = fa[0.5, 0]; εv3 = fav[0.5, 0];
    ns1 = ns[0.5, 0.5, e]; ns2 = ns[0, 0.5, e]; ns3 = ns[0.5, 0, e];
    ns4 = ns[0.333333, 0.333333, e]; ns5 = ns[0.6, 0.2, e];
    ns6 = ns[0.2, 0.6, e]; ns7 = ns[0.2, 0.2, e]; nx1 = fanx[0.5,
0.5];
    ny1 = fany[0.5, 0.5]; nx2 = fanx[0, 0.5]; ny2 = fany[0, 0.5];
    nx3 = fanx[0.5, 0]; ny3 = fany[0.5, 0]; nx4 = fanx[0.333333,
0.333333];
    ny4 = fany[0.333333, 0.333333]; nx5 = fanx[0.6, 0.2];
    ny5 = fany[0.6, 0.2]; nx6 = fanx[0.2, 0.6]; ny6 = fany[0.2,
0.6];
    nx7 = fanx[0.2, 0.2]; ny7 = fany[0.2, 0.2]; jac = j[e];
    integrandk01 = Transpose[tt[x[0.5, 0.5, e], y[0.5, 0.5, e]] .
ns1] . ns1*
       jac - Transpose[ee[x[0.5, 0.5, e], y[0.5, 0.5, e]] . ε1] .
εv1*
       jac - Transpose[rr[x[0.5, 0.5, e], y[0.5, 0.5, e]] . ns1] .
εv1*jac;
    integrandk02 = Transpose[tt[x[0, 0.5, e], y[0, 0.5, e]] . ns2] .
ns2*
       jac - Transpose[ee[x[0, 0.5, e], y[0, 0.5, e]] . ε2] .
εv2*jac -
      Transpose[rr[x[0, 0.5, e], y[0, 0.5, e]] . ns2] . εv2*jac;
    integrandk03 = Transpose[tt[x[0.5, 0, e], y[0.5, 0, e]] . ns3] .
ns3*
       jac - Transpose[ee[x[0.5, 0, e], y[0.5, 0, e]] . ε3] .
εv3*jac -
      Transpose[rr[x[0.5, 0, e], y[0.5, 0, e]] . ns3] . εv3*jac;
    integrandk11 = Transpose[ss[x[0.5, 0.5, e], y[0.5, 0.5, e]] .
ns1 +
          vv[x[0.5, 0.5, e], y[0.5, 0.5, e]] . ε1] . ns1*jac -
      Transpose[cc[x[0.5, 0.5, e], y[0.5, 0.5, e]] . ns1] . εv1*jac;
    integrandk12 = Transpose[ss[x[0, 0.5, e], y[0, 0.5, e]] . ns2 +
          vv[x[0, 0.5, e], y[0, 0.5, e]] . ε2] . ns2*jac -
      Transpose[cc[x[0, 0.5, e], y[0, 0.5, e]] . ns2] . εv2*jac;
    integrandk13 = Transpose[ss[x[0.5, 0, e], y[0.5, 0, e]] . ns3 +
          vv[x[0.5, 0, e], y[0.5, 0, e]] . ε3] . ns3*jac -
      Transpose[cc[x[0.5, 0, e], y[0.5, 0, e]] . ns3] . εv3*jac;
    integrandk21 =
     Transpose[gg[x[0.333333, 0.333333, e], y[0.333333, 0.333333,
e]] .
          ns4] . ns4*jac +
```

