Fluid Force Identification

Verifying and Validating a Method to Predict Wear of a Control Rod in a Nuclear Power Plant

MSc. Thesis

author : J.W.M. de Jong
supervisor EDF : C. Bodel
supervisor TUDelft : Prof. Dr. Ir. D.J. Rixen

March 8, 2010

Abstract

In this thesis a method to model and identify the force inducing a vibration on the control rods of a nuclear power plant is verified, but could not be validated. The aim of the method is to be able to predict wear in future nuclear power plant designs. This study is based on a study by Bodel [1], and reviews, adjusts and extents its contents.
The liquid coolant in a nuclear power plant, flowing passed the control rods induces them to vibrated, wearing them out at the positions of the guide plates. The method aims to model and identify the fluid forces of the liquid coolant, by looking at the dynamic behaviour of a model of the control rod. With this modelled fluid force, predictions can be made on the wear in future power plant designs. To describe the dynamic behaviour of the model of the rod properly, the mode-shapes of the dynamic system need to be mass normalised taking into account the added mass of the water surrounding the model. After the fluid force is modelled and identified it should be validated by applying it to a model which is closer to the real control rod. By comparing the results to the experimental results obtained on this model, the fluid forces can be validated.
The results are however, that the experimental mode-shapes were not well mass normalised by the measurement system and the dynamic system in water could therefore not be obtained properly. By using 'hand'-normalised mode-shapes the study is continued and the method could be verified. Only the modelling of the fluid force needs to be reconsidered, because it can be shown that the way the fluid force is modelled will not give the desired results. The validation of the fluid forces could not by accomplished, because the more realistic model of the control rod does not function properly. Furthermore, a fundamental problem with the method at hand emerges, when using the Polymax algorithm to identify the fluid force.

Acknowledgement

With completing this master thesis and with that, my studies, I am closing an important period of my live. Writing this in Amsterdam makes it even more special, because in this city I started my studies in Econometrics some eleven years ago. But the study could not keep my attention and after also trying a study in Economics in Groningen, I finally found my true interest in Mechanical Engineering at Delft University of Technology.
The final year, I was fortunate to do my thesis research at EDF near Paris. I found myself a nice, but small apartment and started the research of which this thesis is the result.
There are numerous people who helped me during my studies at Delft and at EDF. I can not express my gratitude to all of them here and I will surely forget many, so for that I apologize. But I would like to use this opportunity to thank a few.
First of all, I would like to thank Charles Bodel, who was a great help during my studies at EDF. Teaching me the ins and outs of his study, he jump started me in this thesis topic. Not only did he helped me a lot with the research itself, but also his many efforts organising the bureaucratic process of my stay at EDF, made my year in Paris the pleasant period it was. Next to that, I also would like to thank him for showing me some of the nightlife in Paris and of course for the crash course in French Wine.
Many thanks also go to Prof. Dr. Ir. Daniel J. Rixen. His extensive knowledge and his capability to explain difficult matter in a simple manner, have truly been contributing to this project. I really enjoyed the discussions with him, in which he shed light on those aspects of the study, which were troubling me.
Furthermore, I would like to thank Mathieu Corus for his effort helping me with the data from the experiments. And for explaining me the more theoretical parts of my research.
Next is Didier Bosselut, who I would like to thank for explaining the more general functionality of the nuclear power plant and of course for his cooking efforts, with which he introduced me to the French 'cuisine'.
Last but not least, I would like to thank my family, especially my mother, who kept her confidence in me over these many years..

Amsterdam,
March 2010

Joost de Jong

Contents

Nomenclature

List of Symbols

General Symbols

Symbol	Meaning	SI Units
x	x-direction in cartesian coordinate system	[m]
y	y-direction in cartesian coordinate system	[m]
z	z-direction in cartesian coordinate system	[m]
θ, ψ	angle	[rad]
ρ	density	[kg/m^3]
ε	strain	[−]
L	length of the tube	[m]
R	outer radius of the tube	[m]
r	inner radius of the tube	[m]
m	mass	[kg]
A	cross sectional surface area	[m^2]
E	Young's modulus	[N/m^2]
I	area moment of inertia	[m^4]
U	speed of the fluid flow	[m/s]
f	frequency	[Hz]
ω	rotational frequency	[rad/s]
k	spring constant	[N/m]
C, c	constant	[−]
j	complex variable complying with $j^2 = -1$	[−]

Symbols in Continuous Systems

Symbol	Meaning	SI Units
X	displacement field	[m]
ϕ_i	eigen mode-shape in displacement	[m]
ψ_i	eigen mode-shape in strain	[−]
κ_i	modal stiffness	[N/m]
μ_i	modal mass	[kg]
L_{eq}	equivalent length	[m]
ε_{eq}	equivalent strain-length	[1/m]

Symbols in Discreet Systems

Symbol	Meaning	SI Units
$\mathbf{x}$	displacement vector	[m]
$\boldsymbol{\varepsilon}$	strain vector	[m]
$\mathbf{q}$	modal force vector	[Nm]
$\mathbf{f}$	force vector	[N]
$\mathbf{u}$	fluid force vector	[N]
ϕ_{ij}	eigen mode-shape entry in displacement	[m]
$\boldsymbol{\phi}_{(i)}$	eigen mode-shape vector in displacement	[m]
$\boldsymbol{\Phi}$	eigen mode-shape matrix in displacement	[m]
ψ_{ij}	eigen mode-shape entry in strain	[−]
$\boldsymbol{\psi}_{(i)}$	eigen mode-shape vector in strain	[−]
$\boldsymbol{\Psi}$	eigen mode-shape matrix in strain	[−]
$\mathbf{M}$	mass matrix	[kg]
$\mathbf{K}$	stiffness matrix	[N/m]
a_{ij}, d_{ij}, g_{ij}	eigen mode participation factor in modal expansion	[−]
$\mathbf{a}_{(i)}, \mathbf{d}_{(i)}, \mathbf{g}_{(i)}$	eigen mode participation factor vector in modal expansion	[−]
$\mathbf{A}, \mathbf{D}, \mathbf{G}$	eigen mode participation factor matrix in modal expansion	[−]
$\mathbf{C}^{\mathrm{a}}$	is boolean matrix for accelerometer measurement devices	[−]
$\mathbf{C}^{\mathrm{s}}$	is boolean matrix for strain-gauge measurement devices	[−]
$\mathbf{B}$	is boolean matrix for fluid force load points	[−]
N	number of experimental DoF	[−]
M	number of experimentally obtained mode-shapes	[−]
n	number of numerical DoF	[−]
m	number of numerically obtained mode-shapes	[−]
P	number of DoF of the obtained point forces in the force vector	[−]

Subscripts

Symbol	Meaning
x, y, z	direction
w	in water
a	in air
i, j	index of a scalar in a vector or matrix
(i), (j)	index of a vector in a matrix
eq	equivalent

Superscripts

Symbol	Meaning
a	accelerometers
s	strain gauges
ex	experimental
et	'etendre' with translates to 'spread out' so expanded
num	numerical
Ph	Phacetie model
Ma	Magaly model

Diacritics

Symbol	Meaning	Operation
$\dot{a}$	a derived ones with respect to time t	$\frac{\partial}{\partial t}$
$\ddot{a}$	a derived twice with respect to time t	$\frac{\partial^2}{\partial t^2}$
a'	a derived ones with respect to space coordinate x	$\frac{\partial}{\partial x}$
a''''	a derived four times with respect to space coordinate x	$\frac{\partial^4}{\partial x^4}$
$\tilde{a}$	approximation of the variable a	
$\overline{a}$	mass normalised variable a	

Abbreviations

Abreviation	Meaning
EDF	Électricité du France.
PWR	Pressurized Water Reactor, type of nuclear reactor which is known for its stable operation.
EPR	European Pressurized Reactor, new type nuclear reactor of the PWR family.
DoF	Degrees of Freedom.
FEM	Finite Element Model.
PAK-system	Measurement system built by Müller BBM, in this study used to make measurements on the accelerometers.
LMS-system	Measurement system built by LMS international, in this study used to make measurements on the strain gauges.
MAC-table	table containing the MAC-numbers in a predefined order.
MAC-number	entry in the MAC-table, showing the correlation between to shapes.
SVD	Singular Value Decomposition.
PSD	Power Spectral Density is the Fourier transform of the auto-correlation function of a measurement point in the time domain.
CSD	Cross Spectral Density is the Fourier transform of the cross-correlation function between two measurement points in the time domain.

Notations

In general continues variables and scalars will be depicted with plane letters. Matrices and vectors will be depicted using bold characters, using capital letters for matrices, their lower case corresponding letters for vectors and their respective plane characters as scalars within.
As an example, the mode-shape matrices are defined as

$$\boldsymbol{\Phi}^{\text{ex}}_{N\times M} = \begin{bmatrix} s & s & & s \\ h & h & & h \\ a & a & & a \\ p & p & \cdots & p \\ e & e & & e \\ & & & \\ 1 & 2 & & M \end{bmatrix} \qquad \boldsymbol{\Psi}^{\text{num}}_{n\times m} = \begin{bmatrix} s & s & & s \\ h & h & & h \\ a & a & & a \\ p & p & \cdots & p \\ e & e & & e \\ & & & \\ 1 & 2 & & m \end{bmatrix} \qquad \boldsymbol{\Phi}^{\text{et}}_{n\times M} = \begin{bmatrix} s & s & & s \\ h & h & & h \\ a & a & & a \\ p & p & \cdots & p \\ e & e & & e \\ & & & \\ 1 & 2 & & M \end{bmatrix}$$

thus

$$\boldsymbol{\Phi}^{\text{exp}}_{N\times M} = \begin{bmatrix} \boldsymbol{\phi}_{(1)} & \boldsymbol{\phi}_{(2)} & \cdots & \boldsymbol{\phi}_{(M)} \end{bmatrix} = \begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1M} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{N1} & \phi_{N2} & \cdots & \phi_{NM} \end{bmatrix}$$

with N the number of measurement points and M the number of obtained experimental mode-shapes. n is the number of numerical DoF and m is the number of numerically obtained mode-shapes. An example for a participation matrix is

$$\mathbf{A}_{m\times M} = \begin{bmatrix} \mathbf{a}_{(1)} & \mathbf{a}_{(2)} & \cdots & \mathbf{a}_{(M)} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1M} \\ a_{12} & a_{22} & \cdots & a_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mM} \end{bmatrix}.$$

The Boolean matrices are defined as

$$\mathbf{C}^{s}_{N\times n} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{B}_{n\times P} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where P is the number of points with which the fluid force is modelled. N is the number of DoF in the measurements, thus the number of sensors and n is the number of DoF in the numerical model.

Chapter 1

Introduction

1.1 Research Context

Nuclear power has begun a revival now more and more people are becoming concerned with the effect of global warming. This global warming is due to the green house effect and is by some considered responsible for the climate change and exceptional whether conditions the last few decades. This green house effect is worsened by the carbon-dioxide which is emitted when burning fossil fuels. For this reason people are searching for different sources of energy like wind and hydro power. Also nuclear comes into the picture, especially since the safety of these kind of power plant is greatly improved.

1.1.1 Électricité du France

France has a long history of nuclear power. After the oil crisis in 1973 the French government decided to direct the demand for energy to nuclear power. In the following 15 years France has build and since operated, around 53 nuclear power plants. Électricité du France (EDF) is the company that build and operates the nuclear power plants, while Framatome (now part of Areva) designs them.

1.1.2 European Pressurized Reactor

The European Pressurized Reactor (EPR) is the new design by Framatome (Areva) and Siemens, with a pilot power plant in Finland, the Olkiluoto nuclear power plant. In figure 1.1 there is a picture of the unfinished plant. In France there is a second site, it is near Flamanville and should be operational in 2012. The EPR is the evolutionary descendant of the Framatome N4 and Siemens Power Generation Division KONVOI reactors. It can have an electrical power output of 1650 MW_e with a thermal power of 4500 MW_{th}, generated in its fuel rod assemblies.
The EPR will have four emergency cooling installations (300% redundancy) to cool down the reactor after shutdown.
The EPR is of the Pressurized Water Reactor (PWR) family. A Pressurised Water Reactor is a reactor with two coolant cycles. The coolant in the primary cycle is heated by the nuclear reaction in the fuel rod assembly. This primary coolant is pumped to a steam

Figure 1.1: The Olkiluoto nuclear power plant under construction

generator where the heat is transferred to the secondary cycle. The steam generated in the secondary cycle, drives the steam turbine. The turbine drives the generator, which in turn produces the electric power.
This primary circuit is the pressurized cycle. The pressure is in the order of 15.5 MPa at temperatures between 275 and 315 °C. At these temperatures and this pressure the coolant remains liquid. This coolant, water, has an important reaction controlling behaviour. To get the high speed neutrons to be part of the nuclear reaction, they need to be slowed down. The water reduces the speed of these neutrons flying in between the rods to the desired speed to interact in the reaction process. If the reaction intensifies, the temperature in the water coolant is increased, the increased pressure is released by expanding the volume of the first circuit in the expansion vessel (the pressuriser). In this expanded volume the water molecules are further apart and will not slow down the neutrons as much as before. Therefore, more neutrons will fly out of the fuel rod assembly and less neutrons will participate in the nuclear reaction, thus the reaction is slowed. This is the main reason why a PWR is such a stable nuclear power plant.

1.1.3 Fuel Rod Assembly with Control Rods

A fuel rod assembly consist of tubes containing the stacked uranium, or more commonly, uranium-oxide fuel cells, between which, the nuclear fission reaction takes place. These tubes are bundled in square fuel rod assemblies of 17 by 17 tubes. In specific places, instead of fuel rods, there are control rods. There are 25 control rods per fuel rod assembly. These control rods are also tubes, but these are filled with silver-indium-cadmium alloys or boron, that readily capture neutrons. Figure 1.2 shows a fuel rod assembly and the spider with the control rods next to it. In operation these control rods are inserted into the top of the fuel rod assembly, ready to put down into the control rod guiding tubes, to damp the reaction.

So, by lowering or raising the control rods in or out of the fuels rod assembly, the nuclear fission reaction is controlled. When the control rods are lowered into the fuel rod assembly, the nuclear reaction is slowed due to the increasing amount of neutrons which are captured by the control rods. When the control rods are raised, less neurons are captured and the nuclear reaction, and with that the heat generated, increases.
The reaction can in this way, be controlled with the control rods, in an active matter

- to start up the reactor. To gradually increase the nuclear reaction speed.

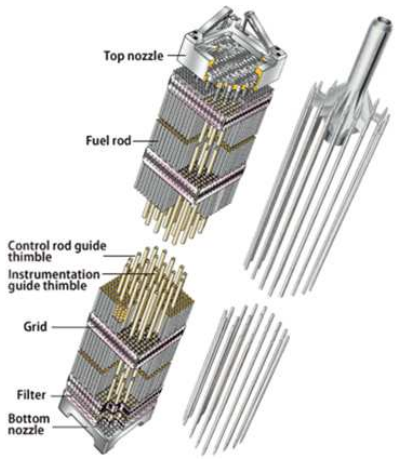


Figure 1.2: The fuel rod assembly and the control rods attached to the spider

- to shut down the reactor. By dropping down the control rod, the reaction can be stopped quite suddenly.
- to accommodate short term transients such as changes to load on the turbine.

More minor changes can also be controlled by the control rods

- to compensate for nuclear poison depletion; the phenomenon that some of the materials absorbing the neutrons deplete over time,
- to compensate for nuclear fuel depletion; when the nuclear fuel becomes less active, the nuclear reaction becomes less active and the control rods can be raised to compensate,

but these effects are more usually accommodated by altering the primary coolant boric acid concentration. In operating position the control rods are almost totally sticking out of the fuel rod assembly. Outside the fuel rod assembly the control rods are supported by guide plates.

1.1.4 The Study

The part of the control rod sticking out of the fuel rod assembly can start to vibrate. This vibration is induced by the fluid force of the coolant flowing past the control rod. Due to this vibration the control rods wears out mainly at the position of the guide plates. The aim of this study is to obtain a method to predict this wear for future designs in nuclear reactors.
In 2008 Bodel [1] finished a study in which the fluid forces were modelled. If this fluid force could be applied on a numerical model of the real future design, the wear can be predicted. For his study the 'Phacetie' model, a physical model similar to the real structure was build and submitted to a similar flow as the real control rod. Instead of directly modelling the

fluid force looking at the properties of the fluid flow, the attention is shifted to the control rod. By analysing the control rod and measuring its displacements, a fluid force can be calculated. Instead of a realistic fluid force, which would have a distributed load, the fluid force was modelled as two point-concentrated loads. By now, in this study, applying the modelled fluid force on a numerical model of the real structure and comparing the result to the test results obtained on the real control rods in the Olkiluoto nuclear power plant, this method can be validated. If this method gives similar results, than this method can be used to predict wear in future designs.

1.2 Thesis Assignment

The assignment of this study is to verify and validate the method initiated by Bodel [1] in 2008. Therefore, this study follows a similar path as the study done by Bodel and will broaden its basis, extend were needed and adjust if necessary. The conclusion by Bodel that the speed of the fluid has no significant influence on the measured modal parameters of the rod, is stated here as an assumption.
Figure 1.3 shows a scheme in which the study by Bodel is shown. As depicted in the figure, some extensions and new concepts will be tested versus the existing ones in the study by Bodel:

- A 2D shell-element model is constructed and tested against the original 3D volume-element model, to see which of the two models preforms better, considering computational speed and as a expansion space.

- Comparing the use of a static-shape space instead of a eigen mode-shapes space to expand the experimentally obtained mode-shapes data.

- All the analyses in this study are done in the two perpendicular directions to the tube simultaneously. The rod can vibrate in two directions in a similar manner. The measurements in water were done in these two directions simultaneously. For the study done by Bodel [1], these measurements where split into the different directions. In this study the expansion space will be made in two directions, so the measurements in water can be expanded without splitting them.

- A method of calculating the added mass due to the water surrounding the rod was developed by Bodel [1]. With this method the eigen mode-shapes of the experiments in water can be mass normalised. By integrating the calculation of the added mass and the normalisation of the mode-shapes, also mode-shapes in air can be use for force identification. If this new method gives similar good results, it would simplify the method greatly and make the measurements a lot easier.

Next to these points, elements of the study will be verified using other methods or new insights. After everything is verified, the method can be validated using the obtained fluid forces on a numerical model of the Magaly model, an experimental model of Areva, which is closer to the real control rod. So, in this thesis, decisions will be made considering these points. Next to that, the general method will be verified and then validated.

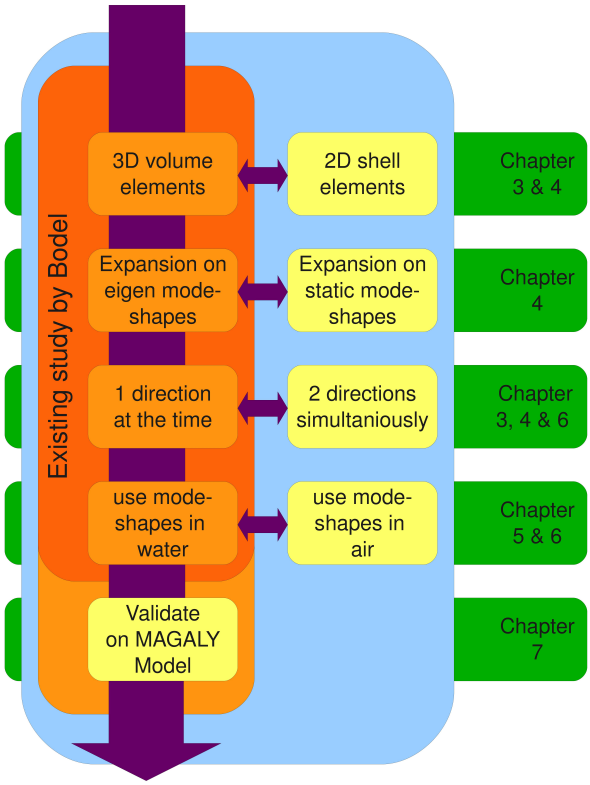


Figure 1.3: Overview of subjects with respect to the existing study by Bodel [1]

1.3 Set-up of the Study

A system in water can be modelled in general from the model in vacuum with a force due to the water surrounding the tube as

$$\mathbf{M}_0\ddot{\mathbf{x}} + \mathbf{C}_0\dot{\mathbf{x}} + \mathbf{K}_0\mathbf{x} = \mathbf{f}_\mathrm{w}\left(\ddot{\mathbf{x}}, \dot{\mathbf{x}}, \mathbf{x}, \mathbf{u}\right) \tag{1.1}$$

where, in vacuum, $\mathbf{M}_0$ is the mass matrix, $\mathbf{C}_0$ the damping matrix and $\mathbf{K}_0$ is the stiffness matrix. The force, $\mathbf{f}_\mathrm{w}\left(\ddot{\mathbf{x}}, \dot{\mathbf{x}}, \mathbf{x}, \mathbf{u}\right)$, models the water surrounding the tube and is dependent on the acceleration $\ddot{\mathbf{x}}$, speed $\dot{\mathbf{x}}$, and displacement $\mathbf{x}$, of the tube, and on the speed $\mathbf{u}$, of the fluid.
Coupling between the modes of the tube and in the water can be present in the real fuel assembly. A very simple calculation shows that for two rods, which are between three centimetres and fifteen centimetres apart, as in one fuel rod assembly, the first fluid mode will be around 100000 Hz to 5000 Hz respectively. From [4] using

$$f = c\frac{n}{2L}[\mathrm{Hz}], \tag{1.2}$$

where c is the speed of sound in the fluid, n the mode-shape number and L is the distance between the two rods. The frequency is basically the number of times the sound wave travels up and down to the other rod. For these high frequencies there will be no coupling between the acoustic modes in the water and the vibrational modes of the rods, which are in a range of 50 Hz to 200 Hz. Acoustic modes with lower frequencies, due to objects further away, which could have coupling with the structural modes are also assumed to be of minor influence and are left out of the modelling. This, because the vibration of the rod is radiated out in all directions and therefore fast decreasing in energy, for objects further away as depicted in figure 1.4.