```
      Transpose[g1[x[0.333333, 0.333333, e], y[0.333333, 0.333333,
e]] .
          nx4] . nx4*jac +
      Transpose[g2[x[0.333333, 0.333333, e], y[0.333333, 0.333333,
e]] .
          ny4] . ny4*jac; integrandk22 =
     Transpose[gg[x[0.6, 0.2, e], y[0.6, 0.2, e]] . ns5] . ns5*jac +
      Transpose[g1[x[0.6, 0.2, e], y[0.6, 0.2, e]] . nx5] . nx5*jac
+
      Transpose[g2[x[0.6, 0.2, e], y[0.6, 0.2, e]] . ny5] . ny5*jac;
    integrandk23 = Transpose[gg[x[0.2, 0.6, e], y[0.2, 0.6, e]] .
ns6] . ns6*
       jac + Transpose[g1[x[0.2, 0.6, e], y[0.2, 0.6, e]] . nx6] .
nx6*jac +
      Transpose[g2[x[0.2, 0.6, e], y[0.2, 0.6, e]] . ny6] . ny6*jac;
    integrandk24 = Transpose[gg[x[0.2, 0.2, e], y[0.2, 0.2, e]] .
ns7] . ns7*
       jac + Transpose[g1[x[0.2, 0.2, e], y[0.2, 0.2, e]] . nx7] .
nx7*jac +
      Transpose[g2[x[0.2, 0.2, e], y[0.2, 0.2, e]] . ny7] . ny7*jac;
    integrandfbody1 = Transpose[εv1] . σ0[x[0.5, 0.5, e], y[0.5,
0.5, e]]*
       jac - Transpose[ns1] . b0[x[0.5, 0.5, e], y[0.5, 0.5, e],
t]*j[e];
    integrandfbody2 = Transpose[εv2] . σ0[x[0, 0.5, e], y[0, 0.5,
e]]*
       jac - Transpose[ns2] . b0[x[0, 0.5, e], y[0, 0.5, e], t]*jac;
    integrandfbody3 = Transpose[εv3] . σ0[x[0.5, 0, e], y[0.5, 0,
e]]*
       jac - Transpose[ns3] . b0[x[0.5, 0, e], y[0.5, 0, e], t]*jac;
    k0[e] = 0.5*(0.333333*integrandk01 + 0.333333*integrandk02 +
       0.333333*integrandk03); k1[e] = 0.5*(0.333333*integrandk11 +
       0.333333*integrandk12 + 0.333333*integrandk13);
    k2[e] = 0.5*(-0.5625*integrandk21 + 0.520833*integrandk22 +
       0.520833*integrandk23 + 0.520833*integrandk24);
    f0[e] = 0.5*(0.333333*integrandfbody1 + 0.333333*integrandfbody2
+
       0.333333*integrandfbody3), {e, 1, ne}];
```

Global stiffness matrices

```
k0g = Table[0, {i, 1, dof*nn}, {j, 1, dof*nn}];
k1g = Table[0, {i, 1, dof*nn}, {j, 1, dof*nn}];
k2g = Table[0, {i, 1, dof*nn}, {j, 1, dof*nn}];
```

```
Table[k0g[[dof*elements[[e,i + 1]] - dof + 1 + l,dof*elements[[e,j +
1]] -
      dof + 1 + m]] += k0[e][[dof*i - dof + 1 + l,dof*j - dof + 1 +
m]],
   {l, 0, dof - 1}, {m, 0, dof - 1}, {i, 1, 3}, {j, 1, 3}, {e, 1,
ne}];
```

```
Table[k1g[[dof*elements[[e,i + 1]] - dof + 1 + l,dof*elements[[e,j +
1]] -
      dof + 1 + m]] += k1[e][[dof*i - dof + 1 + l,dof*j - dof + 1 +
m]],
   {l, 0, dof - 1}, {m, 0, dof - 1}, {i, 1, 3}, {j, 1, 3}, {e, 1,
ne}];
```

```
Table[k2g[[dof*elements[[e,i + 1]] - dof + 1 + l,dof*elements[[e,j +
1]] -
      dof + 1 + m]] += k2[e][[dof*i - dof + 1 + l,dof*j - dof + 1 +
m]],
   {l, 0, dof - 1}, {m, 0, dof - 1}, {i, 1, 3}, {j, 1, 3}, {e, 1,
ne}];
```

Global load vector

ü Assembling local load vectors

```
fg = Table[0, {i, 1, dof*nn}];
```

```
Table[fg[[dof*elements[[e,i + 1]] - dof + 1 + l]] +=
    f0[e][[dof*i - dof + 1 + l]], {l, 0, dof - 1}, {i, 1, 3}, {e, 1,
ne}];
```

ü Boundary external load

```
fgbound = Table[0, {i, 1, dof*nn}];
```

```
fgtot = fg + fgbound;
```

Essential boundary conditions

ü Fixed nodes list building

```
nodi = Table[0, {i, 1, dof}, {j, 1, nn}];
```

```
Table[If[IntegerDigits[segments[[i,3]] - 1, 2, dof][[j]] == 1,
    {nodi[[j,segments[[i,1]]]] = 1, nodi[[j,segments[[i,2]]]] = 1}],
   {i, 1, Length[segments]}, {j, 1, dof}];
```

```
listavincoli = Table[0, {i, 1, nn}, {j, 1, dof}];
```

```
Table[listavincoli[[nn + 1 - i,dof - j + 1]] = nodi[[j,i]], {i, 1,
nn},
   {j, 1, dof}];
```

ü Global matrices reduction