For non or small fluid flow speeds $\mathbf{u}$, the fluid force is merely dependant on acceleration

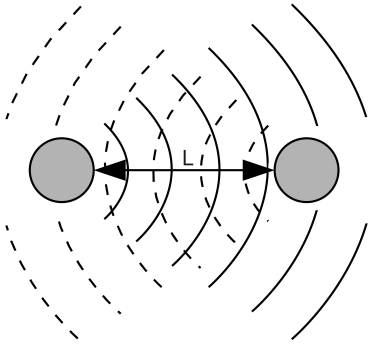


Figure 1.4: Schematic picture of two rods vibrating above a fuel cell assembly

and speed of the tube [8]. In linearised form

$$\mathbf{f}_{\mathrm{w}}\left(\mathbf{u} \simeq 0\right) \simeq \mathbf{M}_{\mathrm{added}}\ddot{\mathbf{x}} + \mathbf{C}_{\mathrm{added}}\dot{\mathbf{x}} \tag{1.3}$$

In this study, the influence of damping is not modelled and taken out of the equation. The influence of the acceleration of the tube on the fluid force is linearised and can therefore be added as an added mass to the mass matrix

$$\mathbf{M}_{\mathrm{added}} \simeq \frac{\partial}{\partial \ddot{\mathbf{x}}}\mathbf{f}_{\mathrm{w}} \tag{1.4}$$

$$\left(\mathbf{M}_0 + \mathbf{M}_{\mathrm{added}}\right)\ddot{\mathbf{x}} + \mathbf{K}_0\mathbf{x} = \tilde{\mathbf{f}}_{\mathrm{w}}\left(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{u}\right) \tag{1.5}$$

This theoretical system can be obtained in several manners. Figure 1.5 shows a scheme of all different possibilities to obtain a representation for the system in water. In general, the sequence is from top to bottom. On the left hand side the more experimentally orientated path is shown and on the right hand side a path that emphasises more on the numerical models. This study follows the more experimentally orientated path, which is depicted as the red line through the field of different possibilities.

For the chosen path, the mode-shapes of the system, $\boldsymbol{\phi}_{(i)}$, are to be obtained, which then need to be normalised taking into account the added mass. With these normalised mode-shapes the modal participations of the fluid forces can then be modelled and identified.

To be able to compare the experimental results obtained by Areva on the Magaly model,

with the results of the numerical Magaly model, this numerical model must be excited with the modelled fluid forces. These forces are identified on the Phacetie model, using the measured strains, obtained by the strain gauges in the tube of the model, while the model vibrates in the fluid flow. These strain measurements can only be used to calculate the fluid force if the mode-shapes which are used to model the system in water are well mass normalised. The mode-shapes obtained from the experiments in water cannot be mass normalised because the exciting force cannot be measured, this is the fluid force which is to be identified. In air however, the mode-shapes can be obtained using hammer excitation and thus the input force can be measured, making it possible to mass normalise these mode-shapes. By calculating the added mass due to the water surrounding the tube and using the mass normalised mode-shapes in air as a reference, the mode-shapes in water can be mass normalised. The study starts therefore by obtaining both the mode-shapes in water as well as in air and getting the ones in air to be mass normalised.
The thesis presented here is divided into the same parts as shown in figure 1.5.

1. Part I: Obtaining the Mode-shapes (red).
2. Part II: Mass Normalising the Mode-shapes (blue).
3. Part III: Validating the model (green).

Each part starts with an explanation on the different possibilities and the path chosen.
The first part further consists of chapter on the experiments, chapter 2. Chapter 3 is about the numerical models and the expansion of the experimentally obtained mode-shapes is described in chapter 4.
The second part starts again with some explanation. It shows the different approaches to obtaining the added mass due to the water surrounding the rod. This part consists of chapter 5 elaborating on the method used in the study by Bodel [1] to obtain the added mass and the normalisation of the mode-shapes in water thereafter. In this chapter also a new method is shown, integrating the calculation of the added mass and the normalising, making it possible to mass normalised the mode-shapes in air in such a way that they can be used as a basis for the system in water.
The third part shows in its introduction the field of possibilities to identify the fluid force. Chapter 6 then elaborates on the way chosen in this research, namely to invert the FRF-matrix. This part ends with the validation of the method by applying the identified forces on the Magaly model in chapter 7.
Finally the conclusion and recommendations are given in chapter 8.
The method in this study is partially based on theory shown in the appendix A. The other appendices have more of a supporting character. In appendix B the tests used throughout the thesis are explained. In appendix C some curiosities considering the numerical modelling are shown. And finally, appendix D, shows some information about software used, as well as the programs which were built to process the measurement data and obtain the results.

Remark: *This remark environment is use to address those who read this thesis for more practical purposes and it will be used mainly regarding Code_Aster.*

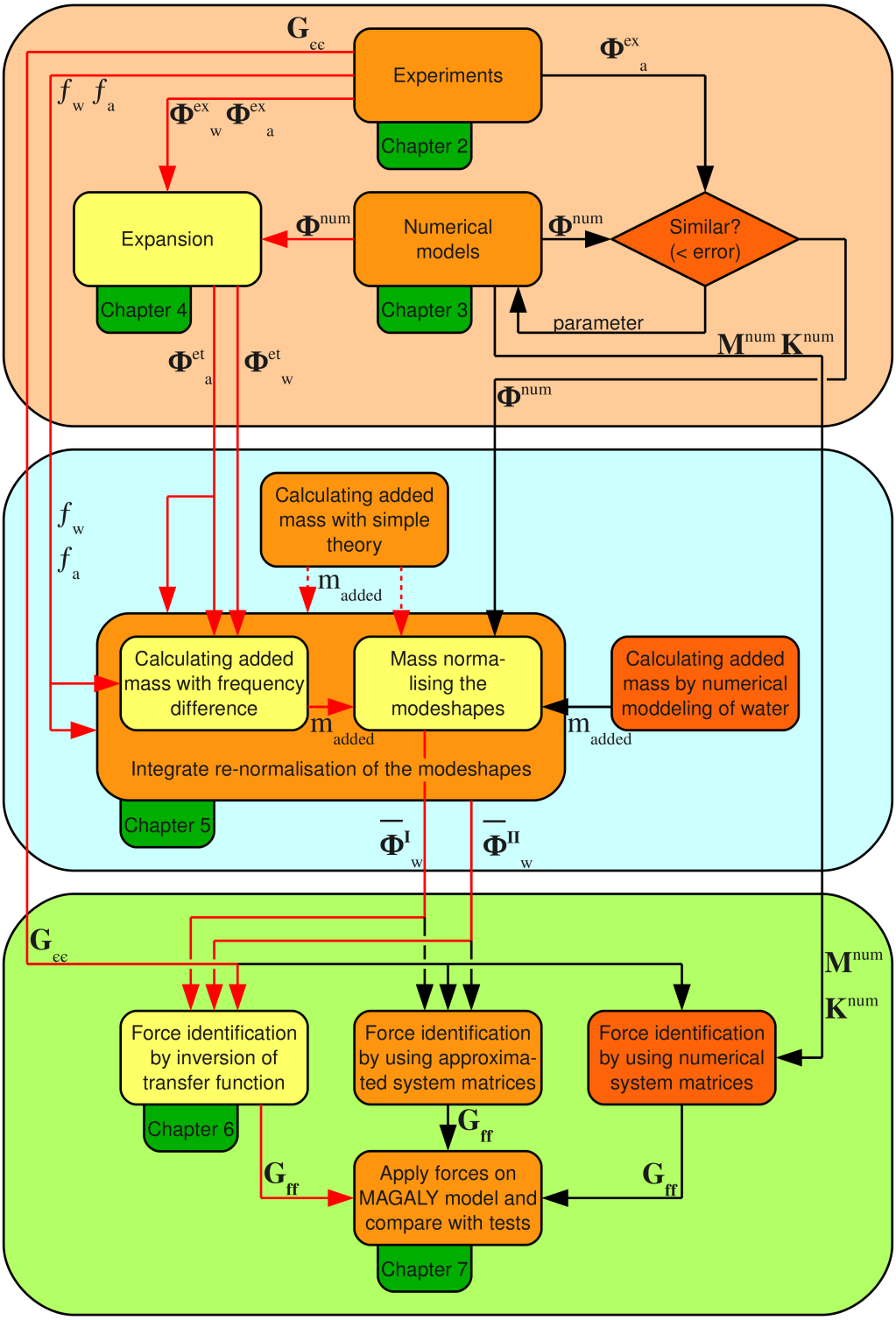


Figure 1.5: Scheme of the field of possibilities for a study and the study chosen as the red path. The quantities and their units, transferred by the arrows, are: f frequency, $\mathbf{G}$ spectral density, $\mathbf{M}$ mass matrix , $\mathbf{K}$ stiffness matrix and $\mathbf{\Phi}$ mode-shape matrix.

Part I

Obtaining the Mode-shapes

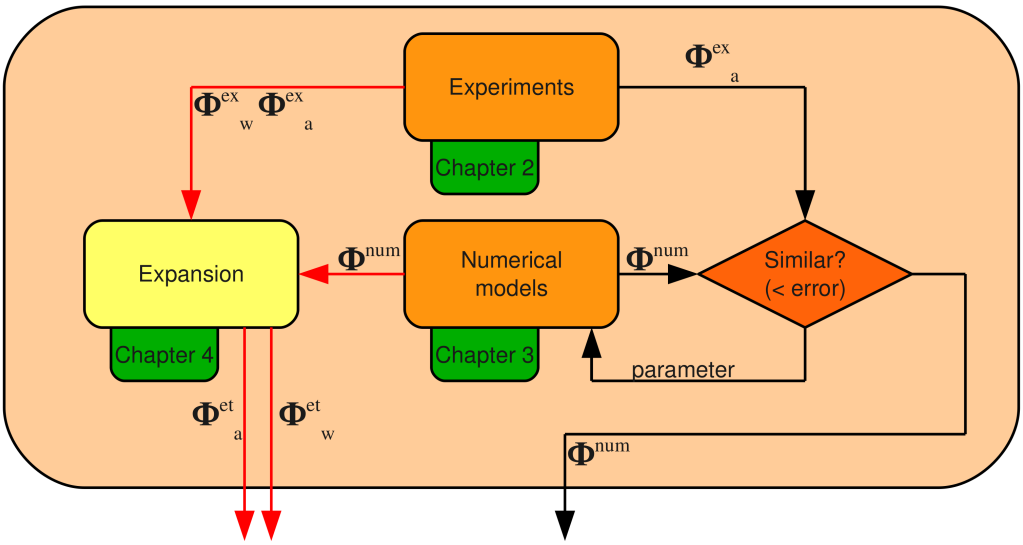


Figure 1.6: Scheme of the field of possibilities for the first part of the study. In red, the chosen path.

This part explains the first part of figure 1.5, which is again depicted in figure 1.6. To obtain well discretized eigen mode-shapes of the physical model 'Phacetie', two paths can be chosen in general.
First, model updating on the right hands side of the figure. This method aims to improve the numerical model by updating specifically chosen parameters like material properties or external spring stiffness. When the model finally complies with the criteria, the numerical eigen mode-shapes are used.
Second, the numerical models mode-shapes, which can be eigen mode-shapes or static-shapes, are used as a well discretized numerical space to expand the experimentally obtained eigen mode-shapes.
In this study the latter one is chosen because this does not require a well defined numerical model and is thus faster to apply. Next to that, there are sufficiently many sensors to make a good expansion. In this part the different expansion spaces (eigen and static) will be tested as well as the final well discretized eigen mode-shapes. Along the way two numerical models will be tested, the 3D volume-element model versus the 2D shell-element model. The criteria for testing are two fold, the MAC number between the different eigen mode-shapes and a test is being done to see whether the mode-shapes in air are well mass normalised.

Chapter 2

The Experiments

2.1 Introduction

The experiments are done to obtain the modal parameters, in this case, of the Phacetie model, shown in figure 2.2. The modal parameters need to be obtained for both the Phacetie model in water as well as in air. This because the eigen mode-shapes in air can be mass normalised and will be used as a reference when normalising the mode-shapes in water.
There are two measurement device-systems; (1) the strain gauges, to primarily obtain the modal parameters in water and (2) the accelerometers, which can only be used in air.
The measurements in air were done twice, the first time by Bodel [1], but tests during this study showed that the obtained eigen mode-shapes were not normalised by the algorithms of the measurement system. And could also not be renormalised by either one of the measurement systems or by the MATLAB SDTools developed by Balmès [17]. So in July 2009, new measurements in air were done (using the LMS system instead of the PAK-system). However, these new measurements came out with a lot lower natural frequencies, probably because of the use of different springs. These different springs had to be used because the spring used during the original measurement, were in use in the submerged set-up. For this reason the second measurements are classified as useless and are not further used.
Finally, as a compromise, the measurements done by Bodel [1] were used, but 'hand' normalised using a Python program. This 'hand'-normalisation can be done in two ways: (1) By changing the normalisation factor in the data-file, and (2), by changing the actual measurement data using a Python program that basically rewrites the datafile.
The measurements in air are done using hammer excitation and are therefore in just one direction perpendicular to the tube. In contrast to the study done by Bodel, in this study the analysis will be done in to directions simultaneously. The measurements in air will therefore be adjusted to form mode-shapes pairs, covering the two directions simultaneously.
This chapter is written on the last iteration step in the process to get the best measurement data. In this chapter the 'hand'-normalised mode-shapes will be tested for orthogonality and they are tested to see if they are in the order of being well mass normalised.

The test for orthogonality is done by the MAC-criterion as explained in appendix B.1. To see if the mode-shapes are well mass normalised a method is developed in appendix B.2. To get some insight in how the measurement systems obtains the mass normalised mode-shapes, appendix A.2 can be read.
In this chapter, first the Phacetie model is described in section 2.2. During the experiments to different measurement systems are used, explained in section 2.3. As said, there are two measurements done; one in air and one in water. This is described in section 2.4 for the measurement in air, and in section 2.5 for the measurements in water. The final section, section 2.6, shows the comments, conclusions and recommendations for this chapter.

2.2 The Phacetie Experimental Model

2.2.1 Set-up

The Phacetie model was built to represent the real control rod and so be able to obtain eigen mode-shapes similar to the mode-shapes of the real structure. The Phacetie model models the bottom half of the 4 m long control rods which stick out of the fuel assembly. The model consists of a steal tube hung by springs as shown in figure 2.1. The tube is 2.17 m long, has an inner diameter of 4 mm and an outer diameter of 5 mm. The Young's modulus is $E = 210\ \mathrm{N\,m^{-2}}$ and the density is $\rho = 7800\ \mathrm{kg\,m^{-3}}$. The guide plates in the nuclear power plant are modelled as aluminium blocks. The tube is connected to certain specific blocks by springs, to model the non-linear interaction between the plates and the control rod. With these spring the vibration of the tube can be kept in a linear domain. The tube is put through the springs and attached with the bolt visualised in the photo of figure 2.1. Only the positions of aluminium blocks 1, 3, 5, 7 and 8 (see figure 2.2) are having a spring in the Phacetie model. These blocks are selected while keeping

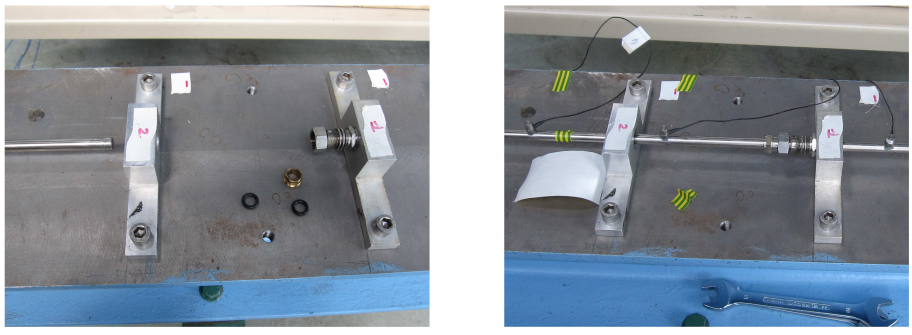


Figure 2.1: The spring connecting the tube to the aluminium blocks, (a) before assembly, (b) after assembly and with accelerometers

in mind two things; (1) the tube must not be totally restricted to vibrate, so only few blocks have a spring connecting the tube and, (2) the rod must have more freedom at the

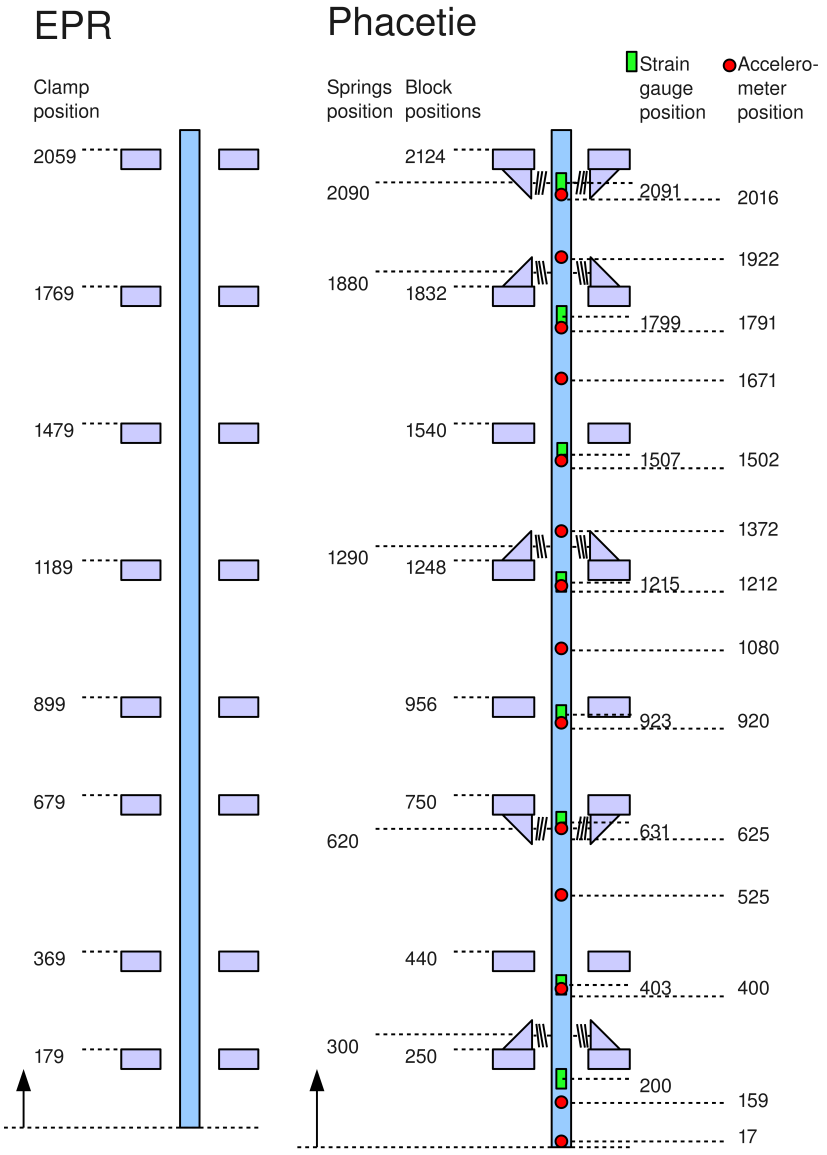


Figure 2.2: Schematic overview of the EPR and the experimental Phacetie model. The positions of the guide plates, the aluminium blocks, the springs and the different measurement sensors are given in mm.

bottom, when hung in water. This, to imitate that the part of the real control rod which is modelled, is the bottom half. And that the bottom end of the real control rod is a free end. During measurements, the aim is to avoid direct interaction between the remaining aluminium blocks and the tube. The final spring configuration is schematically depicted

in figure 2.2

2.2.2 Measurement Devices

There are two different measurement devices, accelerometers and strain gauges.