```
Table[If[listavincoli[[i,j]] == 1,
    {k0g = Delete[k0g, dof*nn + dof + 1 - dof*i - j],
     k1g = Delete[k1g, dof*nn + dof + 1 - dof*i - j],
     k2g = Delete[k2g, dof*nn + dof + 1 - dof*i - j],
     fgtot = Delete[fgtot, dof*nn + dof + 1 - dof*i - j]}], {i, 1,
nn},
   {j, 1, dof}];
```

```
k0g = Transpose[k0g];
k1g = Transpose[k1g];
k2g = Transpose[k2g];
```

```
Table[If[listavincoli[[i,j]] == 1,
    {k0g = Delete[k0g, dof*nn + dof + 1 - dof*i - j],
     k1g = Delete[k1g, dof*nn + dof + 1 - dof*i - j],
     k2g = Delete[k2g, dof*nn + dof + 1 - dof*i - j]}], {i, 1, nn},
   {j, 1, dof}];
```

```
k0g = Transpose[k0g];
k1g = Transpose[k1g];
k2g = Transpose[k2g];
```

```
NM = Length[k0g];
```

Modal analisys

ü Eigenvalues and eigenvectors of the uncoupled systems

```
mm = Inverse[k0g] . k2g; NM = Length[k0g];
{vals, vecs} = Chop[Eigensystem[mm], 10^(-6)];
```

ü Joining the boundary fixed nodes with the solution

```
vecs1 = vecs;
test = 30;
Table[If[listavincoli[[nn + 1 - i,dof + 1 - j]] == 1,
    vecs1[[m]] = Insert[vecs1[[m]], 0, dof*i - dof + j]], {i, 1,
nn},
   {j, 1, dof}, {m, 1, test}];
```

ü Eigenvectors decomposition (electric and mechanical)

```
abb = Table[vecs1[[m,dof*i - 1]], {m, 1, test}, {i, 1, nn}];
rotx = Table[vecs1[[m,dof*i - 3]], {m, 1, test}, {i, 1, nn}];
roty = Table[vecs1[[m,dof*i - 2]], {m, 1, test}, {i, 1, nn}];
elec = Table[vecs1[[m,dof*i]], {m, 1, test}, {i, 1, nn}];
normmec = Table[Sum[Abs[abb[[m,i]]]^2 + Abs[rotx[[m,i]]]^2 +
      Abs[roty[[m,i]]]^2, {i, 1, nn}], {m, 1, test}];
normelec = Table[Sum[Abs[elec[[m,i]]]^2, {i, 1, nn}], {m, 1, test}];
```

ü Choosing the basis of eigenvecters

```
nmodi = 9;
```

```
base = {}; freq = {};
ve = 0;
vm = 0;
For[i = 1, i < NM + 1, i++, If[ve == nmodi, Break,
    If[normelec[[i]] > 0.5 && ve < nmodi, base = Append[base,
vecs[[i]]];
      freq = Append[freq, Sqrt[vals[[i]]]]; ve++]]];
For[i = 1, i < NM + 1, i++, If[vm == nmodi, Break,
    If[normmec[[i]] > 0.5 && vm < nmodi, base = Append[base,
vecs[[i]]];
      freq = Append[freq, Sqrt[vals[[i]]]]; vm++]]];
tr = Transpose[base];
freq = 1/freq;
```

ü Projecting the entire system on the choosen basis

```
k0gn = base . k0g . tr;
k1gn = base . k1g . tr;
k2gn = base . k2g . tr;
fgtotn = base . fgtot;
```

Initial conditions problem

ü Initial conditions

```
gmesh = MeshGraphics[nodes, elements];
nodenumbers := LabeledListPlot[Table[{nodes[[i,2]], nodes[[i,3]]},
     {i, 1, nn}], DisplayFunction -> Identity];
Show[{gmesh, nodenumbers}, AspectRatio -> Automatic, Axes ->
Automatic,
   PlotRange -> All];
```

```
position[nodenumber_, dof_] := Module[{cont = 0},
   Table[If[listavincoli[[nn - i + 1,5 - j]] == 1, cont = cont + 1],
     {i, 1, nodenumber}, {j, 1, dof}]; nodenumber*4 - (4 - dof) -
cont]
```

```
X0 = Table[0, {NM}]; V0 = Table[0, {NM}]; V0[[position[200, 3]]] =
1;
```

ü Projecting the initial conditions on the basis

```
Z0 = Table[X0 . base[[i]], {i, 1, 2*nmodi}];
DZ0 = Table[V0 . base[[i]], {i, 1, 2*nmodi}];
```