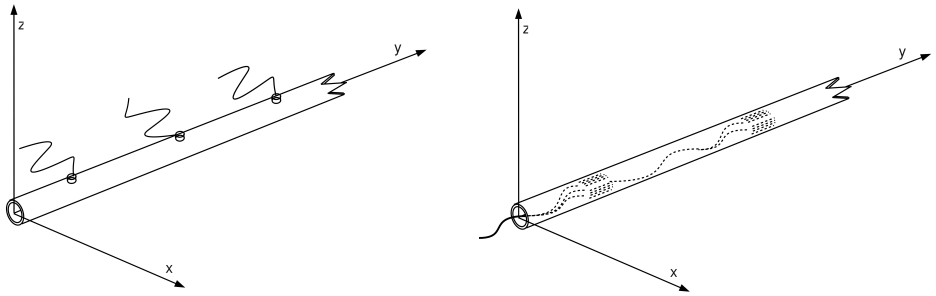


Figure 2.3: (a) Part of the tube showing the accelerometer sensors, in this configuration capable of measuring in one direction, (b) Part of the tube showing the strain gauge sensors on the inside. In this configuration capable of measuring in two directions.

Accelerometers

Accelerometers are used to do the measurements in air, they are not water resistant, and therefore not usable for the measurements in water. They would also influence the fluid flow, because they are positioned on the outside of the tube. The accelerometers are of the type B&K 4375, have a low mass ($2.4 \cdot 10^{-3}$ kg) compared to the mass of the tube (0.478 kg) and thus do not influence the frequencies too much (this was verified by adding these sensor masses to the tube in the numerical model). There are 14 accelerometers 'glued' to the tube with bee-wax. They are all in a line and so measure in just one direction as shown in figure 2.3(a). The devices measure acceleration and are placed on specific intervals along the tube of which a schematic overview is given in figure 2.2.

Strain Gauges

In water strain gauges are used instead of accelerometers. The strain gauges are placed on the inside of the tube, to avoid them getting wet. They are constructed by VISHAY with an element code 19227, and from each one there is a wire connecting the sensor to the measurement system. All these wires together come out at the top end of the tube as a thick bundle. There are 8 pairs of strain gauges as depicted in 2.3(b). The pairs are on a 90° angle and can therefore measure in two directions simultaneously. The strain gauges are glued on the inside of the tube at specific intervals which are shown in the scheme of figure 2.2. These strain gauges operate in air similar well as in water, so in air both measurement devices can be used.

2.3 Measurement Systems

The output of the sensors is connected to a measurement system. There were two measurement systems used; the PAK- and the LMS-system. The measurements done on the strain gauges were done using the LMS-system. The measurements done in air by Bodel [1] were done using the PAK-system. The second attempt to measure with the accelerometers was done with the LMS-system, but, as stated before, these came out wrong and are not used. Thus, all measurements done with the accelerometers were done with the PAK-system and all measurements done with the strain gauges were done with the LMS-system.

2.3.1 PAK System

The PAK system was developed by MÜLLER BBM in Germany. This system is used in the first attempt to obtain the measurements using the accelerometers during Bodels [1] study. It is the MKII-system, using software 'PAK 5.3' and for the modal identification, ME-scope software was used. The measurement system gives the mode-shape data, instead of what is maybe expected using accelerometers, in displacement in meters.

2.3.2 LMS System

The second attempt to obtain the mass normalised mode-shapes of the experimental model in air was done in July 2009, using the LMS system. This system is also the system used to obtain the measurements in water, using the strain gauges. This system is developed by LMS International, in Leuven, Belgium. The software used is the LMS Testlab 8B package using the algorithm Polymax to obtain the mode-shapes from the transfer function.

2.4 Measurements in Air

As explained in the introduction of this chapter, the measurements in air will be checked using the MAC criterion and the mass normalisation will be checked using a method developed in appendix B.2. The measurements in air are done using the accelerometers as measurement device and a roving hammer for excitation. The hammer is to be struck in the same direction as the one in which the accelerometers are able to measure. With this method the input force can be measured, with which the mass normalised mode-shapes should be obtainable as explained in appendix A.2. The first measurements, gave mode-shapes which could not be mass normalised be either the PAK-system nor by the MATLAB SDTools developed by Balmès [17]. However, these first measurements, the ones for Bodels [1] study, will be used during further studies, because the second measurements in July 2009 were not good at all. A measure to be able to continue the study with the non-normalised mode-shapes is to 'hand'-normalise them. This is done by changing the experimental data to comply with the mass normalisation check, explained in appendix B.2. The Python 'hand' mass normalisation program is in appendix D. This can of course be classified as 'cheating' and is a major flaw in the study.

2.4.1 MAC-Number in Air

The MAC-number and the table, are elaborated on in appendix B.1. The measurements in air are done in only one direction. Because the expansion will be done on a two dimensional (x and z) numerical mode-shape space, the measurements are adjusted by coping each mode-shape in the second direction. So, if the first mode-shape is in the x-direction, than the second mode-shape is the same shape, but in the z-direction. With this adjustment the mode-shapes will be easily expandable in the two-directional mode-shape space later on.
Only four mode-shapes were initially identified, so the auto MAC becomes an eight times eight table, in which the diagonal entries are the same in pairs. So, the diagonal entry for shape one and two are the same, just like three and four, etcetera. In table 2.1 the auto-MAC of measurements with PAK on accelerometers is depicted.

Table 2.1: Auto MAC of experimental mode-shapes in air using the PAK-system in displacement

vs.		PAK Experimental mode-shapes							
	nr. [freq.]	1 [70.26]	2 [70.26]	3 [111.89]	4 [111.89]	5 [154.69]	6 [154.69]	7 [202.61]	8 [202.61]
PAK Experimental mode-shapes	1 [70.26]	1.000	0.000	0.007	0.000	0.011	0.000	0.004	0.000
	2 [70.26]	0.000	1.000	0.000	0.007	0.000	0.011	0.000	0.004
	3 [111.89]	0.007	0.000	1.000	0.000	0.021	0.000	0.003	0.000
	4 [111.89]	0.000	0.007	0.000	1.000	0.000	0.021	0.000	0.003
	5 [154.69]	0.011	0.000	0.021	0.000	1.000	0.000	0.031	0.000
	6 [154.69]	0.000	0.011	0.000	0.021	0.000	1.000	0.000	0.031
	7 [202.61]	0.004	0.000	0.003	0.000	0.031	0.000	1.000	0.000
	8 [202.61]	0.000	0.004	0.000	0.003	0.000	0.031	0.000	1.000

This MAC is calculated without a mass matrix as weighting matrix At the same time the strain gauges measured the mode-shapes in strain, using the LMS-system. These mode-shapes should also be orthogonal as shown in appendix A.1. The MAC of measurements with LMS on strain gauges is shown in table 2.2

2.4.2 Mass Normalisation in Air

The mode-shapes in air need to be mass normalised to be a good reference in the later calculation of the added mass. Mass normalisation is explained in appendix A.1 and the way to check whether the measured mode-shapes are mass normalised, or at least in the order of mass normalisation, is explained in appendix B.2. The results of the mode-shapes normalisation test of the original mode-shapes are in the second column of table 2.3. It is clear from this table that the original mode-shapes are not even in the order of being well mass normalised. Still, only after the mode-shapes were expanded a better, well discretized calculation could be made to determine whether the mode-shapes are mass normalised well enough. After this was done the files containing the experimental mode-shape data were rewritten. The new, 'hand' normalised experimental data gives the result

Table 2.2: Auto MAC of experimental mode-shapes in air using the LMS-system in strain

vs.		LMS Experimental mode-shapes in air							
	nr. [freq.]	1 [70.19]	2 [70.19]	3 [111.89]	4 [111.89]	5 [154.70]	6 [154.70]	7 [202.60]	8 [202.60]
LMS Experimental mode-shapes in air	1 [70.19]	1.000	0.000	0.001	0.000	0.000	0.000	0.046	0.000
	2 [70.19]	0.000	1.000	0.000	0.001	0.000	0.000	0.000	0.046
	3 [111.89]	0.001	0.000	1.000	0.000	0.006	0.000	0.080	0.000
	4 [111.89]	0.000	0.001	0.000	1.000	0.000	0.006	0.000	0.080
	5 [154.70]	0.000	0.000	0.006	0.000	1.000	0.000	0.000	0.000
	6 [154.70]	0.000	0.000	0.000	0.006	0.000	1.000	0.000	0.000
	7 [202.60]	0.046	0.000	0.080	0.000	0.000	0.000	1.000	0.000
	8 [202.60]	0.000	0.046	0.000	0.080	0.000	0.000	0.000	1.000

in the third column. Keep in mind that these approximations are calculated from just the few data points from the experimental data and not from the well discretized data after the expansion. Therefore these numbers just give an indication on whether or not the mode-shapes are well mass normalised.

Table 2.3: Calculated equivalent length to show whether the obtained experimental eigen mode-shapes are mass normalised. The value of the equivalent lengths should be in the order of $L_{eq} = \int_L \phi^2 \, \mathrm{d}y = \frac{1}{\rho A} \approx 4.53$. In the first column the approximated equivalent length of original mode-shape $\tilde{L}_{eq}$ and in the second column the approximated equivalent length of 'hand'-normalised mode-shape $\tilde{\tilde{L}}_{eq}$.

		$\tilde{L}_{eq}$	$\tilde{\tilde{L}}_{eq}$
Experimental Mode-shape	1	$9.87{\cdot}10^{-9}$	4.88
	2	$1.40{\cdot}10^{-9}$	4.97
	3	$4.29{\cdot}10^{-9}$	5.89
	4	$6.24{\cdot}10^{-10}$	5.88

2.5 Measurements in Water

In water the tube is excited by the fluid force. Because this input signal can not be measured, the obtained mode-shapes from LMS can not be mass normalised. To obtain the mode-shapes in water, LMS uses the Polymax algorithm. Polymax will use an uncorrelated random excitation in both space and time if the real excitation signal is not measured at all. So, to obtain the mode-shapes in water, the fluid forces is 'modelled' to be randomly distributed in space and time, while the real fluid force has a correlation due to the vortices shedding along the tube.
Due to this random excitation the mode-shapes obtained from the measurements in water

cannot be mass normalised. So, only the MAC table is created to test the orthogonality of the obtained mode-shapes. This is now much more important, because if there would be any peaks in the Fourier transform of the true fluid force, they would transform into artificial, non system based mode-shapes, which should not be orthogonal to the true system based mode-shapes. By applying this orthogonality criterion, the obtained mode-shape space is checked for non system based mode-shapes.

2.5.1 MAC-Number in Water

Using the strain gauges the mode-shapes are obtained with the tube having a two-directional space to move in. One would expect to find eight slightly in frequency shifted mode-shapes (there is only added mass due to the water surrounding the rod and no added damping as explained in chapter 5), but only six are found. The first two correspond to the first two mode-shapes in air with frequency 70.2 Hz. The third mode-shape in water to the third and fourth in air, number four and five in water to five and six in air. And the last one in water, number six to the seventh and eighth in air. The MAC-table is shown in table 2.4. After the expansion is done in chapter 4, the mode-shapes in air and water can be compared in a MAC-table showing the similarity between the referencing mode-shapes (Table 4.16 and 4.15).

Table 2.4: Auto MAC of experimental mode-shapes in water using the LMS-system in strain.

vs.			LMS Experimental mode-shapes in water					
		nr.	1	2	3	4	5	6
	nr.	[freq.]	[54.40]	[55.37]	[99.03]	[130.35]	[135.83]	[179.22]
LMS Experimental mode-shapes in water	1	[54.40]	1.000	0.001	0.000	0.052	0.005	0.044
	2	[55.37]	0.001	1.000	0.008	0.085	0.032	0.017
	3	[99.03]	0.000	0.008	1.000	0.002	0.004	0.037
	4	[130.35]	0.052	0.085	0.002	1.000	0.073	0.050
	5	[135.83]	0.005	0.032	0.004	0.073	1.000	0.078
	6	[179.22]	0.044	0.017	0.037	0.050	0.078	1.000

2.6 Comments, Conclusions & Recommendations

2.6.1 Comments

The discovery, that the initial mode-shapes were not mass normalised arose only in the end of the studies at EDF. Therefore, there was no time to properly redo the measurements. The resolution to use the 'hand'-normalised mode-shapes is just done to be able to continue the study, but is a major flaw considering the quality of the study.
A remark need to be made considering the two practical ways to 'hand' mass normalise the mode-shapes; method (1), in which the mode-shapes are mass normalised by using the modal mass factor in the data file can only be used for experiments in displacement, because the function `NORM_MODE` used in Code_Aster to mass normalise the mode-shapes

read with LIRE_RESU does not work for strain measurements.
The use of a 2-directional space to do the expansions later on, avoid the necessity to split the measurements of the LMS system on the strain gauges into two separate directions, as was done in the study by Bodel [1]. In this study, the measurements obtained by the PAK-system must be copied in the second direction to be able to do the expansion on the numerical 2-directional space.
Realising that the method Polymax uses to obtain mode-shapes from measurements when the exciting force is not measured, has invoked some doubt in the general process of this study. Polymax only uses a random signal in both space and time when the true exciting force cannot be measured, to obtain the mode-shapes, while the true fluid force is really exciting the tube. The obtained mode-shapes will therefore not represent the system correct, but when the fluid force shows a quite random character, the model-shapes will be quite good. More important is, that from a operational modal analysis point of view, the system represented by the obtained mode-shapes, is a system that would give the measured response if the force would be a random signal in both space and time. If this system is then inverted to find the 'fluid force', one can expect to find back the random signal and not the true fluid force.

2.6.2 Conclusions

The MAC-numbers of the measurements in air show that the eigen mode-shapes obtained with either the PAK-system on the accelerometers, or the LMS-system on the strain gauges, are orthogonal. Because both measurement systems give similar MAC-numbers, the strain measurement system is shown to be good enough to obtain the modal parameters, even though it has only eight sensors in one direction and not fourteen as the PAK-system has.
The MAC-numbers of the eigen mode-shapes obtained from the LMS measurements in water show a slightly less orthogonality, compared to the ones in air, but are still classified as orthogonal. There are no non-system related mode-shapes identified, so the Fourier transform of the fluid force is assumed to be rather flat.
Determining that the mode-shapes obtained by the PAK-system were not mass normalised is an important observation. Also considering the study by Bodel [1]. Bodel used the measurement data as they were and a factor 1.0 for the modal mass of the mode-shapes in air, this made him induce a scaling error to the mode-shapes and therefore the modal forces. The summation of these differences made the finally found modelled fluid forces false.

2.6.3 Recommendations

Light should be shed on why the mode-shapes were not mass normalised by the measurement system and why also the efforts to mass normalise the FRF-data using the MATLAB SDtools [17], had no desirable results. The general thought so far is that it probably has got something to do with calibration of the set-up or maybe due to different units being mixed up. After identifying the problem, the measurements should be redone.
The function NORM_MODE in Code_Aster should be adjusted to be able to directly normalise strain measurements.

Chapter 3

The Numerical Models

3.1 Introduction

Finite Element Modelling is a discretization technique to describe the real model in discrete elements. Using these finite elements, one can make a computer model to do static and dynamic analyses. There are two numerical models of the Phacetie model; a 3D-volume element model, which already existed from Bodels [1] study, and a 2D-shell element model, which is built for this study. The mode-shape spaces from these numerical models are used later on, to expand the experimentally obtained mode-shapes.
The modelling of the springs, from which the tube is hung in the Phacetie model, is adjusted to make the numerical models suitable for analysis in 2-directions simultaneously, in stead of only in 1-direction.
Also the spring stiffness is decided upon, using an iterative process to get the frequencies of the numerical models close to the experimental ones. When using a discrete numerical model compared to an analytical analysis, there is a difference called 'The added stiffness due to modelling'. This iterative process takes care of this added stiffness due to modelling and is described in appendix C.1. This method is used to iterate to a reasonable spring stiffness for the numerical model.
Finally the two models are compared. First, the auto-MAC-numbers between the numerical eigen mode-shapes of the two models are calculated to check if the MAC, without a mass matrix as a weighting matrix, shows a good orthogonal property. Then the two numerical models are compared to one another in two different ways. First, the displacement fields of the numerical models are extracted, reducing the models to have the same DoFs in the field and then they are compared. And second, a method projecting the numerical mode-shapes on a 1D bar model, after which the two numerical models can be compared again. The choice of which numerical model to use during further studies is postponed until after the expansions are compared, to be sure that the overall best numerical model is picked.
First the two numerical models are explained, in section 3.2. Then in section 3.3 the modelling of the springs is described. In the final section, section 3.4 the two different numerical models are compared. This chapter is finished with comments, conclusions and recommendations.

3.2 The Two Numerical Models

Both models are discretized in a similar manner; both have 217 element along the length of the tube, each with a length of 1 centimetre. There are 20 elements along the circumference of the cross-section of the tube.

3.2.1 3D Volume-element Model

The 3-D model consists of hexahedron elements as depicted in figure 3.1. The nodes

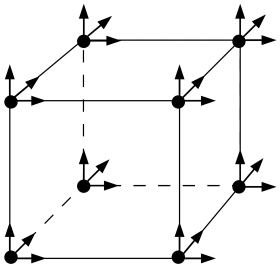


Figure 3.1: 3D Reference Volume-element with nodes, each having three DoF.

are situated in the corners and have only three translational DoF each. The rod model with 3-D elements is depicted in figure 3.2. The thickness of the tube is modelled with 2 elements.

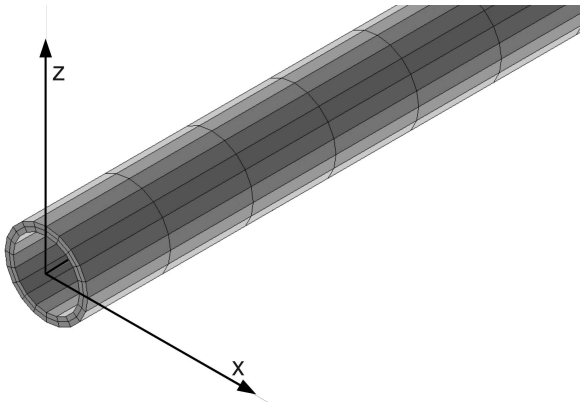


Figure 3.2: Mesh of the tube to use with 3D Volume-elements

3.2.2 2D Reference Shell-element Model

The 2-D model has a mesh of nodes which is like a surface, for the tube it look a bit like a paper roll. On each rectangle a shell element is placed as depicted in figure 3.3. Each node has six DoF (the three translational ones and the three rotational ones). The

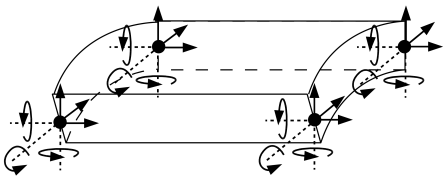


Figure 3.3: 2D shell-element with nodes, each having six DoF.

shell elements used are discrete Kirschoff elements, equivalent to a Euler-Bernoulli beam modelling. The 2-D model with the mesh for shell elements is depicted in figure 3.4.

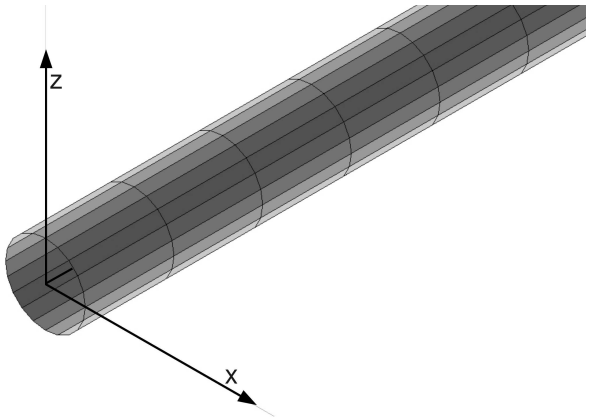


Figure 3.4: Mesh of the tube to use with 2D Shell-elements

3.3 Modelling of the Springs

The springs of the Phacetie model will be modelled by applying stiffness to the nodes at the positions where the springs are connected to the tube. A table containing the node-numbers of the nodes used to model the springs in the numerical models, is shown in appendix C.4.

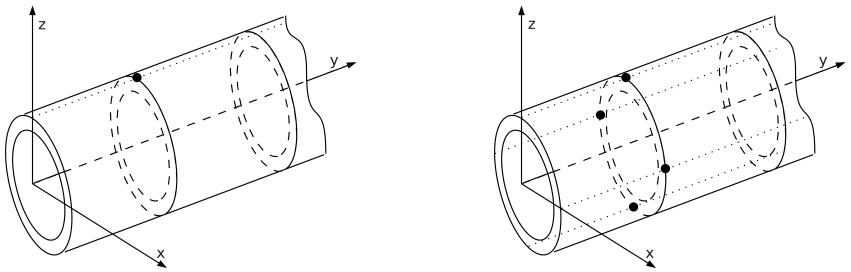


Figure 3.5: (a) One node modelling a spring, and (b) Four nodes modelling a spring

3.3.1 Spring Model

In Bodels study [1] each spring was modelled by prescribing a stiffness to one node on the circumference of the cross section as depicted in figure 3.5(a). For all DoF of this node, transfers and rotational, a stiffness was prescribed. By modelling the springs in this way, there will be a difference in the behaviour of the tube in x-direction, compared to the z-direction. For the analysis in both directions simultaneously, the tube should behave the same in both directions and thus the modelling of the springs needs to be improved. This can be done by adding a node to the middle of the cross-sectional area or by assigning four nodes on the circumference of the cross-section. While there is no node in the centre in the numerical models, four nodes will be assigned for each spring, as see figure 3.5(b).
With this node configuration there will be no rotational stiffness assigned to the nodes, because the rotational stiffness is indirectly taken care of by the axial stiffness prescribed on the nodes. In this configuration the stiffness in axial direction, together with the radius, the distance to the centre of the cross-section, will take account for the rotational stiffness.

3.3.2 Spring Stiffness

The stiffness of the springs has been a troubling factor. The stiffness data for the different directions was given in threefold as shown in table 3.1. The translational stiffness, the axial stiffness and the bending stiffness were given. These data come from the test department of EDF at Chatoux, France. These different values were of no help defining

Table 3.1: Spring stiffness data from test department of EDF at Chatoux.

	Visco analytic value	Theoretical value	Manufacturers value
$k_x,\ k_z$	$3 \cdot 10^6$ [N/m]	$1.8 \cdot 10^7$ [N/m]	$1.05 \cdot 10^6$ [N/m]
k_y	$1.4 \cdot 10^6$ [N/m]	$1.25 \cdot 10^6$ [N/m]	$1. \cdot 10^6$ [N/m]
k_θ	20 [Nm/rad]	41.3 [Nm/rad]	33 [Nm/rad]

the spring constants in the numerical model. So, to find realistic values for the spring stiffness an iterative process was used aiming for newly set experimental frequencies. These newly set frequencies take into account the added stiffness due to modelling. This process is described in appendix C.1. In order to get some insight in the ratio between the translational and the axial spring stiffness a simple model is obtained from appendix C.3. This process is basically a form of updating (depicted on the right hand side of the scheme in figure 1.6), but was necessary to identify a suitable spring stiffness. In table 3.2 and 3.3 the data used in the iteration process is shown. The spring stiffness for which both numerical models came closest to the newly set aiming frequencies is per node: $k_x = k_z = 1.8 \cdot 10^6$ and $k_y = 2.1 \cdot 10^7$. The ratio between the translational and the rotational stiffness comes to within the order of the 11.3% as predicted in appendix C.3.

Table 3.2: Table showing the factorisation of the 'added stiffness due to modelling' for the 3D volume-element model and the calculation of the numerical 'aiming' frequencies (bold) for the iteration process.

1	2	3	4	Mode-shape number		
14.31	39.44	77.30	127.75	Numerical frequencies [Hz]		
12.56	34.63	67.88	112.21	Real frequencies [Hz]		
1.14				70.26	**80.03**	1
	1.14			111.89	**127.44**	2
		1.14		154.69	**176.15**	3
			1.14	202.61	**230.65**	4

Table 3.3: Table showing the factorisation of the 'added stiffness due to modelling' for the 2D shell-element model and the calculation of the numerical 'aiming' frequencies (bold) for the iteration process.

1	2	3	4	Mode-shape number		
13.38	36.86	72.22	119.33	Numerical frequencies [Hz]		
12.56	34.63	67.88	112.21	Real frequencies [Hz]		
1.07				70.26	**74.85**	1
	1.06			111.89	**119.09**	2
		1.06		154.69	**164.57**	3
			1.06	202.61	**215.46**	4

3.4 Comparing the Two Models

Before the eigen mode-shapes of the two numerical models can be compared, MAC-numbers calculated without a mass matrix in Python need to be checked, to see if they give a similar good result as the ones calculated with the mass matrix as weighting matrix.

This because when using mode-shapes in strain the mass matrix cannot be used and, next to that, the mass matrix is too big to be handled straightforward in Python .
Then the two models can be compared. Keeping in mind that the 2-D model has rotational degrees of freedom, and the 3-D model does not, one cannot compare the models directly. By extracting the nodal displacement from both models in Python and aligning their nodes, a MAC can be made of displacement or strain mode-shapes. Another way of comparing the two models is by projecting them on an 1-D model of a bar and making a MAC between the two different projections.

3.4.1 Comparing the Auto-MAC of the Two Numerical Models

In the algorithm to obtain the numerical eigen mode-shapes from a numerical model, the orthogonality with respect to the mass matrix is used. So, if an auto-MAC-table would be calculated for the numerical eigen mode-shapes with the use of the mass matrix, the result would be a perfect identity matrix. However, Code_Aster is not capable of calculating MAC-table for mode-shapes in strain and thus Python is used to calculate these MAC-tables. Python is not capable of handling the mass matrix and, next to that, discrete mode-shapes in strain are not orthogonal with respect to the mass matrix. Appendix A.1 shows that their continues counterpart should be orthogonal with respect to other continues mode-shapes in strain. Therefore, when the mass matrix is 'lumped', the discrete mode-shape in displacement should be quite orthogonal. Then also the discrete mode-shape in strain should be quite orthogonal. This can be tested using the Python built MAC-number routine, shown in appendix D. The auto MAC for the eigen mode-shapes from the 3D volume-element numerical model, is shown in table 3.4. Again, by definition of the numerical model, the auto-MAC numbers would form an identity matrix if the mass matrix was used and so, by calculating this auto-MAC without the mass matrix, it gives an indication of how good the result still is, doing this approximation.

Table 3.4: Auto MAC of 3D Model Full in Displacement.

vs.		3D Model Full							
	nr. [freq.]	1 [74.58]	2 [74.58]	3 [133.54]	4 [133.54]	5 [177.97]	6 [177.97]	7 [230.34]	8 [230.34]
3D Model Full	1 [74.58]	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	2 [74.58]	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000
	3 [133.54]	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000
	4 [133.54]	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000
	5 [177.97]	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000
	6 [177.97]	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000
	7 [230.34]	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000
	8 [230.34]	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000

The strain fields of these mode-shapes can easily be calculated in Code_Aster . The auto-MAC-numbers with the mode-shapes in strain are shown in table 3.5. They have a similar results to the auto-MAC-numbers in displacement. So, also in strain, MAC-

numbers calculated without a mass matrix, give a good approximation.
The auto-MAC-numbers for the 2D shell-element model for displacement and strain are

Table 3.5: Auto MAC of 3D Model Full in Strain.

vs.		3D Model Full							
	nr.	1	2	3	4	5	6	7	8
	nr. [freq.]	[74.58]	[74.58]	[133.54]	[133.54]	[177.97]	[177.97]	[230.34]	[230.34]
3D Model Full	1 [74.58]	1.000	0.000	0.001	0.001	0.007	0.001	0.002	0.000
	2 [74.58]	0.000	1.000	0.001	0.001	0.001	0.007	0.000	0.002
	3 [133.54]	0.001	0.001	1.000	0.000	0.001	0.000	0.000	0.000
	4 [133.54]	0.001	0.001	0.000	1.000	0.000	0.001	0.000	0.000
	5 [177.97]	0.007	0.001	0.001	0.000	1.000	0.000	0.000	0.000
	6 [177.97]	0.001	0.007	0.000	0.001	0.000	1.000	0.000	0.000
	7 [230.34]	0.002	0.000	0.000	0.000	0.000	0.000	1.000	0.000
	8 [230.34]	0.000	0.002	0.000	0.000	0.000	0.000	0.000	1.000

in table 3.6 and 3.7 respectively.

Table 3.6: Auto MAC of 2D Model Full in Displacement.

vs.		2D Model Full							
	nr.	1	2	3	4	5	6	7	8
	nr. [freq.]	[69.11]	[69.11]	[124.23]	[124.23]	[165.99]	[165.99]	[215.84]	[215.84]
2D Model Full	1 [69.11]	1.000	0.000	0.001	0.000	0.005	0.002	0.005	0.002
	2 [69.11]	0.000	1.000	0.000	0.001	0.002	0.005	0.002	0.005
	3 [124.23]	0.001	0.000	1.000	0.000	0.014	0.000	0.003	0.000
	4 [124.23]	0.000	0.001	0.000	1.000	0.000	0.014	0.000	0.003
	5 [165.99]	0.005	0.002	0.014	0.000	1.000	0.000	0.002	0.000
	6 [165.99]	0.002	0.005	0.000	0.014	0.000	1.000	0.000	0.002
	7 [215.84]	0.005	0.002	0.003	0.000	0.002	0.000	1.000	0.000
	8 [215.84]	0.002	0.005	0.000	0.003	0.000	0.002	0.000	1.000

Again, the MAC-numbers for displacement and strain are calculated without the mass matrix and so show that these MAC-numbers still are showing orthogonality.
So far, both numerical models show good orthogonal properties for their respective eigen mode-shapes.

3.4.2 MAC after Extracting Nodal Displacement

After extracting the nodal displacement of both the 3D volume-element and the 2D shell-element model, their cross MAC numbers can be calculated. The auto-MAC-numbers of

Table 3.7: Auto MAC of 2D Model Full in Strain.

vs.		2D Model Full							
	nr. [freq.]	1 [69.11]	2 [69.11]	3 [124.23]	4 [124.23]	5 [165.99]	6 [165.99]	7 [215.84]	8 [215.84]
2D Model Full	1 [69.11]	1.000	0.000	0.001	0.000	0.005	0.002	0.001	0.001
	2 [69.11]	0.000	1.000	0.000	0.001	0.002	0.005	0.001	0.001
	3 [124.23]	0.001	0.000	1.000	0.000	0.001	0.000	0.000	0.000
	4 [124.23]	0.000	0.001	0.000	1.000	0.000	0.001	0.000	0.000
	5 [165.99]	0.005	0.002	0.001	0.000	1.000	0.000	0.000	0.000
	6 [165.99]	0.002	0.005	0.000	0.001	0.000	1.000	0.000	0.000
	7 [215.84]	0.001	0.001	0.000	0.000	0.000	0.000	1.000	0.000
	8 [215.84]	0.001	0.001	0.000	0.000	0.000	0.000	0.000	1.000

these reduced mode-shapes give similar results as the MAC-numbers of the full mode-shapes[1]. The cross-MAC-numbers are shown in table 3.8 and 3.9. These MAC-tables are again calculated without a mass matrix, because it is different for both the 2D and 3D model, and it is simply too big to handle straight forward in Python .

Table 3.8: Cross MAC of 3D Model Reduced vs 2D Model Reduced in Displacement.

vs.		3D Model Reduced							
	nr. [freq.]	1 [74.58]	2 [74.58]	3 [133.54]	4 [133.54]	5 [177.97]	6 [177.97]	7 [230.34]	8 [230.34]
2D Model Reduced	1 [69.11]	0.931	0.069	0.000	0.000	0.000	0.000	0.000	0.000
	2 [69.11]	0.069	0.931	0.000	0.000	0.000	0.000	0.000	0.000
	3 [124.23]	0.000	0.000	0.286	0.714	0.000	0.000	0.000	0.000
	4 [124.23]	0.000	0.000	0.714	0.286	0.000	0.000	0.000	0.000
	5 [165.99]	0.000	0.000	0.000	0.000	0.715	0.285	0.000	0.000
	6 [165.99]	0.000	0.000	0.000	0.000	0.285	0.715	0.000	0.000
	7 [215.84]	0.000	0.000	0.000	0.000	0.000	0.000	0.760	0.240
	8 [215.84]	0.000	0.000	0.000	0.000	0.000	0.000	0.240	0.760

Similarly, after extracting the nodal strain data for similar nodes of the two numerical models, also the strain auto-MAC numbers can be calculated as depicted in table 3.9.
In both cross-MAC tables the entries are not just ones and zeros for reasons explained in appendix C.2. For the entries where the mode-shapes pairs of the two models interact, the columns and rows added up to about one, which shows the similarity between both

[1]For the 2D model, the reduced shapes even give a better result, because the rotational DoF, which are not present in the reduced form, are responsible for a very slight disturbance in the full analysis without the mass matrix as weighting matrix.

Table 3.9: Cross MAC of 3D Model Reduced vs 2D Model Reduced in Strain.

vs.		3D Model Reduced							
	nr. [freq.]	1 [74.58]	2 [74.58]	3 [133.54]	4 [133.54]	5 [177.97]	6 [177.97]	7 [230.34]	8 [230.34]
2D Model Reduced	1 [69.11]	0.860	0.064	0.001	0.000	0.007	0.000	0.002	0.000
	2 [69.11]	0.064	0.860	0.000	0.001	0.000	0.007	0.000	0.002
	3 [124.23]	0.001	0.000	0.265	0.660	0.000	0.001	0.000	0.000
	4 [124.23]	0.000	0.001	0.660	0.265	0.001	0.000	0.000	0.000
	5 [165.99]	0.006	0.000	0.000	0.000	0.661	0.263	0.000	0.000
	6 [165.99]	0.000	0.006	0.000	0.000	0.263	0.661	0.000	0.000
	7 [215.84]	0.002	0.000	0.000	0.000	0.000	0.000	0.703	0.221
	8 [215.84]	0.000	0.002	0.000	0.000	0.000	0.000	0.221	0.703

models. These MAC numbers calculated from the obtained numerical eigen mode-shapes from the two models show an orthogonal behaviour with respect to each other.

3.4.3 MAC after Projecting on 1-D Model

It is also possible to make a MAC-table of the two different numerical models by projecting[2] the displacements on a 1D bar-element model and then calculating the MAC between the different projected models. A bar-element model is used because the 3D volume-element model just has three degrees of freedom. A beam model would require 6 degrees of freedom (also the rotational ones) and they are not present in the 3D model. But, by using a bar-element it is not possible to extract a useful strain component as in the original models. In a bar model there is no bending, so no bending stress and so no strain due to bending can be calculated. So, strain MAC-numbers are not possible.
The auto-MAC-table of the projected mode-shapes is similar to the one of the full mode-shapes of the different numerical models. The cross-MAC-numbers are, again, calculated without using the mass matrix, and depicted in table 3.10.

3.5 Comments, Conclusions & Recommendations

3.5.1 Comments

The numerically obtained eigen frequencies are different from the experimentally obtained ones. For this reason efforts were made to improve the numerical model by adding mass at the positions of the springs and at the positions of the accelerometers. By adding mass to the nodes in the numerical model at the spring and accelerometer positions, slightly

[2] 'projection' here is not the mathematical principle, but rather projection in the sense of projecting or expanding (see chapter 4) given data on a bases. If the 1-D model has the same number of DoF in the y-direction as the different numerical models, this projection becomes a one on one transformation of the data.

Table 3.10: Cross MAC of 3D Model Projected vs 2D Model Projected in Displacement.

vs.		3D Model Projected							
	nr. [freq.]	1 [74.58]	2 [74.58]	3 [133.54]	4 [133.54]	5 [177.97]	6 [177.97]	7 [230.34]	8 [230.34]
2D Model Projected	1 [69.11]	0.929	0.069	0.000	0.000	0.000	0.000	0.000	0.000
	2 [69.11]	0.069	0.930	0.000	0.000	0.000	0.000	0.000	0.000
	3 [124.23]	0.000	0.000	0.286	0.713	0.000	0.000	0.000	0.000
	4 [124.23]	0.000	0.000	0.713	0.286	0.000	0.000	0.000	0.000
	5 [165.99]	0.000	0.000	0.000	0.000	0.714	0.284	0.000	0.000
	6 [165.99]	0.000	0.000	0.000	0.000	0.285	0.715	0.000	0.000
	7 [215.84]	0.000	0.000	0.000	0.000	0.000	0.000	0.759	0.239
	8 [215.84]	0.000	0.000	0.000	0.000	0.000	0.000	0.239	0.760

different frequencies were obtained. These differences in frequencies did not show any consistent improvement and were therefore not further implemented.

3.5.2 Conclusions

The cross-MAC-tables of the 'reduced' MAC, table 3.9 and the 'projected' MAC, table 3.10 give a similar good result. The 'reduction'-method, makes it also possible to calculating MAC-numbers for mode-shapes in strain, while the 'projecting'-method can not. But more important, whether the method of reduction or projection is used, both numerical models, when compared to each other, show a good orthogonality property. So, both numerical models give similar eigen mode-shapes.
The 2D shell-element model is slightly faster, in computational term speaking and the 3D volume-element model gives a slightly better auto-MAC number table when no matrix is used as a weighting matrix. For now, there is no decision made which numerical model to prefer, this is left until after the expansions are made, to first be able to see how the mode-shape spaces of both models preform in the expansion.

3.5.3 Recommendations

Although for this study the numerical models are only used to provide eigen mode-shapes to form an expansion space, the numerical models could be updated using parameter updating, schematically shown on the right hand side of figure 1.6. This to improve the expansion space for the experimentally obtained mode-shapes.