```
Z0 = Table[0, {i, 1, 2*nmodi}];
DZ0 = Table[0, {i, 1, 2*nmodi}];
Z0[[2 + nmodi]] = 1;
```

ü Setup of the Runge Kutta sub-routine

```
tmax = 200;
timestep = 0.1;
accuracy = 0.0001;
```

ü Creating input files for Runge Kutta sub-routine

```
strm = OpenWrite["Abc"];
Write[strm, 2*nmodi];
Table[WriteString[strm, i, "  ", j, "  ", Part(k2gn,i,j), "\n"],
   {i, 1, 2*nmodi}, {j, 1, 2*nmodi}];
Table[WriteString[strm, i, "  ", j, "  ", Part(k1gn,i,j), "\n"],
   {i, 1, 2*nmodi}, {j, 1, 2*nmodi}];
Table[WriteString[strm, i, "  ", j, "  ", Part(k0gn,i,j), "\n"],
   {i, 1, 2*nmodi}, {j, 1, 2*nmodi}];
Table[WriteString[strm, i, "  ", Part(Z0,i), "\n"], {i, 1,
2*nmodi}];
Table[WriteString[strm, i, "  ", Part(DZ0,i), "\n"], {i, 1,
2*nmodi}];
Table[WriteString[strm, i, "  ", 0, "\n"], {i, 1, 2*nmodi}];
WriteString[strm, 0, "\n"];
Close[strm];
```

```
strm1 = OpenWrite["Rkdata"];
Write[strm1, tmax]; Write[strm1, timestep]; Write[strm1, accuracy];
Close[strm1];
```

ü Solving

```
LinkOpen["V.exe"];
```

ü Results reading

```
npoints = OpenRead["NPOINTS"];
sampl = Read[npoints, Real];
```

```
res = OpenRead["Results"];
NM1 = Read[res, Real];
matres = Table[Read[res, Real], {j, 1, sampl}, {i, 1, 4*nmodi + 2}];
```

```
matres = Transpose[matres];
matres = Delete[matres, 2];
```

ü Results decoding

```
time = Table[matres[[1,j]], {j, 1, sampl}];
```

```
solution1 = Table[Table[matres[[i,j]], {j, 1, sampl}], {i, 2,
4*nmodi + 1}];
solution2 = Table[Sum[tr[[i,j]]*solution1[[j]], {j, 1, 2*nmodi}],
    {i, 1, NM}];
solution3 = Table[Sum[tr[[i,j - 2*nmodi]]*solution1[[j]],
     {j, 2*nmodi + 1, 4*nmodi}], {i, 1, NM}];
solution4 = Join[solution2, solution3];
```

```
solution = Table[Table[{time[[j]], solution4[[i,j]]}, {j, 1,
sampl}],
    {i, 1, 2*NM}];
```

Solution building

ü Joining the solution with the boundary fixed nodes

```
zero = Table[{time[[i]], 0}, {i, 1, sampl}];
```

```
Table[If[listavincoli[[nn + 1 - i,dof + 1 - j]] == 1,
    solution = Insert[solution, zero, dof*i - dof + j]], {i, 1, nn},
   {j, 1, dof}];
Table[If[listavincoli[[nn + 1 - i,dof + 1 - j]] == 1,
    solution = Insert[solution, zero, dof*(i + nn) - dof + j]], {i,
1, nn},
   {j, 1, dof}];
```

```
base1 = base;
Table[If[listavincoli[[nn + 1 - i,dof + 1 - j]] == 1,
    base1[[m]] = Insert[base1[[m]], 0, dof*i - dof + j]], {i, 1,
nn},
   {j, 1, dof}, {m, 1, 2*nmodi}];
```

ü Separating position and velocity vectors

```
posizioni = Take[solution, dof*nn];
velocità = Take[solution, -dof*nn];
```

ü Matrices and vectors decomposition

```
abb = Table[posizioni[[dof*i - 1]], {i, 1, nn}];
vel = Table[velocità[[dof*i - 1]], {i, 1, nn}];
rotx = Table[posizioni[[dof*i - 3]], {i, 1, nn}];
drotx = Table[velocità[[dof*i - 3]], {i, 1, nn}];
roty = Table[posizioni[[dof*i - 2]], {i, 1, nn}];
droty = Table[velocità[[dof*i - 2]], {i, 1, nn}];
alf = Table[posizioni[[dof*i]], {i, 1, nn}];
dalf = Table[velocità[[dof*i]], {i, 1, nn}];
```