Chapter 4

Expanding and Comparing the Experimental Mode shapes

4.1 Introduction

To better discretize the experimentally obtain mode-shapes, they can be expanded in the numerical spaces obtained in chapter 3. An expansion is basically fitting the numerical shapes, spanning the space, on the measurement data. This is done by using an algorithm which minimizes equation 4.1.

$$\min \left\| \sum_{i=1}^{m} a_{ij} \mathbf{C}^{\mathrm{a}} \boldsymbol{\phi}_{(i)}^{\mathrm{num}} - \boldsymbol{\phi}_{(j)}^{\mathrm{ex,a}} \right\|^2 \quad \text{for } j = 1, 2, 3, \ldots, M \tag{4.1}$$

where $\mathbf{C}^{\mathrm{a}}_{N \times n}$ is a boolean matrix identifying the positions of the N accelerometers along the numerical eigen mode-shapes, $\boldsymbol{\phi}^{\mathrm{num}}_{(i)\ n\times 1}$, with n DoF. The $\boldsymbol{\phi}^{\mathrm{ex,a}}_{(j)\ N\times 1}$ is the experimentally obtained eigen mode-shape obtained using the N accelerometers.
So, by minimizing the values for the factors a_{ij} are obtained. In the equation there are m numerical eigen mode-shapes $\boldsymbol{\phi}^{\mathrm{num}}_{(i)}$. And a_{ij} is the factor for the i^{th} numerical ($^{\mathrm{num}}$) eigen mode-shapes to express the j^{th} experimental eigen mode-shapes. These M mode-shapes $\boldsymbol{\phi}^{\mathrm{ex,a}}_{(j)}$, are the experimentally ($^{\mathrm{ex}}$) obtained eigen mode-shapes in displacement, making use of accelerometers(^{a}) and the PAK-system.
To be able to identify the participation factors a_{ij}, there must be equal number or more experimental data N, than the number of numerical shapes m, on which they are fitted

$$m \leqslant N. \tag{4.2}$$

The expanded ($^{\mathrm{et}}$) experimental mode-shapes $\boldsymbol{\phi}^{\mathrm{et}}_{j}$ are now defined, using the factors a_{ij} as

$$\boldsymbol{\phi}^{\mathrm{et}}_{j} = \sum_{i=1}^{m} \boldsymbol{\phi}^{\mathrm{num}}_{i} a_{ij}. \tag{4.3}$$

Expanding the experiments in strain, obtained using the LMS-system, will give

$$\min \left\| \sum_{i=1}^{m} g_{ij} \mathbf{C}^{\mathrm{s}} \boldsymbol{\psi}_{(i)}^{\mathrm{num}} - \boldsymbol{\psi}_{(j)}^{\mathrm{ex,s}} \right\|^2 \quad \text{for } j = 1, 2, 3, \ldots, M \tag{4.4}$$

and the expanded mode-shapes become

$$\boldsymbol{\psi}_j^{\mathrm{et}} = \sum_{i=1}^{m} \boldsymbol{\psi}_i^{\mathrm{num}} g_{ij} \tag{4.5}$$

The equations 4.1 and 4.4 are mathematically the same as the pseudo-inverses as explained in appendix A.3 and [2], respectively

$$\mathbf{A} = \left[\mathbf{C}^{\mathrm{a}} \boldsymbol{\Phi}^{\mathrm{num}}\right]^{+} \boldsymbol{\Phi}^{\mathrm{ex,a}} \qquad \mathbf{G} = \left[\mathbf{C}^{\mathrm{s}} \boldsymbol{\Psi}^{\mathrm{num}}\right]^{+} \boldsymbol{\Psi}^{\mathrm{ex,s}} \tag{4.6}$$

and the expansions in matrix form become

$$\boldsymbol{\Phi}^{\mathrm{et,a}} = \boldsymbol{\Phi}^{\mathrm{num}} \mathbf{A} \qquad \boldsymbol{\Psi}^{\mathrm{et,s}} = \boldsymbol{\Psi}^{\mathrm{num}} \mathbf{G}. \tag{4.7}$$

But the measurements always contain some error. Due to this error, the higher numerical eigen mode-shapes can be wrongly overestimated in the expansion. This is phenomenon obviously occurs more using only eight strain gauges able to measure movement in one direction, than when using the fourteen accelerometers. As a measure to avoid these problems, only the numerical eigen mode-shapes, which are in the same frequency range as the experimental ones, which are about to be expanded, will be used in the mode-shapes space. Therefore, only the first four numerical eigen mode-shapes will be used in the mode-shape space, the same ones as which were tested in chapter 3.
Another way to counteract this problem, is to use static-shapes in the expansion space. Static shapes are obtained by prescribing a zero displacement to specific points along the tube and then, one by one placing a unit displacement on each specific point. This way, a static-shape space is developed, containing the as many static-shapes as there are prescribed points along the tube.
In this chapter, the experiments will be expanded on eigen mode-shapes; numerical and earlier expanded experimental spaces, and on static-shapes; having their prescribed points on measurement device positions as well as on spring positions. After the expansion on these different spaces, the experiments are tested by the auto-MAC criterion, to see if they still show their orthogonal property. They are also visually checked to see if the mode-shapes are still smooth, as expected from the physical theory.
In table 4.1 all the different possibilities of expansions, of the different experiments on the different expansion spaces, are shown. The table refers to the auto-MAC-table in which the specific expansion is tested, or if it is not possible in Code_Aster (not poss.), or if it is just not shown (not shown).

After deciding which mode-shape space to use for the expansion, the experimental mode-shapes in air and in water can be compared, as well as the different measurement systems. This will be done, again, with the MAC criterion as well as by visual conformation. The two numerical models were already compared in chapter 3 and showed little difference. Now they will be compared after they are used as a mode-shape expansion space, to see if show a different result.

Table 4.1: Table showing a reference for an auto-MAC-table for the different possibilities for the expansion of the different experimental data on the different mode-shape spaces

Auto-MAC lookup table		Expansion Spaces							
		Eigen(4.2)				Static(4.3)			
		PAK Earlier		Numerical					
		Expand(4.2.2)		(4.2.1)		Spring(4.3.1)		Sensor(4.3.2)	
		3D	2D	3D	2D	3D	2D	3D	2D
Experimental mode-shape data	Accelerometer PAK in Air	not shown	not shown	MAC 4.2	MAC 4.3	MAC 4.9	MAC 4.10	MAC 4.11	MAC 4.12
	Strain gauges LMS in Air	MAC 4.8	not shown	MAC 4.4	MAC 4.5	not poss.	not poss.	not poss.	not poss.
	Strain gauges LMS in Water	not shown	not shown	MAC 4.6	MAC 4.7	not poss.	not poss.	not poss.	not poss.

In the first section, section 4.2, eigen mode-shapes spaces are tested, first numerically obtained eigen mode-shapes are used as a space, then earlier expanded experimental eigen mode-shapes are used as a space. Section 4.3 shows the static-shapes, one with the boundary conditions applied on the spring positions and one with them applied on the accelerometer positions. Then all the different comparisons can be made: The eigen mode-shape spaces versus static-shape spaces in section 4.4. For the 3D volume-element model versus the 2D shell element model a final decision is made in section 4.5. The mode-shapes obtained with the accelerometers and the PAK-system versus the ones obtained with strain gauges and the LMS-system are shown in section 4.6. And last, in section 4.7, the mode-shapes in air versus the mode-shapes in water are compared. This is of major importance for the rest of the study. The comments and conclusions are in the final section, section 4.8, of this chapter.

4.2 Eigen Mode-shapes as Space

Eigen mode-shapes extracted form the numerical models, can be used in different ways, either direct or indirect. The direct way is to use the eigen mode-shapes of the numerical model to span the space, an the indirect way is to use already expanded experimental mode-shapes as the space to expand other measurements on. So, for example, first expand the LMS strain measurements on the numerical eigen mode-shape space, and than these expanded mode-shapes are than used as a space to expand the PAK displacement shapes. Keep in mind that the first expansion is done in strain, thus, to continue, the expanded experimental mode-shapes need to be put in displacement, before they can be used as a space to expand the PAK displacement data. This all seems a bit of a useless hustle maybe, but it will show the incentive for questioning the mass normalisation of the original measurements.

4.2.1 Numerical Eigen Mode-shape Space

In table 4.2 and 4.3, the MAC-numbers are shown of the experimental mode-shapes in air, obtained using the accelerometers and the PAK-system, expanded on the 3D volume-element model and on the 2D shell-element model. The MAC-tables are calculated for the mode-shapes in displacement.

Table 4.2: Auto MAC of PAK experimental mode-shapes in air, expanded on mode-shapes of 3D model in displacement

vs.		PAK experiments in air, expanded on 3D model							
	nr. [freq.]	1 [70.26]	2 [70.26]	3 [111.89]	4 [111.89]	5 [154.69]	6 [154.69]	7 [202.61]	8 [202.61]
PAK experiments in air, expanded on 3D model	1 [70.26]	1.000	0.000	0.000	0.000	0.001	0.000	0.001	0.000
	2 [70.26]	0.000	1.000	0.000	0.000	0.000	0.001	0.000	0.001
	3 [111.89]	0.000	0.000	1.000	0.000	0.005	0.000	0.000	0.000
	4 [111.89]	0.000	0.000	0.000	1.000	0.000	0.005	0.000	0.000
	5 [154.69]	0.001	0.000	0.005	0.000	1.000	0.000	0.001	0.000
	6 [154.69]	0.000	0.001	0.000	0.005	0.000	1.000	0.000	0.001
	7 [202.61]	0.001	0.000	0.000	0.000	0.001	0.000	1.000	0.000
	8 [202.61]	0.000	0.001	0.000	0.000	0.000	0.001	0.000	1.000

Table 4.3: Auto MAC of PAK experimental mode-shapes in air, expanded on mode-shapes of 2D model in displacement

vs.		PAK experiments in air, expanded on 2D model							
	nr. [freq.]	1 [70.26]	2 [70.26]	3 [111.89]	4 [111.89]	5 [154.69]	6 [154.69]	7 [202.61]	8 [202.61]
PAK experiments in air, expanded on 2D model	1 [70.26]	1.000	0.000	0.002	0.000	0.004	0.000	0.009	0.000
	2 [70.26]	0.000	1.000	0.000	0.002	0.000	0.004	0.000	0.009
	3 [111.89]	0.002	0.000	1.000	0.000	0.041	0.000	0.002	0.000
	4 [111.89]	0.000	0.002	0.000	1.000	0.000	0.041	0.000	0.002
	5 [154.69]	0.004	0.000	0.041	0.000	1.000	0.000	0.007	0.000
	6 [154.69]	0.000	0.004	0.000	0.041	0.000	1.000	0.000	0.007
	7 [202.61]	0.009	0.000	0.002	0.000	0.007	0.000	1.000	0.000
	8 [202.61]	0.000	0.009	0.000	0.002	0.000	0.007	0.000	1.000

The experimental mode-shapes obtained using the LMS-system in air, expanded on the different numerical models are shown in table 4.5 and 4.4. The mode-shapes used are, just like the measurement data, in strain.

The mode-shapes from the experiments in water, and thus again obtained with the LMS-system and strain gauges, are expanded on both numerical models. Their MAC-tables are shown in table 4.6 and 4.7. These MAC-table are also calculated with the

Table 4.4: Auto MAC of LMS experimental mode-shapes in air, expanded on mode-shapes of 3D model in strain

vs.		LMS experiments in air, expanded on 3D model							
	nr. [freq.]	1 [70.19]	2 [70.19]	3 [111.89]	4 [111.89]	5 [154.70]	6 [154.70]	7 [202.60]	8 [202.60]
LMS experiments in air, expanded on 3D model	1 [70.19]	1.000	0.000	0.011	0.000	0.000	0.000	0.096	0.000
	2 [70.19]	0.000	1.000	0.000	0.011	0.000	0.000	0.000	0.096
	3 [111.89]	0.011	0.000	1.000	0.000	0.007	0.000	0.050	0.000
	4 [111.89]	0.000	0.011	0.000	1.000	0.000	0.007	0.000	0.050
	5 [154.70]	0.000	0.000	0.007	0.000	1.000	0.000	0.081	0.000
	6 [154.70]	0.000	0.000	0.000	0.007	0.000	1.000	0.000	0.081
	7 [202.60]	0.096	0.000	0.050	0.000	0.081	0.000	1.000	0.000
	8 [202.60]	0.000	0.096	0.000	0.050	0.000	0.081	0.000	1.000

Table 4.5: Auto MAC of LMS experimental mode-shapes in air, expanded on mode-shapes of 2D model in strain

vs.		LMS experiments in air, expanded on 2D model							
	nr. [freq.]	1 [70.19]	2 [70.19]	3 [111.89]	4 [111.89]	5 [154.70]	6 [154.70]	7 [202.60]	8 [202.60]
LMS experiments in air, expanded on 2D model	1 [70.19]	1.000	0.000	0.007	0.000	0.000	0.000	0.095	0.000
	2 [70.19]	0.000	1.000	0.000	0.008	0.000	0.000	0.000	0.095
	3 [111.89]	0.007	0.000	1.000	0.000	0.013	0.001	0.031	0.000
	4 [111.89]	0.000	0.008	0.000	1.000	0.000	0.013	0.000	0.031
	5 [154.70]	0.000	0.000	0.013	0.000	1.000	0.000	0.080	0.000
	6 [154.70]	0.000	0.000	0.001	0.013	0.000	1.000	0.001	0.080
	7 [202.60]	0.095	0.000	0.031	0.000	0.080	0.001	1.000	0.000
	8 [202.60]	0.000	0.095	0.000	0.031	0.000	0.080	0.000	1.000

mode-shapes in strain.

4.2.2 Earlier Expanded Experimental Mode-shape Space

When using one expanded experimental eigen mode-shapes as space for the expansion of the other, the auto-MAC-numbers of the expanded mode-shapes are exactly the same as the ones directly expanded on the numerical eigen mode-shape bases. Compare for example MAC-table 4.4 with 4.8. This is due to the fact that, when doing the second expansion, one obtains a new set of factors, which forms multiplied with the first set of factors, the same set of factors as in a single, direct expansion.
To make an expansions of the LMS, strain gauge measurements, on the already expanded PAK, accelerometer measurements, the expanded PAK measurements must be put in strain. This is easily done in Code_Aster, and so $\boldsymbol{\psi}_{(i)}^{\mathrm{et,a}}$ are the mode-shapes in strain, cal-

Table 4.6: Auto MAC of LMS experimental mode-shapes in water, expanded on mode-shapes of 3D model in strain

vs.		LMS experiments in water, expanded on 3D					
	nr.	1	2	3	4	5	6
	[freq.]	[54.40]	[55.37]	[99.03]	[130.35]	[135.83]	[179.22]
LMS experiments in water, expanded on 3D	1 [54.40]	1.000	0.022	0.015	0.011	0.015	0.074
	2 [55.37]	0.022	1.000	0.002	0.056	0.025	0.346
	3 [99.03]	0.015	0.002	1.000	0.045	0.193	0.132
	4 [130.35]	0.011	0.056	0.045	1.000	0.334	0.470
	5 [135.83]	0.015	0.025	0.193	0.334	1.000	0.249
	6 [179.22]	0.074	0.346	0.132	0.470	0.249	1.000

Table 4.7: Auto MAC of LMS experimental mode-shapes in water, expanded on mode-shapes of 2D model in strain

vs.		LMS experiments in water, expanded on 2D					
	nr.	1	2	3	4	5	6
	[freq.]	[54.40]	[55.37]	[99.03]	[130.35]	[135.83]	[179.22]
LMS experiments in water, expanded on 2D	1 [54.40]	1.000	0.016	0.016	0.015	0.018	0.061
	2 [55.37]	0.016	1.000	0.001	0.045	0.024	0.310
	3 [99.03]	0.016	0.001	1.000	0.035	0.163	0.110
	4 [130.35]	0.015	0.045	0.035	1.000	0.345	0.479
	5 [135.83]	0.018	0.024	0.163	0.345	1.000	0.239
	6 [179.22]	0.061	0.310	0.110	0.479	0.239	1.000

Table 4.8: Auto MAC of LMS experimental mode-shapes in air, expanded on earlier expanded PAK experimental mode-shapes on 3D model in strain

vs.		LMS in air, expanded on earlier expanded PAK on 3D model							
	nr.	1	2	3	4	5	6	7	8
	[freq.]	[70.19]	[70.19]	[111.89]	[111.89]	[154.70]	[154.70]	[202.60]	[202.60]
LMS in air, expanded on earlier expanded PAK on 3D model	1 [70.19]	1.000	0.000	0.011	0.000	0.000	0.000	0.096	0.000
	2 [70.19]	0.000	1.000	0.000	0.011	0.000	0.000	0.000	0.096
	3 [111.89]	0.011	0.000	1.000	0.000	0.007	0.000	0.050	0.000
	4 [111.89]	0.000	0.011	0.000	1.000	0.000	0.007	0.000	0.050
	5 [154.70]	0.000	0.000	0.007	0.000	1.000	0.000	0.081	0.000
	6 [154.70]	0.000	0.000	0.000	0.007	0.000	1.000	0.000	0.081
	7 [202.60]	0.096	0.000	0.050	0.000	0.081	0.000	1.000	0.000
	8 [202.60]	0.000	0.096	0.000	0.050	0.000	0.081	0.000	1.000

culated from the expanded accelerometer ($^{\mathrm{a}}$) experimental mode-shapes in displacement. The factors for the expansion are then obtained by minimizing equation 4.8

$$\min \left\| \sum_{i=1}^{m} \mathbf{C}^{\mathrm{s}} \boldsymbol{\psi}_{(i)}^{\mathrm{et,a}} d_{ij} - \boldsymbol{\psi}_{(j)}^{\mathrm{ex,s}} \right\|^2 \quad \text{for } j = 1, 2, 3, \ldots, M \tag{4.8}$$

and so the expansion becomes

$$\boldsymbol{\phi}_{j}^{\mathrm{et,s}} = \sum_{i=1}^{m} \boldsymbol{\phi}_{i}^{\mathrm{et,a}} d_{ij} \tag{4.9}$$

In matrix notation $\mathbf{D}$ becomes

$$\mathbf{D} = \left[\mathbf{C}^{\mathrm{s}} \boldsymbol{\Psi}^{\mathrm{et,a}} \right]^{+} \boldsymbol{\Psi}^{\mathrm{ex,s}} \tag{4.10}$$

so, the expanded experimental eigen mode-shapes become

$$\boldsymbol{\Phi}^{\mathrm{et,s}} = \boldsymbol{\Phi}^{\mathrm{et,a}} \mathbf{D}. \tag{4.11}$$

Note that the final expanded mode-shape space is in displacement, while the factor matrix is obtained in strain. This should not make a difference because the participation of a specific mode-shape is the the same independent of how it is obtained.
From equation 4.11 it is now clear that if the experiments gave the same mode-shapes and the experimental mode-shapes are mass normalised, $\mathbf{D}$ should be the unity matrix. It is in fact a measurement for how well the accelerometer and strain-gauge measurement are similar, and give similar expansions. The matrix $\mathbf{D}$ obtained for the original experiments was the incentive to doubt the mass normalisation of the original measurements. The matrix $\mathbf{D}$ for the 'hand'-normalised experimental data is shown in equation 4.12.

$$\mathbf{D} = \begin{bmatrix} 0.983 & 0.000 & 0.009 & 0.000 & -0.124 & 0.000 & -0.692 & 0.000 \\ 0.000 & 0.993 & 0.000 & 0.060 & 0.000 & 0.055 & 0.000 & -0.060 \\ -0.003 & 0.000 & -0.970 & 0.000 & -0.095 & 0.000 & -0.548 & 0.000 \\ 0.000 & 0.011 & 0.000 & -0.899 & 0.000 & 0.154 & 0.000 & 0.328 \\ 0.041 & 0.000 & -0.062 & 0.000 & -0.717 & 0.000 & 0.323 & 0.000 \\ 0.000 & 0.049 & 0.000 & -0.021 & 0.000 & -0.572 & 0.000 & 0.833 \\ -0.158 & 0.000 & -0.234 & 0.000 & -0.695 & 0.000 & -0.386 & 0.000 \\ 0.000 & -0.119 & 0.000 & -0.038 & 0.000 & -0.009 & 0.000 & 2.030 \end{bmatrix} \tag{4.12}$$

Participation factor matrix $\mathbf{D}$ shows that the first mode-shapes obtained by the LMS-system in air are quite similar to the ones obtained with the PAK-system in air, the higher modes show more and more distortion, this is probably due to the fact that higher mode have a lower amplitude and more curvature, and are therefore harder to measure especially with only eight strain-gauges.
Substitution of the equations 4.7 in equation 4.11 and the fact that expansions done in strain are similar to expansions done in displacement, gives

$$\boldsymbol{\Phi}^{\mathrm{num}} \mathbf{G} = \boldsymbol{\Phi}^{\mathrm{num}} \mathbf{A} \mathbf{D} \tag{4.13}$$

It is confirmed that for both the 'hand'-normalised and not-normalised, original experimental mode-shapes this equation holds. If an element-wise division is made between the matrices $\mathbf{G}$ and the result of $\mathbf{AD}$, a matrix filled with ones, is obtained.

$$\mathbf{G} \div \mathbf{AD} = \mathbf{1} \tag{4.14}$$

4.3 Static-shape as Space

As described in the introduction of this chapter, static shapes can also be used as a expansion space. Static spaces are build from shapes of the tube for static loads and are therefore called static-shapes. A number of shapes is obtained by applying boundary conditions of zero displacement, speed and acceleration to specific points and then, for each shape, changing one of the specific point displacements to unity. The obtained static-shapes form a static-shaped space, of which two will be discussed. First the static shapes where the boundary conditions are applied to the positions of the springs. And second, where they are applied to the accelerometer measurement device positions. Testing with static-shapes on the strain gauges positions is left out of this research, because, it is not possible to convert displacement static-shapes into strain static-shapes and therefore an expansion of the strain gauge data is not possible in Code_Aster for now.