```
k0gpr = k0g;
k2gpr = k2g;
k1gpr = k1g;
zero = Table[0, {NM}];
zero1 = Table[0, {dof*nn}];
Table[If[listavincoli[[nn + 1 - i,dof + 1 - j]] == 1,
    k0gpr = Insert[k0gpr, zero, dof*i - dof + j]], {i, 1, nn}, {j,
1, dof}];
k0gpr = Transpose[k0gpr];
Table[If[listavincoli[[nn + 1 - i,dof + 1 - j]] == 1,
    k0gpr = Insert[k0gpr, zero1, dof*i - dof + j]], {i, 1, nn}, {j,
1, dof}];
k0gpr = Transpose[k0gpr];
Table[If[listavincoli[[nn + 1 - i,dof + 1 - j]] == 1,
    k2gpr = Insert[k2gpr, zero, dof*i - dof + j]], {i, 1, nn}, {j,
1, dof}];
k2gpr = Transpose[k2gpr];
Table[If[listavincoli[[nn + 1 - i,dof + 1 - j]] == 1,
    k2gpr = Insert[k2gpr, zero1, dof*i - dof + j]], {i, 1, nn}, {j,
1, dof}];
k2gpr = Transpose[k2gpr];
Table[If[listavincoli[[nn + 1 - i,dof + 1 - j]] == 1,
    k1gpr = Insert[k1gpr, zero, dof*i - dof + j]], {i, 1, nn}, {j,
1, dof}];
k1gpr = Transpose[k1gpr];
Table[If[listavincoli[[nn + 1 - i,dof + 1 - j]] == 1,
    k1gpr = Insert[k1gpr, zero1, dof*i - dof + j]], {i, 1, nn}, {j,
1, dof}];
k1gpr = Transpose[k1gpr];
```

```
k0gmec = k0gpr;
Table[k0gmec = Delete[k0gmec, dof*(nn - i + 1)], {i, 1, nn}];
k0gmec = Transpose[k0gmec];
Table[k0gmec = Delete[k0gmec, dof*(nn - i + 1)], {i, 1, nn}];
k0gmec = Transpose[k0gmec];
k2gmec = k2gpr;
Table[k2gmec = Delete[k2gmec, dof*(nn - i + 1)], {i, 1, nn}];
k2gmec = Transpose[k2gmec];
Table[k2gmec = Delete[k2gmec, dof*(nn - i + 1)], {i, 1, nn}];
k2gmec = Transpose[k2gmec];
k1gmec = k1gpr;
Table[k1gmec = Delete[k1gmec, dof*(nn - i + 1)], {i, 1, nn}];
k1gmec = Transpose[k1gmec];
Table[k1gmec = Delete[k1gmec, dof*(nn - i + 1)], {i, 1, nn}];
k1gmec = Transpose[k1gmec];
k0galfa = Table[k0gpr[[dof*i,dof*j]], {i, 1, nn}, {j, 1, nn}];
k2galfa = Table[k2gpr[[dof*i,dof*j]], {i, 1, nn}, {j, 1, nn}];
k1galfa = Table[k2gpr[[dof*i,dof*j]], {i, 1, nn}, {j, 1, nn}];
```

```
mecpos = posizioni;
Table[mecpos = Delete[mecpos, dof*(nn - i + 1)], {i, 1, nn}];
mecvel = velocità;
Table[mecvel = Delete[mecvel, dof*(nn - i + 1)], {i, 1, nn}];
```

```
alfp = Transpose[Table[Flatten[Transpose[Delete[Transpose[alf[[i]]],
1]]],
     {i, 1, nn}]];
dalfp =
Transpose[Table[Flatten[Transpose[Delete[Transpose[dalf[[i]]], 1]]],
     {i, 1, nn}]];
abbp = Transpose[Table[Flatten[Transpose[Delete[Transpose[abb[[i]]],
1]]],
     {i, 1, nn}]];
velp = Transpose[Table[Flatten[Transpose[Delete[Transpose[vel[[i]]],
1]]],
     {i, 1, nn}]];
mecposp =
Transpose[Table[Flatten[Transpose[Delete[Transpose[mecpos[[i]]],
        1]]], {i, 1, 3*nn}]];
mecvelp =
Transpose[Table[Flatten[Transpose[Delete[Transpose[mecvel[[i]]],
        1]]], {i, 1, 3*nn}]];
```

ü Energy computation

```
enpotel = -(1/2)*Table[alfp[[t]] . k0galfa . alfp[[t]], {t, 1,
sampl}];
encinel = -(1/2)*Table[dalfp[[t]] . k2galfa . dalfp[[t]], {t, 1,
sampl}];
enel = enpotel + encinel;
enpotmec = -(1/2)*Table[mecposp[[t]] . k0gmec . mecposp[[t]], {t, 1,
sampl}];
encinmec = -(1/2)*Table[mecvelp[[t]] . k2gmec . mecvelp[[t]], {t, 1,
sampl}];
enmec = enpotmec + encinmec;
```