4.3.1 Spring-position Static-shapes

The five springs from which the tube is hung, give five static-shapes. Using this static space to expand the experiments obtained using the accelerometers, the expansion shows their auto-MAC-numbers in table 4.9 for the static-shapes of the 3D model and in table 4.10 for the 2D model.

Table 4.9: Auto MAC of PAK experimental mode-shapes in air, expanded on static-shapes of 3D model on spring positions in displacement

vs.		PAK experiments, expanded on static-shapes on spring pos.							
	nr.	1	2	3	4	5	6	7	8
	nr. [freq.]	[70.26]	[70.26]	[111.89]	[111.89]	[154.69]	[154.69]	[202.61]	[202.61]
PAK experiments, expanded on static-shapes on spring pos.	1 [70.26]	1.000	0.010	0.080	0.001	0.254	0.003	0.003	0.002
	2 [70.26]	0.010	1.000	0.000	0.095	0.012	0.250	0.004	0.028
	3 [111.89]	0.080	0.000	1.000	0.002	0.044	0.000	0.000	0.001
	4 [111.89]	0.001	0.095	0.002	1.000	0.000	0.048	0.005	0.003
	5 [154.69]	0.254	0.012	0.044	0.000	1.000	0.042	0.160	0.037
	6 [154.69]	0.003	0.250	0.000	0.048	0.042	1.000	0.026	0.071
	7 [202.61]	0.003	0.004	0.000	0.005	0.160	0.026	1.000	0.176
	8 [202.61]	0.002	0.028	0.001	0.003	0.037	0.071	0.176	1.000

4.3.2 Sensor-position Static-shapes

When making a space containing the static-shapes based on the accelerometer positions, the space becomes much more refined with respect to the one based on the spring positions. The auto-MAC-table based on this two times fourteen static-shaped space is in table 4.11 for the 3D model and in table 4.12 for the 2D model.

Table 4.10: Auto MAC of PAK experimental mode-shapes in air, expanded on static-shapes of 2D model on spring positions in displacement

vs.		PAK experiments, expanded on static-shapes on spring pos.							
	nr.	1	2	3	4	5	6	7	8
	nr. [freq.]	[70.26]	[70.26]	[111.89]	[111.89]	[154.69]	[154.69]	[202.61]	[202.61]
PAK experiments, expanded on static-shapes on spring pos.	1 [70.26]	1.000	0.011	0.050	0.001	0.256	0.003	0.000	0.003
	2 [70.26]	0.011	1.000	0.000	0.075	0.015	0.248	0.006	0.026
	3 [111.89]	0.050	0.000	1.000	0.001	0.053	0.000	0.001	0.002
	4 [111.89]	0.001	0.075	0.001	1.000	0.000	0.051	0.006	0.006
	5 [154.69]	0.256	0.015	0.053	0.000	1.000	0.050	0.187	0.053
	6 [154.69]	0.003	0.248	0.000	0.051	0.050	1.000	0.032	0.073
	7 [202.61]	0.000	0.006	0.001	0.006	0.187	0.032	1.000	0.223
	8 [202.61]	0.003	0.026	0.002	0.006	0.053	0.073	0.223	1.000

Table 4.11: Auto MAC of PAK experimental mode-shapes in air, expanded on static-shapes of 3D model on accelerometer positions in displacement

vs.		PAK experiments, expanded on static-shapes on accel. pos.							
	nr.	1	2	3	4	5	6	7	8
	nr. [freq.]	[70.26]	[70.26]	[111.89]	[111.89]	[154.69]	[154.69]	[202.61]	[202.61]
PAK experiments, expanded on static-shapes on accel. pos.	1 [70.26]	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	2 [70.26]	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000
	3 [111.89]	0.000	0.000	1.000	0.000	0.005	0.000	0.000	0.000
	4 [111.89]	0.000	0.000	0.000	1.000	0.000	0.004	0.000	0.000
	5 [154.69]	0.000	0.000	0.005	0.000	1.000	0.000	0.000	0.000
	6 [154.69]	0.000	0.000	0.000	0.004	0.000	1.000	0.000	0.001
	7 [202.61]	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000
	8 [202.61]	0.000	0.000	0.000	0.000	0.000	0.001	0.000	1.000

4.3.3 Comparing Spring- vs. Sensor-position Static-shapes

Obviously, both figure 4.1, as well as both MAC-tables 4.11 and 4.12, show that expansion on the sensor-position static-shapes give a better result than on the spring position shapes. First and most important reason is the number of participating shapes. In one direction, the space containing the sensor based static-shapes has fourteen shapes and the space with the spring based shapes has only five. The second reason is from a more physical point of view, the tube vibrates more where it is not restrained by the springs. By using the spring positions as points to impose a unit displacement for the different shapes will give shapes which are not likely to fit the measurement data. So, if static-shape are to be used, the ones based on the accelerometer sensor positions should be used.

Table 4.12: Auto MAC of PAK experimental mode-shapes in air, expanded on static-shapes of 2D model on accelerometer positions in displacement

vs.		PAK experiments, expanded on static-shapes on accel. pos.							
	nr. [freq.]	1 [70.26]	2 [70.26]	3 [111.89]	4 [111.89]	5 [154.69]	6 [154.69]	7 [202.61]	8 [202.61]
PAK experiments, expanded on static-shapes on accel. pos.	1 [70.26]	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	2 [70.26]	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000
	3 [111.89]	0.000	0.000	1.000	0.000	0.005	0.000	0.000	0.000
	4 [111.89]	0.000	0.000	0.000	1.000	0.000	0.004	0.000	0.000
	5 [154.69]	0.000	0.000	0.005	0.000	1.000	0.000	0.000	0.000
	6 [154.69]	0.000	0.000	0.000	0.004	0.000	1.000	0.000	0.001
	7 [202.61]	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000
	8 [202.61]	0.000	0.000	0.000	0.000	0.000	0.001	0.000	1.000

4.4 Comparing Eigen Mode-shapes vs. Static-shapes

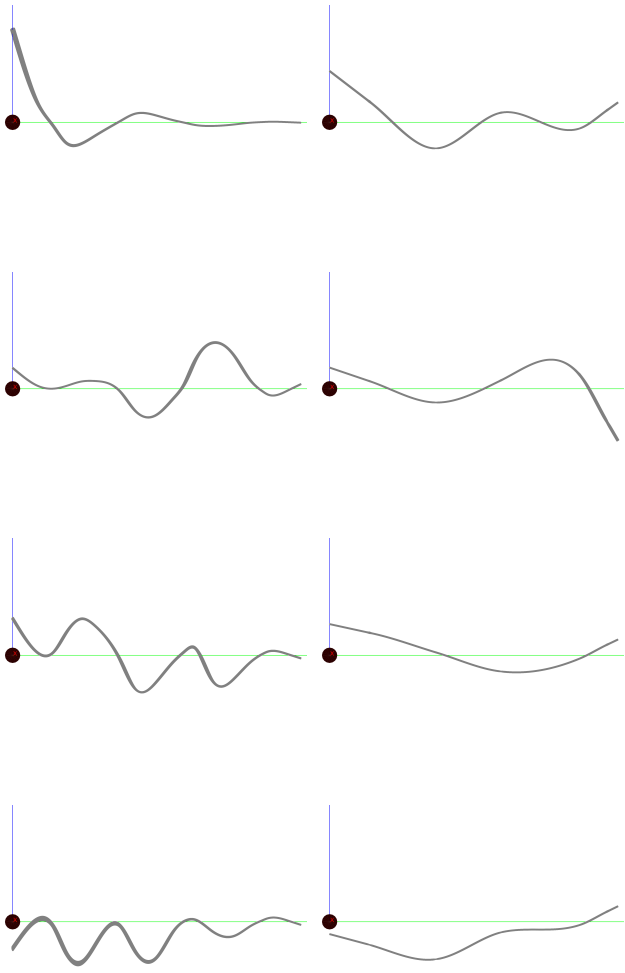
The experiments expanded on the static-shapes on the accelerometer measurement device positions show a similar well orthogonal behaviour as the experiments expanded on the eigen mode-shapes of the numerical model. Figure 4.2 shows this similarity. Fact is, the measurements in water are only done with the strain gauges and these strain measurements cannot be expanded on static-shapes, because the static shapes cannot be calculated in strain. Therefore, at this moment, the static shapes cannot be used.

4.5 Comparing the 2D Shell-element model vs. the 3D Volume-element model

In chapter 3, no final conclusion was made on which of the two numerical models to use. As shown in the MAC-tables in section 4.2.1, there is not much difference between the performance of the 3D volume-element model and the 2D-shell element model. Considering the arguments in the conclusion of chapter 3, the 3D volume-element model is selected for further studies.

4.6 Comparing Accelerometer vs. Strain Gauge Measurements

To see if the stain gauge based measurements are as good as the accelerometer based measurements a MAC-table can be made between the different expanded measurements. Both mode-shapes must be in either strain or in displacement. So by now computing the strain mode-shapes in Code_Aster for the expanded accelerometer measurements, the mode-shapes of this PAK system can be compared to the LMS strain gauge obtained mode-shapes. The MAC-numbers between them are shown in table 4.13.

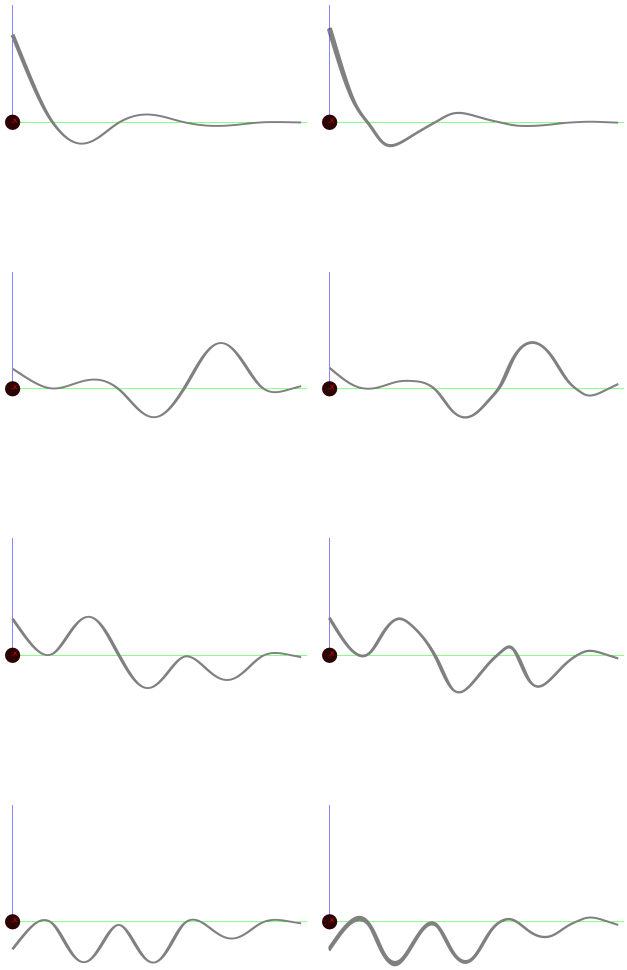


(a) (b)

Figure 4.1: PAK-experimental mode-shapes expanded on static-shapes, (a) based on accelerometer sensor positions, and (b) based on spring positions.

And, the other way around, putting the expanded LMS-measured mode-shapes in displacement, and comparing them to the PAK-measurements gives the MAC-table in table 4.14.

The MAC-tables show the similarity between the two different measurements as well as the visuals of the mode-shapes in figure 4.3. In this figure is also shown that the mode-shape 2 and 3 are flipped over the y-axis, which explains the minus in **D** for the corresponding row and column. The extreme displacement in the last mode-shape of the LMS-measurement is probably due to a measurement error of one specific sensor, or



(a) (b)

Figure 4.2: PAK-experimental mode-shapes expanded on, (a) eigen mode-shapes, and (b) on static-shapes, based on accelerometer sensor positions.

because of wrong calibration, of that sensor. For this reason, and because the mode-shapes in air are obtained by the PAK-system using 14 accelerometers and by the LMS-system by just 8 in one direction, the PAK-system is chosen to obtain the mode-shapes in air.

Table 4.13: Cross MAC of PAK experimental mode-shapes in air, expanded on mode-shapes of 3D model vs LMS experimental mode-shapes in air, expanded on mode-shapes of 3D model in strain

vs.		LMS experiments in air, expanded on 3D model							
	nr. [freq.]	1 [70.19]	2 [70.19]	3 [111.89]	4 [111.89]	5 [154.70]	6 [154.70]	7 [202.60]	8 [202.60]
PAK experiments in air, expanded on 3D model	1 [70.26]	0.847	0.000	0.010	0.000	0.019	0.000	0.000	0.000
	2 [70.26]	0.000	0.847	0.000	0.010	0.000	0.019	0.000	0.000
	3 [111.89]	0.003	0.000	0.992	0.000	0.015	0.000	0.021	0.000
	4 [111.89]	0.000	0.003	0.000	0.992	0.000	0.015	0.000	0.021
	5 [154.69]	0.000	0.000	0.011	0.000	0.961	0.000	0.089	0.000
	6 [154.69]	0.000	0.000	0.000	0.011	0.000	0.961	0.000	0.089
	7 [202.61]	0.102	0.000	0.012	0.000	0.000	0.000	0.899	0.000
	8 [202.61]	0.000	0.102	0.000	0.012	0.000	0.000	0.000	0.899

Table 4.14: Cross MAC of PAK experimental mode-shapes in air, expanded on mode-shapes of 3D model vs LMS experimental mode-shapes in air, expanded on mode-shapes of 3D model in displacement

vs.		LMS experiments in air, expanded on 3D model							
	nr. [freq.]	1 [70.19]	2 [70.19]	3 [111.89]	4 [111.89]	5 [154.70]	6 [154.70]	7 [202.60]	8 [202.60]
PAK experiments in air, expanded on of 3D model	1 [70.26]	0.984	0.000	0.002	0.000	0.005	0.000	0.002	0.000
	2 [70.26]	0.000	0.984	0.000	0.002	0.000	0.005	0.000	0.002
	3 [111.89]	0.001	0.000	0.993	0.000	0.039	0.000	0.026	0.000
	4 [111.89]	0.000	0.001	0.000	0.993	0.000	0.039	0.000	0.026
	5 [154.69]	0.008	0.000	0.008	0.000	0.920	0.000	0.128	0.000
	6 [154.69]	0.000	0.008	0.000	0.008	0.000	0.920	0.000	0.128
	7 [202.61]	0.023	0.000	0.001	0.000	0.000	0.000	0.826	0.000
	8 [202.61]	0.000	0.023	0.000	0.001	0.000	0.000	0.000	0.826

4.7 Comparing the Mode-shapes in Air vs. the Mode-shapes in Water

For the mass normalised mode-shapes in air to be a good reference to the not-normalised mode-shapes in water, when trying to calculate the modal mass in water, it is important that the mode-shapes in air and in water are quite similar. Now, after the expansions, the mode-shapes can be compared. The MAC-numbers between the mode-shapes in air obtained by the PAK-system compared and the mode-shapes in water by the LMS-system are shown in MAC-table 4.15.

And the MAC-table between the mode-shapes in air obtained by the LMS-system versus the mode-shapes obtained in water is in table 4.16. The first mode-shape-pair in

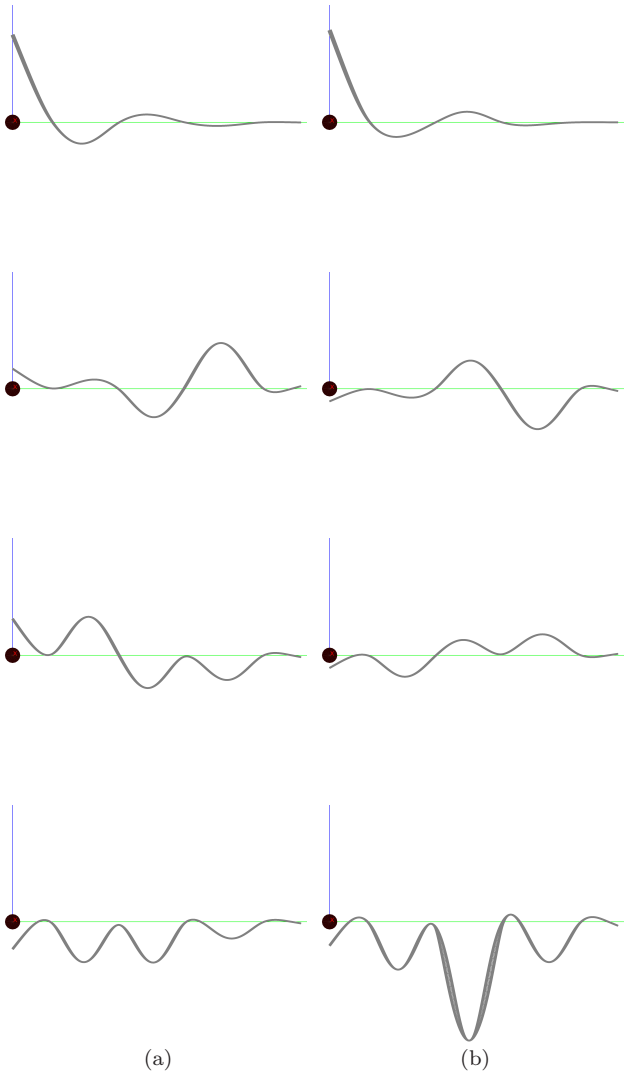


Figure 4.3: (a) PAK-experimental mode-shapes expanded on eigen mode-shapes, (b) LMS-experimental mode-shapes expanded on eigen mode-shapes.

fluid corresponding to a frequency of 55 Hz, is really well covered by the first mode-shape pair of the model experiments in air. The second mode-shape in this pair less than the first, especially for the LMS-measurements in table 4.16.
The second mode-shape in water is not described by a pair, but just as a single shape, it shows great similarity to the first of the mode-shape-pair corresponding to 112 Hz of the measurements in air. The second shape of the pair is apparently on a 90° angle over the y-axis, with respect to the single shape in fluid.
The third mode-shape-pair in fluid shows a good correspondence with the first of its coun-

Table 4.15: Cross MAC of LMS experimental mode-shapes in water, expanded on mode-shapes of 3D model vs PAK experimental mode-shapes in air, expanded on mode-shapes of 3D model in displacement

vs.			PAK experiments in air, expanded on 3D model							
		nr.	1	2	3	4	5	6	7	8
	nr.	[freq.]	[70.26]	[70.26]	[111.89]	[111.89]	[154.69]	[154.69]	[202.61]	[202.61]
LMS experiments in water, expanded on 3D	1	[54.40]	0.780	0.202	0.002	0.004	0.006	0.001	0.010	0.005
	2	[55.37]	0.226	0.695	0.001	0.023	0.002	0.008	0.002	0.041
	3	[99.03]	0.009	0.012	0.929	0.023	0.002	0.020	0.000	0.002
	4	[130.35]	0.022	0.229	0.011	0.080	0.181	0.397	0.008	0.064
	5	[135.83]	0.021	0.113	0.001	0.012	0.755	0.075	0.000	0.000
	6	[179.22]	0.002	0.041	0.049	0.008	0.044	0.013	0.780	0.047

Table 4.16: Cross MAC of LMS experimental mode-shapes in water, expanded on mode-shapes of 3D model vs LMS experimental mode-shapes in air, expanded on mode-shapes of 3D model in strain

vs.			LMS experiments in air, expanded on 3D model							
		nr.	1	2	3	4	5	6	7	8
	nr.	[freq.]	[70.19]	[70.19]	[111.89]	[111.89]	[154.70]	[154.70]	[202.60]	[202.60]
LMS experiments in water, expanded on 3D	1	[54.40]	0.706	0.091	0.003	0.007	0.000	0.002	0.027	0.084
	2	[55.37]	0.132	0.156	0.000	0.044	0.000	0.000	0.003	0.413
	3	[99.03]	0.012	0.008	0.920	0.032	0.017	0.029	0.022	0.025
	4	[130.35]	0.001	0.098	0.011	0.114	0.204	0.466	0.001	0.343
	5	[135.83]	0.004	0.019	0.001	0.015	0.858	0.089	0.094	0.014
	6	[179.22]	0.081	0.000	0.062	0.006	0.025	0.005	0.914	0.066

terpart pair in air, but it shows not a good correspondence with the second mode-shape in that pair in air.it is a lot less clear.

The final mode-shape in fluid is, again, single and shows a quite good similarity to the first shape of its corresponding pair in air.
In general, the measurements in air made by the PAK-system are showing a somewhat better correspondence to the mode-shapes in water, than the ones obtained by the LMS-system. This can be due to the fact that the measurements using the PAK-system are obtained with fourteen sensors while LMS has only eight strain gauges in one direction, to obtain the mode-shape. Therefore, the mode-shapes obtained with the PAK-system are used in further calculations. In figure 4.4 to 4.9, the mode-shapes in water can be compared to the mode-shapes obtained in air using the PAK-system and the accelerometers.

4.8 Comments, Conclusions & Recommendations

4.8.1 Comments

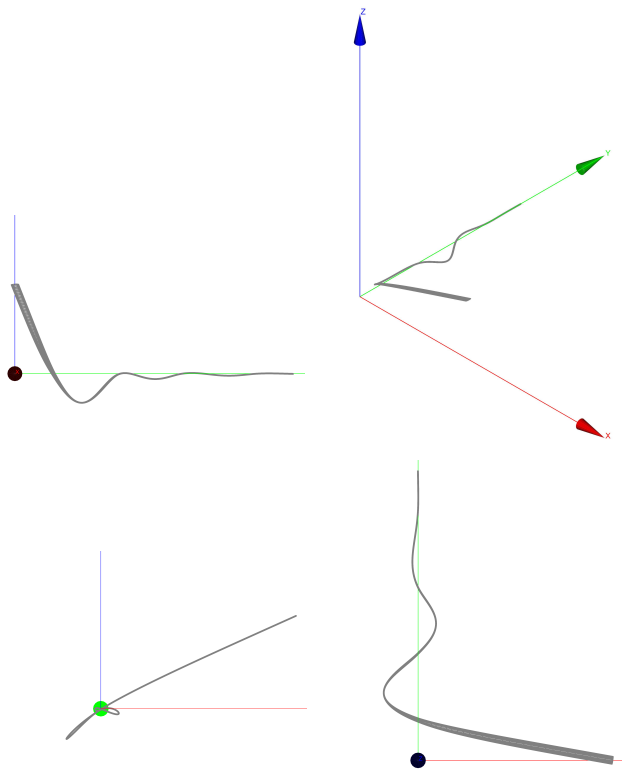
For clarification, it should be better to show the MAC tables for the mode-shapes in just one direction at the time, but because this whole study is in two directions simultaneously and these expanded mode-shapes for the two directions together are needed in further in analysis, the MAC tables are also shown for the full set.
The idea to expanded both the LMS-data and the PAK-data at the same time, to be able to get an even better expansion is not possible in Code_Aster, and is therefore not done.
The Code_Aster function `PROJ_MESU_MODAL` should be improved to be able to handle strain and displacement measurements simultaneously.

4.8.2 Conclusions

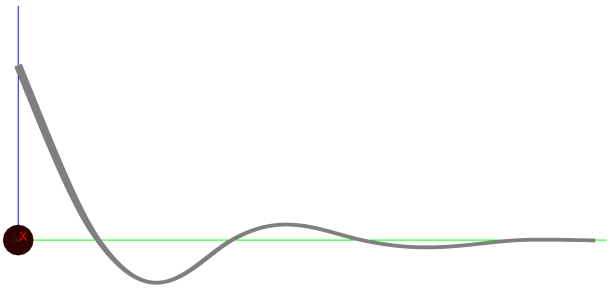
The final expansion, which can be used during the further study will be done on numerical eigen mode-shapes of the 3D volume-element model.
For air, the PAK-data will be used and in water of course the LMS-data.
By doing the study simultaneously in two directions the curvature over the y-axis for the mode-shapes in water can be included in the analysis.

4.8.3 Recommendations

The use of static shapes has been evicted because it is still not possible to operate strain static shapes in Code_Aster. It is therefore strongly recommended to improve Code_Aster to be able to handle strain static-shapes.
Next to that, a two step expansion could maybe improve the expansion considerable. By first expanding the experimentally obtained mode-shapes on the numerical mode-shapes in the same frequency range, and afterwards using static shapes to make adjustments on the expanded mode-shapes using the remaining difference at the sensor positions, the experimentally obtained eigen mode-shapes can maybe be better expanded.

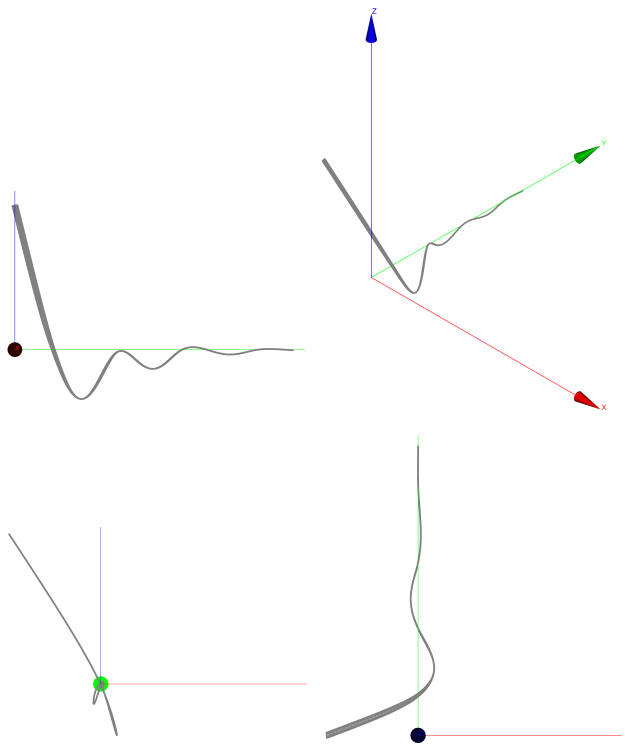


(a) Mode-shape 1, obtained by LMS in water

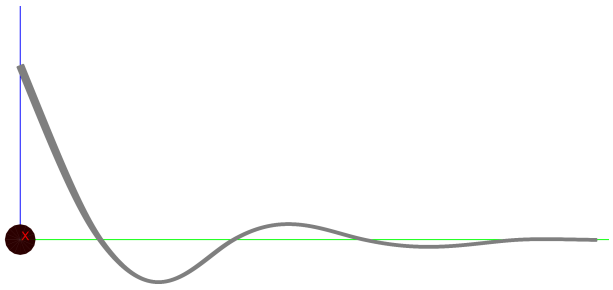


(b) Corresponding Mode-shape obtained by PAK in air

Figure 4.4: Mode-shape 1, obtained for the system in water and the corresponding mode-shape in air, both expanded on the 3D model.

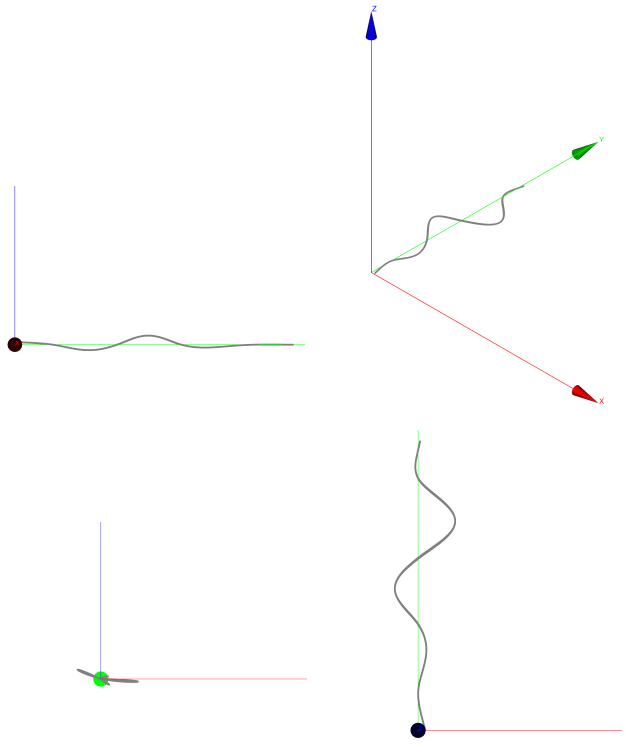


(a) Mode-shape 2, obtained by LMS in water

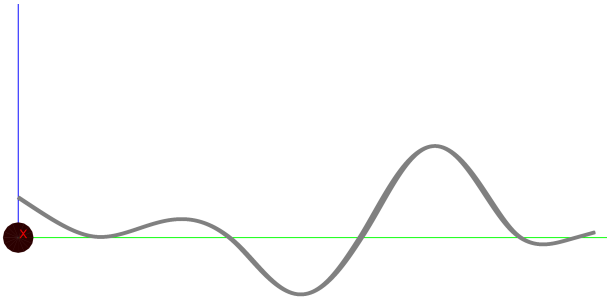


(b) Corresponding Mode-shape obtained by PAK in air

Figure 4.5: Mode-shape 2, obtained for the system in water and the corresponding mode-shape in air, both expanded on the 3D model.

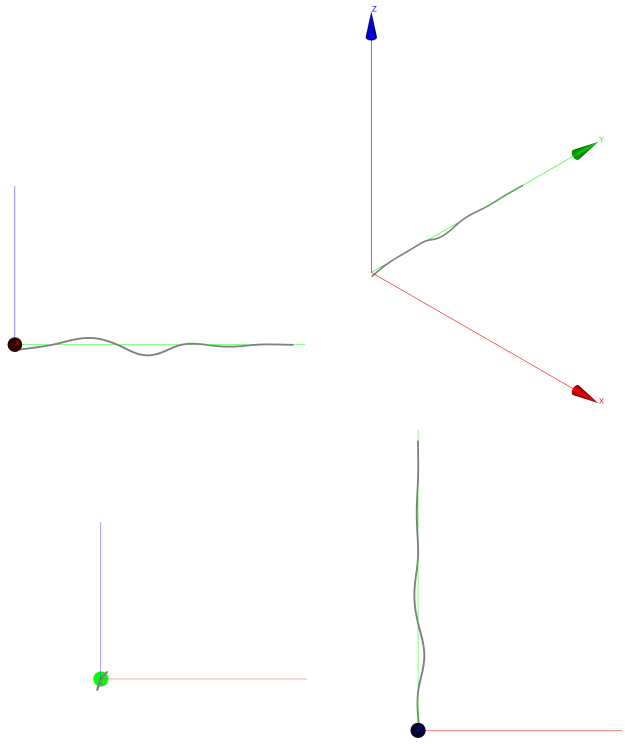


(a) Mode-shape 3, obtained by LMS in water

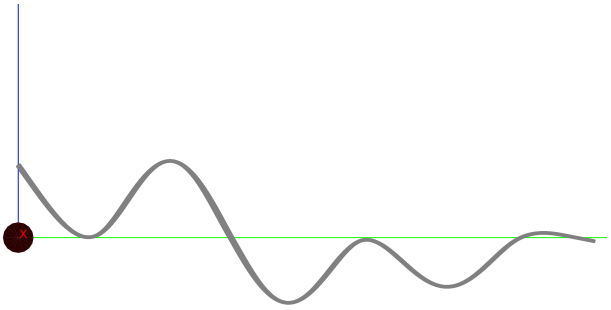


(b) Corresponding Mode-shape obtained by PAK in air

Figure 4.6: Mode-shape 3, obtained for the system in water and the corresponding mode-shape in air, both expanded on the 3D model.

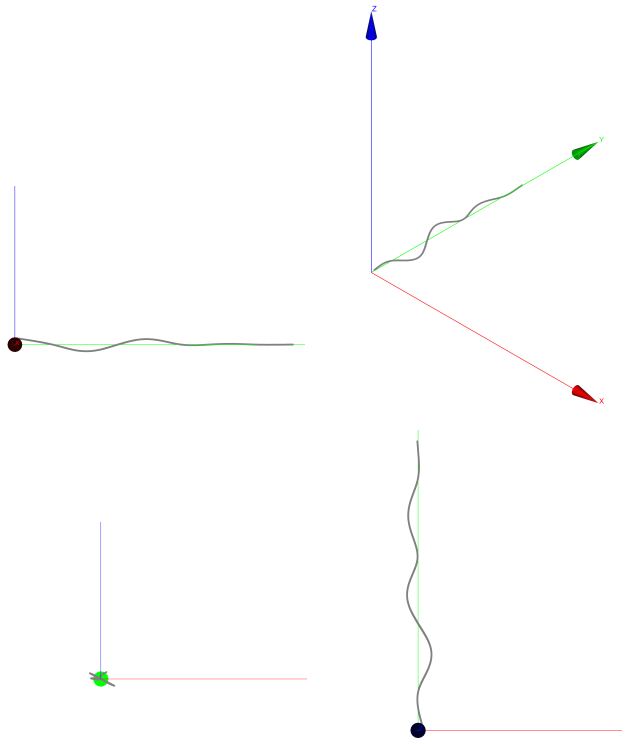


(a) Mode-shape 4, obtained by LMS in water

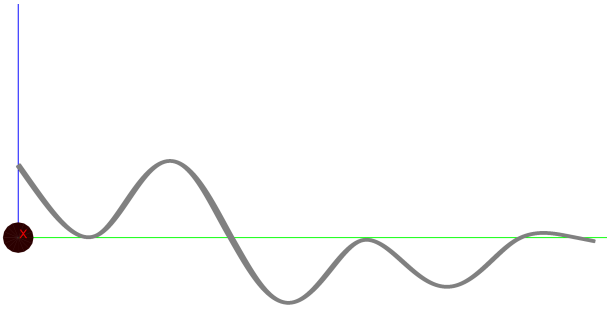


(b) Corresponding Mode-shape obtained by PAK in air

Figure 4.7: Mode-shape 4, obtained for the system in water and the corresponding mode-shape in air, both expanded on the 3D model.

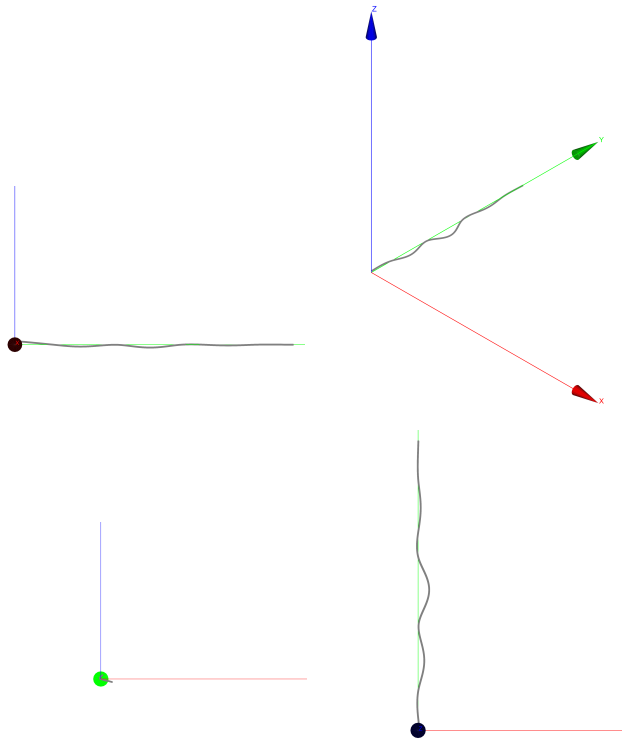


(a) Mode-shape 5, obtained by LMS in water

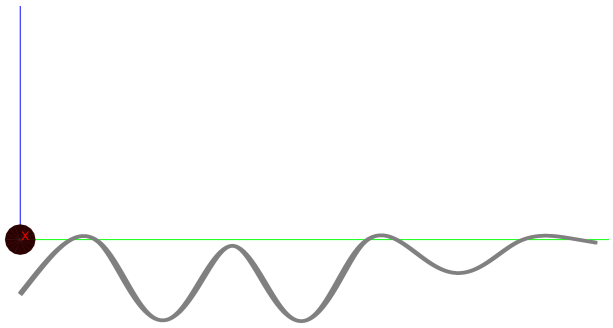


(b) Corresponding Mode-shape obtained by PAK in air

Figure 4.8: Mode-shape 5, obtained for the system in water and the corresponding mode-shape in air, both expanded on the 3D model.



(a) Mode-shape 6, obtained by LMS in water



(b) Corresponding Mode-shape obtained by PAK in air

Figure 4.9: Mode-shape 6, obtained for the system in water and the corresponding mode-shape in air, both expanded on the 3D model.

Part II

Mass Normalising the Mode-shapes

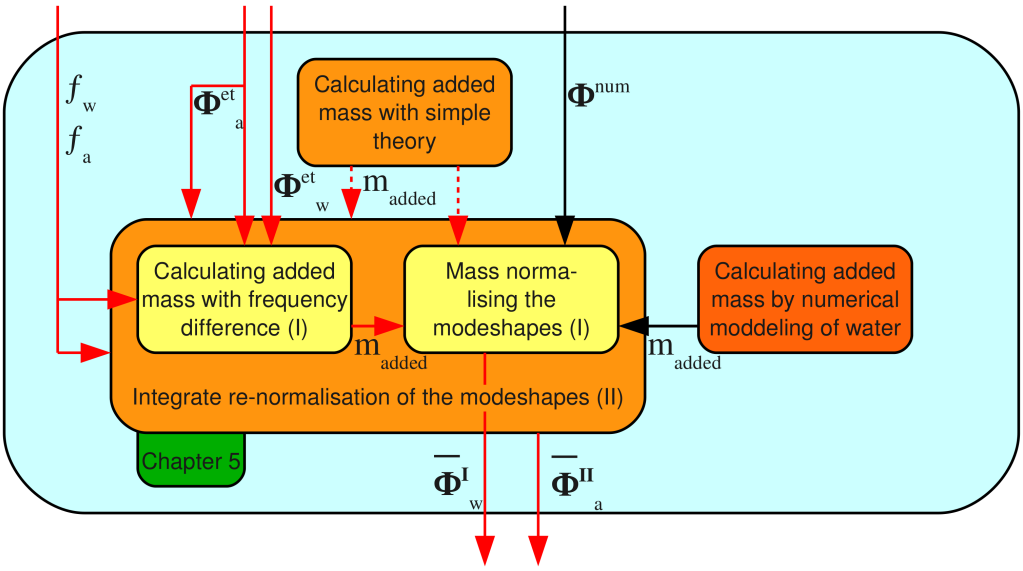


Figure 4.10: Scheme of the field of possibilities for the second part of the study. In red, the chosen path, the dotted line marks the simple theory to test the chosen sub-methods I and II.

In this part, two sub-methods to mass normalise the mode-shapes using the frequency difference, are discussed. These mode-shapes can be used thereafter in the identification process. The mode-shapes used in the identification process need to be mass normalised because of reasons explained in appendix A.2. To mass normalise the mode-shapes the added mass due to the water surrounding the tube needs to be accounted for. There are different methods to calculate this added mass. In figure 4.10 there are three different possibilities shown; (1) using the frequency difference between the corresponding mode-shapes in water and in air, to mass normalise mode-shapes in water, (2) by calculating the added mass using the simple theory that says that the added mass is equal to the mass of the water displaced, and (3) by modelling the water in a numerical model, the added mass can be approximated.

The first method, method (1), is used during this study. It is depicted as the red line in figure 4.10. It consists of two similar sub-methods. The original method, sub-method I, also used by Bodel, is extended by a second method, sub-method II. The original sub-method calculates the added mass using the modal mass in air, the frequency difference and the equivalent length of the mode-shapes in air and water. By integrating the calculation of the added mass and the normalisation, the new sub-method can be obtained which scales the mode-shapes in air to be used as mass normalised mode-shapes in water, using just the frequency difference. The original and the new, sub-method I and II respectively, will be discussed in this part.