```
enelp = Table[{time[[j]], enel[[j]]}, {j, 1, sampl}];
enmecp = Table[{time[[j]], enmec[[j]]}, {j, 1, sampl}];
encinelp = Table[{time[[j]], encinel[[j]]}, {j, 1, sampl}];
enpotelp = Table[{time[[j]], enpotel[[j]]}, {j, 1, sampl}];
enpotmecp = Table[{time[[j]], enpotmec[[j]]}, {j, 1, sampl}];
encinmecp = Table[{time[[j]], encinmec[[j]]}, {j, 1, sampl}];
```

ü Coupling coefficient

```
cmat = Chop[Inverse[Chop[k2gn]] . k1gn, 10^(-5)] .
    Inverse[Sqrt[Chop[Inverse[k2gn] . k0gn]]];
coupl = Table[Abs[cmat[[i,j + nmodi]]], {i, 1, nmodi}, {j, 1,
nmodi}];
coupl = coupl/Max[coupl];
```

ü Plots preparing

```
genpotel = ListPlot[enpotelp, PlotStyle -> Hue[0.1], PlotJoined ->
True,
    DisplayFunction -> Identity];
gencinel = ListPlot[encinelp, PlotStyle -> Hue[0.2], PlotJoined ->
True,
    DisplayFunction -> Identity];
genel = ListPlot[enelp, PlotStyle -> Hue[0.8], PlotJoined -> True,
    DisplayFunction -> Identity];
genpotmec = ListPlot[enpotmecp, PlotStyle -> Hue[0.4], PlotJoined ->
True,
    DisplayFunction -> Identity];
gencinmec = ListPlot[encinmecp, PlotStyle -> Hue[0.7], PlotJoined ->
True,
    DisplayFunction -> Identity];
genmec = ListPlot[enmecp, PlotStyle -> Hue[0.6], PlotJoined -> True,
    DisplayFunction -> Identity];
```

```
galf[i_] := ListPlot[alf[[i]], PlotStyle -> Hue[0.8], PlotJoined ->
True,
    DisplayFunction -> Identity];
gabb[i_] := ListPlot[abb[[i]], PlotStyle -> Hue[0.6], PlotJoined ->
True,
    DisplayFunction -> Identity];
```

```
fr = 1;
```

```
abbassamento = Table[{nodes[[i,2]], nodes[[i,3]], abbp[[t,i]]},
    {t, 1, sampl, fr}, {i, 1, nn}];
alfas = Table[{nodes[[i,2]], nodes[[i,3]], alfp[[t,i]]}, {t, 1,
sampl, fr},
    {i, 1, nn}];
gabbs = Table[FEMSurface[abbassamento[[t]], elements], {t, 1, sampl,
fr}];
galfa = Table[FEMSurface[alfas[[t]], elements], {t, 1, sampl, fr}];
```

```
mode = Plot[Sum[(1/(freq[[i]]^2 - s^2))^2, {i, 1, nmodi}],
    {s, 0.5, freq[[nmodi]] + 0.5}, PlotStyle -> Hue[0.8], PlotPoints
-> 40,
    DisplayFunction -> Identity];
modm = Plot[Sum[(1/(freq[[i]]^2 - s^2))^2, {i, nmodi + 1, 2*nmodi}],
    {s, 0.5, freq[[2*nmodi]] + 0.5}, PlotStyle -> Hue[0.6],
PlotPoints -> 40,
    DisplayFunction -> Identity];
```

```
abbm = Table[{nodes[[i,2]], nodes[[i,3]], base1[[nmodi + m,dof*i -
1]]},
    {m, 1, nmodi}, {i, 1, nn}];
alfm = Table[{nodes[[i,2]], nodes[[i,3]], base1[[m,dof*i]]}, {m, 1,
nmodi},
    {i, 1, nn}];
gmodm = Table[FEMSurface[abbm[[m]], elements], {m, 1, nmodi}];
gmode = Table[FEMSurface[alfm[[m]], elements], {m, 1, nmodi}];
modi := GraphicsArray[Table[{gmodm[[i]], gmode[[i]]}, {i, 1,
nmodi}]];
```

```
gm = Table[Show[gmodm[[i]], BoxRatios -> {1, 1, 0.2}, PlotRange ->
All,
     DisplayFunction -> Identity], {i, 1, nmodi}];
ge = Table[Show[gmode[[i]], BoxRatios -> {1, 1, 0.2}, PlotRange ->
All,
     DisplayFunction -> Identity], {i, 1, nmodi}];
```

```
puls[mo_] := freq[[nmodi + mo]]; pulse[me_] := freq[[me]];
tmaxm[mo_] := 2*Pi/puls[mo]; tmaxe[me_] := 2*Pi/pulse[me];
tsm[mo_] := tmaxm[mo]/20; tse[me_] := tmaxe[me]/20;
vibrm[mo_] := Table[{nodes[[i,2]], nodes[[i,3]],
     base1[[nmodi + mo,dof*i - 1]]*Cos[puls[mo]*t]}, {i, 1, nn}];
vibre[me_] := Table[{nodes[[i,2]], nodes[[i,3]], base1[[me,dof*i]]*
      Cos[pulse[me]*t]}, {i, 1, nn}];
gvibm[mo_] := FEMSurface[vibrm[mo], elements];
gvibe[me_] := FEMSurface[vibre[me], elements];
```

Results displaying

ü Frequency responce

```
Show[{modm, mode}, AxesLabel -> {"ω", "Frequency response"},
  PlotRange -> {0, 500}, Ticks -> {Automatic, None}, AspectRatio ->
0.3,
  DisplayFunction -> $DisplayFunction, ImageSize -> 200]
```

Eigenfrequencies

```
freq
```

ü Modal shapes

```
Show[GraphicsArray[{Table[gm[[i]], {i, 1, 3}], Table[gm[[i]], {i, 4,
6}],
    Table[gm[[i]], {i, 7, 9}]}]]
```

```
Show[GraphicsArray[{Table[ge[[i]], {i, 1, 3}], Table[ge[[i]], {i, 4,
6}],
    Table[ge[[i]], {i, 7, 9}]}]]
```

ü Modes animation

```
mo = 3;
Animate[gvibm[mo], {t, 0, tmaxm[mo], tsm[mo]},
   PlotRange -> {{0, 2}, {0, 2}, {-0.03, 0.03}}, BoxRatios -> {1, 1,
0.2},
   ImageSize -> 400];
```

```
me = 3;
Animate[gvibe[me], {t, 0, tmaxe[me], tse[me]},
   PlotRange -> {{0, 2}, {0, 2}, {-0.13, 0.13}}, BoxRatios -> {1, 1,
0.2},
   ImageSize -> 400];
```

ü Coupling coefficients rapresentation

```
ListDensityPlot[coupl, ColorFunction -> (GrayLevel[1 - #1] & ),
  ColorFunctionScaling -> False, Ticks -> None]
```

ü Energy vs time

```
Show[{genmec, genel}, PlotRange -> All, Ticks -> {Automatic, None},
  AxesLabel -> {Time, Energy}, DisplayFunction -> $DisplayFunction]
```

ü Displacements vs time

```
gmesh = MeshGraphics[nodes, elements];
nodenumbers := LabeledListPlot[Table[{nodes[[i,2]], nodes[[i,3]]},
     {i, 1, nn}], DisplayFunction -> Identity];
Show[{gmesh, nodenumbers}, AspectRatio -> Automatic, Axes ->
Automatic,
   PlotRange -> All];
```

```
Show[{gabb[200], galf[200]}, PlotRange -> All,
  DisplayFunction -> $DisplayFunction, AxesLabel -> {Time,
Displacements},
  ImageSize -> 200]
```

ü 3D animations

```
fr = 1;
ShowAnimation[Table[gabbs[[fr*i - fr + 1]], {i, 1, Quotient[200,
fr]}],
  Axes -> True, AxesLabel -> {x, y, z},
  PlotRange -> {{0, 2}, {0, 2}, {-0.06, 0.06}}, BoxRatios -> {1, 1,
0.2},
  ImageSize -> 400]
```

```
fr = 1;
ShowAnimation[Table[galfa[[fr*i - fr + 1]], {i, 1, Quotient[sampl,
fr]}],
  Axes -> True, AxesLabel -> {x, y, z},
  PlotRange -> {{0, 2}, {0, 2}, {-0.02, 0.02}}, BoxRatios -> {1, 1,
0.2},
  ImageSize -> 400]
```

Bibliography

[1] O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method,* McGraw - Hill, 1989

[2] E. Hinton and D. R. J. Owen, *Finite Element Programming,* Academic Press, 1977

[3] Xu Zhongnian, "A thick-thin triangular plate element ", Int. j. numer. methods eng., **33**, 963-973 (1992)

[4] N.W. Hagood and A. von Flotow, "Damping of structural vibrations with piezoelectric materials and passive electrical networks", J. of Sound and Vibration, **146** (2), 243-268 (1991)

[5] R..C. Batra, S. Vidoli and F. Vestroni, "Plane wave solutions and modal analysis in higher-order shear and normal deformable plate theories", Department of Engineering Science and Mechanics,M/C 0219 Virginia Polytechnic Institute and State University Blacksburg ,VA 24061, Dipartimento di Ingegneria Strutturale e Geotecnica Università di Roma "La Sapienza" 00184 Roma, Italia

[6] R.C. Batra and S.Vidoli, " A Higher Order Piezoelectric Plate Theory Derived from a Three-Dimensional Variational Principle", Department of Engineering Science and Mechanics, M/C. 0219 Virginia Polytechnic Institute and State University Blacksburg, VA 24061

[7] Domingos Alves Rade and Valder Steffen JR, "Optimisation of Dynamic Vibration Absorbers Over a Frequency Band", Mechanical Systems and Signal Processing, **14** (5), 679-690 (2000)

[8] E.Santini, "FEM synthesis of electromagnetic fields", Int. J. Numer. Model., **13**, 321-328 (2000)

[9] A. K. Soh, Z. F. Long, S. Cen, "A new nine DOF triangular element for analysis of thick and thin plates", Computational Mechanics, **24**, 408-417, (1999)., Springer -Verlag 1999

[10] Kai-Uwe Bletzinger, Manfred Bischoff and Ekkehard Ramm, "A unified approach for shear-locking-free triangular and rectangular shell finite elements", Computers and Structures, **75**, 321-334, (2000)

[11] Paolo Bisegna and Giovanni Caruso, "Mindlin-Type Finite Elements for Piezoelectric Sandwich Plate", J. of intelligent material systems and structures, **11**, 14-25, January 2000

[12] O. C. Zienkiewicz, R. L. Taylor and J. M. Too, "Reduced Integration Technique in General Analysis of Plates and Shells", Int. J.. for Numer. Methods in Engineering, **3**, 275-290, (1971)

[13] Robert L. Taylor and Ferdinando Auricchio, "Linked Interpolation for Reissner-Mindlin Plate Elements: Part II-A Simple Triangle", Int. J. for Numer. Methods in Engineering, **36**, 3057-3066, (1993)

[14] Panayiotis Papadopoulo and Robert L. Taylor, "A Triangular Element Based on Reissner-Mindlin Plate Theory", Int. J. for Numer. Methods in Engineering, **30**, 1029-1049, (1990)

[15] F. dell'Isola and S. Vidoli, "Continuum modelling of piezo-electro-mechanical trust beams: an application to vibration damping", Archive of applied mechanics - INGENIEUR ARCHIV, **68,** 1-19, (1998)

[16] F. dell'Isola and S. Vidoli, "Damping of bending waves in trust beams by electrical transmission line with PZT actuators", Archive of applied mechanics - INGENIEUR ARCHIV, **68,** 626-636, (1998)

[17] S. Vidoli and F. dell'Isola, "Modal coupling in one-dimensional electromechanical structured continua", ACTA MECHANICA, **141/1-2,** (2000)

[18] S. Vidoli and F. dell'Isola, "Vibrational control in plates by uniformly distributed actuators interconnected via electric networks", European J. of Mechanics A/Solids, **20**, 435-456, (2001)

[19] S. Alessandroni, F. dell'Isola and F. Frezza "Optimal piezo-electromechanical coupling to control plate vibrations", to appear in International Journal of Applied Electromagnetics and Mechanics

[20] F. dell'Isola, M. Porfiri and E. Henneke "", to appear in International Journal of Applied Electromagnetics and Mechanics

[21] S. Alessandroni, F. dell'Isola and M. Porfiri, "A revival of electric analogs for vibrating mechanical systems aimed to their efficient control by PZT actuators", submitted for publication

[22] A. Guran and D. J. Inman, *Wave motions, intelligent structures and non linear mechanics*, World Scientific, Singapore, 1995

[23] C. D. Near, "Piezoelectric actuators tecnology", proceedings of SPIE conference on smart materials and structures, pp. 24-29, San Diego, California, 1996

VITA

Domenico Vigilante was born in Rome the 04/14/1976. He took his high school degree in Rome at "Liceo scientifico Benedetto Croce" with the maximum grade of 60/60. In 1995 he began his graduate studies in electronic engineering at "Università la Sapienza di Roma" in Rome. He got his master degree in electronic engineering in 12/20/2001. In 2000 he began his graduate studies in mechanical engineering at Virginia Tech University.