The simple theory, method (2), will be used to check the order of the calculated added mass obtained by the old and the new sub-method (I and II) of method (1). This is depicted in figure 4.10 as the red dotted line. The simple theory states that the added mass is equal to the mass of the displaced water by the tube. Thus for the tube the added mass should

be around

$$m_{added} = \rho_w \pi R^2 L = 0,170[\text{kg}]$$

where ρ_w *is the density of the water,* R *is the outer radius of the tube and* L *is the length of the tube. Compared to a mass of the tube of*

$$m_{tube} = \rho_s \pi \left(R^2 - r^2\right) L = 0,479[\text{kg}]$$

where ρ_s *is the density of steal and* r *is the inner tube radius, the added mass is 35.4%. Another option to calculate the added mass is method (3), which models the water surrounding the tube by a FEM. This method is left for further studies.*

Chapter 5

Mass-Normalising the Mode-shapes in Water

5.1 Introduction

This chapter consists of two methods. First, the original method, sub-method I, from now on called just method I, which is the method used by Bodel [1] and which normalises the mode-shapes in water using the mass normalised mode-shapes in air as a reference. Second, the new method, sub-method II, from now on called just method II, for which the process to obtain the added mass and the normalisation are integrated, making it possible to use the mode-shape in air. The experimental mode-shapes in air can be normalised because they are obtained using hammer excitation and thus the input signal can be measured. Via this hypotheses the mode-shapes in water can be referenced to the ones in air.
Both methods rely on the hypothesis that the mode-shapes are the same for the system in water as in air.
This chapter first introduces the hypothesis on which both methods are based. Then, in section 5.3, the old method, sub-method I is explained. In section 5.4 the new method, sub-method II is explained. Both methods are compared with each other and with the simple theory in section 5.5. This chapter ends with comments, a conclusion and recommendations.

5.2 Hypothesis

These methods is based on the Reynolds quotient, stating that the natural frequency of a eigen mode i is dependent on the modal mass and stiffness corresponding to that mode i.

$$\omega_i^2 = \frac{\kappa_i}{\mu_i} = \frac{EI \int_L \left[\frac{\partial^2 \phi}{\partial y^2}\right]^2 \mathrm{d}y}{\rho A \int_L \phi_i^2 \, \mathrm{d}y} \tag{5.1}$$

where κ_i is the modal stiffness of mode i and μ_i is its model mass. When the tube is submerged the system will differ from the situation in air. This difference is represented

by the added modal mass, due to the water surrounding the tube, in the denominator of the equation. The numerator, representing the stiffness of the tube, does not significantly change by the water surrounding the tube. Because, when the tube is given a constant displacement it is not opposed by the water surrounding the tube[10]. The only non-zero stiffness, added by the fluid, is to volume deformation, because a fluid does not sustain shear forces. When the tube moves, the pressure builds up, but flows away with the speed of sound. Thus, when the speed of the moving tube is not in the order of the speed of pressure wave propagation, the speed of sound, the build up pressure gets away and cannot impose any opposition to the movement. The maximum speed of the tube is in the order of, taking an excessive amplitude $X_{\max} = 0.01$ m and for the maximum frequency $f_{\max} = 1000$ Hz

$$\dot{X}_{\max} = 2\pi f_{\max} X_{\max} = 62.83[\text{m/s}] \tag{5.2}$$

The speed of sound is 1484 m/s in water of 20 °C. The relative speed of the pressure wave propagating away from the pressure built up is 1421 m/s, the added stiffness is thus negligible. Now, by making the assumption that, despite the change in modal mass, the mode-shapes remain the same in water as in air. Thus a hypothesis can be stated as

$$\phi_{\text{w}i} = C_i \phi_{\text{a}i}. \tag{5.3}$$

This assumption was verified in chapter 4 by visual conformation, in figure 4.4 to 4.9, and by the MAC-tables 4.15 and 4.16, between the mode-shapes in air and in water.

5.3 Method I: Calculating the Added Mass and Normalising the Mode-shapes in Water

The hypothesis directly shows that the mode-shapes in strain, for the system in air and in water, have the same ratio C_i between them

$$\psi_{\text{w}i} = R\frac{\partial^2 \phi_{\text{w}i}}{\partial y^2} = C_i R\frac{\partial^2 \phi_{\text{a}i}}{\partial y^2} = C_i \psi_{\text{a}i}. \tag{5.4}$$

The Rayleigh equation for the system in air can be written as

$$\omega_{\text{a}i}^2 = \frac{\kappa_{\text{a}i}}{\mu_{\text{a}i}} = \frac{EI \int_L \left[\frac{\partial^2 \phi_{\text{a}i}}{\partial y^2}\right]^2 \text{d}y}{\rho_{\text{s}} A \int_L \phi_{\text{a}i}^2 \,\text{d}y} \tag{5.5}$$

and, because stiffness properties are the same for air and water, the equation for the system in water can be written as

$$\omega_{\text{w}i}^2 = \frac{\kappa_{\text{w}i}}{\mu_{\text{w}i}} = \frac{EI \int_L \left[\frac{\partial^2 \phi_{\text{w}i}}{\partial y^2}\right]^2 \text{d}y}{\mu_{\text{w}i}} = \frac{EI\, C_i^2 \int_L \left[\frac{\partial^2 \phi_{\text{a}i}}{\partial y^2}\right]^2 \text{d}y}{\mu_{\text{w}i}}. \tag{5.6}$$

Rewriting equation 5.5 gives

$$EI \int_L \left[\frac{\partial^2 \phi_{\text{a}i}}{\partial y^2}\right]^2 \text{d}y = \omega_{\text{a}i}^2 \mu_{\text{a}i} \tag{5.7}$$

and substituting it in 5.6 shows

$$\mu_{\mathrm{w}i} = \mu_{\mathrm{a}i} C_i^2 \frac{\omega_{\mathrm{a}i}^2}{\omega_{\mathrm{w}i}^2} \tag{5.8}$$

The constant C_i is correcting for the difference in amplitude between the experientially obtained mode-shapes in water and air. It is a correction to 'scale' the mode-shapes in a similar manner. This constant is therefore obtained for the equivalent length of a mode-shapes as explained in appendix B.2. The constant can be obtained by dividing the respective equivalent lengths

$$C_i^2 = \frac{L_{\mathrm{eq,w}i}}{L_{\mathrm{eq,a}i}} = \frac{\int_L \phi_{\mathrm{w}i}^2 \,\mathrm{d}y}{\int_L \phi_{\mathrm{a}i}^2 \,\mathrm{d}y}. \tag{5.9}$$

These equivalent lengths can be approximated using the trapezium-rule or frustrum-rule in `CALC_ESSAI` as explained in appendix B.2. Finally the modal mass in water can be described as

$$\mu_{\mathrm{w}i} = \mu_{\mathrm{a}i} \frac{\tilde{L}_{\mathrm{eq,w}i}}{\tilde{L}_{\mathrm{eq,a}i}} \frac{\omega_{\mathrm{a}i}^2}{\omega_{\mathrm{w}i}^2}. \tag{5.10}$$

When mass normalised mode-shapes in air are used, the modal masses are unity, $\mu_{\mathrm{a}i} = 1$ kg m^2. The calculated modal masses in water, which mass normalise the mode-shapes, are put in the data files obtained from the measurement systems. These data files can then be read by Code_Aster and the mass normalised mode-shapes in water can be used in further analyses.

Remark: *We are lucky that we need to foul Code_Aster to using the mode-shape strain data as data for displacement (see part 3), because when we use it as displacement data in Code_Aster , the mode-shapes can be normalised using NORM_MODE. This is not possible when the data is used as strain data.*

5.4 Method II: Re-normalising the Mode-shapes in Air to be used as Mode-shapes in Water

This integrated method re-uses the hypothesis, and uses the mode-shapes in air to model the system in water. The mode-shapes in air are therefore re-normalised using a method, which is an extension of the earlier described method I. By rewriting equation 5.10

$$\rho_i' A \int_L \phi_{\mathrm{w}i}^2 \,\mathrm{d}y = \rho_{\mathrm{s}} A \int_L \phi_{\mathrm{a}i}^2 \,\mathrm{d}y \frac{\tilde{L}_{\mathrm{eq,w}i}}{\tilde{L}_{\mathrm{eq,a}i}} \frac{\omega_{\mathrm{a}i}^2}{\omega_{\mathrm{w}i}^2} \tag{5.11}$$

where the added mass is included in the density ρ_i'. Then equation 5.9 can be used to reduce equation 5.11 to

$$\frac{\rho_i'}{\rho_{\mathrm{s}}} = \frac{\omega_{\mathrm{a}i}^2}{\omega_{\mathrm{w}i}^2} \tag{5.12}$$

The mass normalised mode-shapes in water should give

$$\mu_{\mathrm{w}i} = \rho_i' A \int_L \overline{\phi}_{\mathrm{w}i}^2 \,\mathrm{d}y = 1[\mathrm{kg\ m^2}] \tag{5.13}$$

By substituting 5.12 in 5.13 and using the hypothesis, equation 5.3 again

$$\mu_{\mathrm{w}i} = \rho_{\mathrm{s}} A \frac{\omega_{\mathrm{a}i}^2}{\omega_{\mathrm{w}i}^2} \int_L \overline{\phi}_{\mathrm{w}i}^2 \,\mathrm{d}y = \rho_{\mathrm{s}} A \frac{\omega_{\mathrm{a}i}^2}{\omega_{\mathrm{w}i}^2} \int_L C_{\mathrm{N}i}^2 \overline{\phi}_{\mathrm{a}i}^2 \,\mathrm{d}y = 1[\mathrm{kg\ m^2}] \tag{5.14}$$

and thus the factor, to re-normalise the mass normalised mode-shapes in air to be usable in water, is

$$C_{\mathrm{N}i} = \frac{\omega_{\mathrm{w}i}}{\omega_{\mathrm{a}i}}. \tag{5.15}$$

So, by re-normalising the mass normalised modes-shapes in air, using the factor $C_{\mathrm{N}i} = \frac{\omega_{\mathrm{w}i}}{\omega_{\mathrm{a}i}}$, the earlier mass normalised mode-shapes in air become usable as mode-shapes describing the modal system in water. This means in practice, that the modal mass in the data file for the mode-shapes in air needs to be changed from unity to a new modal mass to a 'modal mass' containing this new factor. Because mass normalisation is done by dividing the mode-shapes by the square root of the model mass, the factor needs to be inverted and squared, so the inserted 'modal mass' is $C_{\mathrm{N}i}^{-2} = \frac{\omega_{\mathrm{a}i}^2}{\omega_{\mathrm{w}i}^2}$.
Take note that ρ_i' is not simply the sum of the density of steal plus the density of water, considering the simple theory, because the surface area for steal is the cross sectional area, which is only the rim of the tube, and the area of the displaced water would be the total area included by the circumference of the tube. Thus

$$\rho_i' \neq \rho_{\mathrm{s}} + \rho_{\mathrm{w}}. \tag{5.16}$$

5.5 Comparing both Methods and Checking them by the Simple Theory

Both methods have their specific properties. The first method, method I, normalising the mode-shapes in water, has the following properties

- The modal mass is calculated using the ratio squared between the experimentally determined frequencies, the modal mass in air, and the mode-shapes in both water and air. This calculated modal mass is then inserted in the datafile.
- This method normalises the mode-shapes in water, which were obtained by the strain-gauges, and therefore have a less well discretization compared to the mode-shapes obtained with the accelerometers in air.
- Any 'twist' over the y-axis in the mode-shapes in water, as depicted in figure 4.4 through figure 4.9, is preserved when using these '3D' mode-shapes in water.
- The hypothesis is only used ones and only the ratio between the equivalent lengths is used to obtain the constant, C_i, between the mode-shapes in air and in water. The calculation of the equivalent lengths has an averaging character on the constant and the hypothesis is therefore used in a less stringent way than when it was necessary to state that the shapes are truly the same.
- The equivalent lengths need to be approximated by the trapezium rule, therefore introducing an approximation error.

The method that re-normalises the mode-shapes in air, method II, has the following properties

- This method is only applicable to mass normalised mode-shapes in air and uses the experimentally obtained frequency ratio squared directly to change the modal mass in the data file.
- This method has the advantage that only the mode-shapes in air are used and in water only the frequency needs to be determined. Therefore, the less discretized strain gauges are only used to obtain the frequency in water and the better discretized accelerometer measurements are used to obtain the mode-shapes.
- Because the mode-shapes in air are obtained in just one direction, the modal space built from these mode-shapes will not be able to describe the 'twisted' mode-shapes in water as depicted in figure 4.4 through 4.9.
- It has the disadvantage that the hypothesis is used twice and the second time in an more stringent way, stating that the mode-shapes are exactly the same, making the method much more dependent on the quality of the hypothesis.
- No need for approximation of the equivalent length.

Next, the simple theory is used to calculated a analytical mass normalisation criterion, similar to what is explained in appendix B.2. Both methods will be tested by calculating the equivalent length for both mass normalised mode-shapes and comparing them to the analytically obtained equivalent length from the simple theory

$$\mu_{\text{simple}i} = \frac{m_{\text{tube}} + m_{\text{added}}}{L} \int_L \overline{\phi}_i^2 \, \mathrm{d}y = 1[\text{kg m}^2] \tag{5.17}$$

thus

$$\int_L \overline{\phi}_i^2 \, \mathrm{d}y = \frac{1}{\rho_\text{s}\pi\left(R^2 - r^2\right) + \rho_\text{w}\pi R^2} = 3.345.[\text{m}^3] \tag{5.18}$$

Table 5.1 shows the approximated equivalent lengths for both methods. Both methods show mass normalised mode-shapes with a similar equivalent length and in the order of the value expected by the simple theory. The difference between the first pair of mode-shapes and the pairs corresponding to higher frequencies is probably due to the declination of the amplitude difference between the lower and higher mode-shapes, when comparing a system in air to a system in water. In other words, the difference between amplitudes of mode-shapes for a system in air is larger then for a system in water. This is also reflected by the decrease in relative frequency difference, from $\frac{55}{70}$ to $\frac{180}{203}$. Picturing equation 5.1, a similar influence is visual in the modal mass and thus in the equivalent length.

5.6 Comments, Recommendations & Conclusions

5.6.1 Comments

Both method give their result by the virtue of the 'hand'-normalised mode-shapes in air. The results are therefore artificial, but the results show that the methods comply with the expectancies. Therefore, when using experimental mode-shapes in air which are well mass normalised, the results should give similar equivalent length.

Table 5.1: Comparing the approximated equivalent lengths for both methods, the simple theory predicts and equivalent length of 3.34535 m^3. Method I is the original and is applied on the mode-shapes in water, Method II is the re-normalisation of the mode-shapes in air.

mode-shape pair no.	method I $\tilde{L}_{eq,w}$	method II $\tilde{L}_{eq,a}$
1	2.765	2.815
	2.865	2.815
2	3.613	3.613
	-	3.613
3	3.274	3.414
	3.556	3.414
4	3.609	3.609
	-	3.609

5.6.2 Conclusions

The original method, method I, normalises the mode-shapes in water using the frequency ratio, the modal mass in air, and the equivalent lengths of the mode-shapes in water and in air. The final mass normalised mode-shapes in water show an expected equivalent length.
The second, new method, method II, re-normalises the mode-shapes in air to use them as mass normalised mode-shapes in water. Also the equivalent lengths from these mode-shapes show an expected value compared to the simple theory.
If the second method gives similar well results when it is used as a space to do the identification, it is preferred with respect to the original method, because the mode-shapes obtained in air with the accelerometer are better discretised than the mode-shapes obtained in water using the strain gauges. From the measurement in water then only the frequency needs to be obtained. Next to that there is no approximation needed to calculate the equivalent lengths in this method.
On the other hand, one still needs to obtain the spectral density matrix from the strain sensors for the model submerged in the fluid flow.

5.6.3 Recommendations

Again, to truly test the methods discussed in this chapter, the experiments must be done correctly, giving out well mass normalised mode-shapes in air.
The Code_Aster command `NORM_MODE` needs to be adjusted to be able to mass normalise strain data, making the use of the modal mass data entry in a strain measurement file useful.

Part III

Identifying the Forces

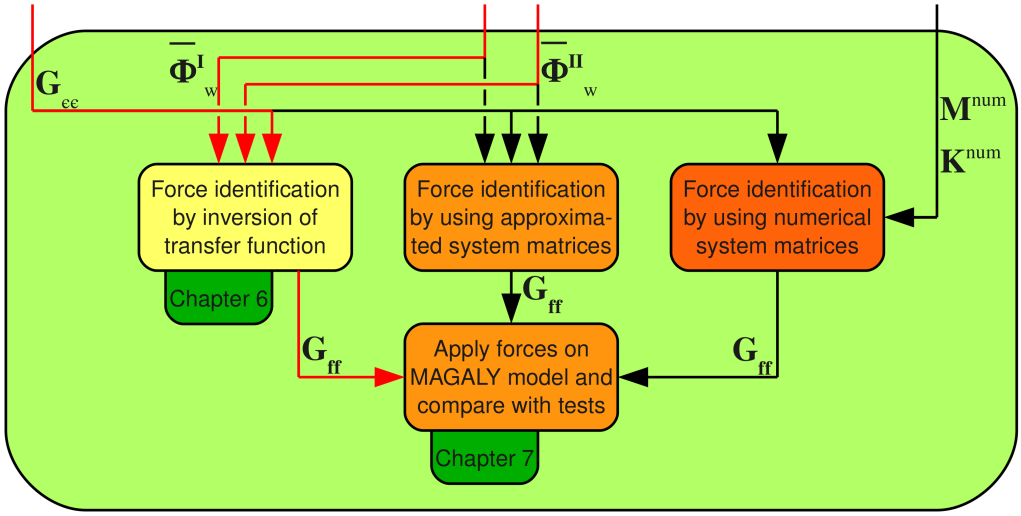


Figure 5.1: Scheme of the field of possibilities for the third part of the study. In red, the chosen path.

The forces modelling the fluid flow can be identified using a number of different possibilities. From the most left option of inversion of the FRF-matrix built on experimental mode-shapes, to the use of an updated numerical model on the far right of the scheme. In between there are all different ideas to obtain the mass and stiffness matrix from the experimental data. The numerical model could be updated with experimental mode-shapes as described in [12] and [13] or the mass and stiffness matrix could be approximated using the experimentally obtained mode-shapes as

$$\mathbf{M} = \left[\sum_i \overline{\boldsymbol{\phi}}_{(i)} \overline{\boldsymbol{\phi}}_{(i)}^{\mathrm{T}}\right]^{-1} = \left[\overline{\boldsymbol{\Phi}}\, \overline{\boldsymbol{\Phi}}^{\mathrm{T}}\right]^{-1} \qquad \mathbf{K} = \left[\frac{\sum_i \overline{\boldsymbol{\phi}}_{(i)} \overline{\boldsymbol{\phi}}_{(i)}^{\mathrm{T}}}{\omega_i^2}\right]^{-1} = \left[\overline{\boldsymbol{\Phi}} \boldsymbol{\Omega}^{-1} \overline{\boldsymbol{\Phi}}^{\mathrm{T}}\right]^{-1} \qquad \textit{Ref. [5]}$$

In this study is chosen to identify the forces by inverting the FRF-matrix built from the experimentally obtained mode-shapes, expanded on the numerical basis. This option is depicted as the red line through figure 5.1.

Chapter 6

Identifying the Fluid Forces

6.1 Introduction

In this chapter the fluid forces are modelled and identified by inverting the FRF-matrix. This FRF-matrix is described in appendix A.1. This identification process is done twice; (1) inverting the FRF-matrix built from the mass normalised mode-shapes in water (method I) of chapter 5, and (2) inverting the FRF-matrix built from the re-normalised mode-shapes in air (method II). While the measurements are done in water, and the data is thus in strain, the FRFs need to be adjusted. The measurement data from the LMS-mesurement system is obtain as a spectral density matrix in strain. The identification of the spectral densities of the forces is split in two stages, first the spectral densities of the modal forces is obtained and then the spectral density matrix of the physical forces is identified. These identified forces, obtained for both methods are finally compared.
First the general theory, with the use of strain measurements in the spectral density matrix is explained in section 6.2. Then the limitation when processing practical data and the way the fluid force is therefore modelled, are shown in 6.3. In section 6.4 the regulation is explained and the results of the identified force for both methods is shown in section 6.5. This chapter ends with comments, a conclusion and recommendations.

6.2 General Theory

To identify the forces of the fluid flow exciting the tube, the FRF-matrix needs to be inverted. The FRF-matrix showing the relation between the response, $\mathbf{x}(j\omega)$, and the fluid force, $\mathbf{f}(j\omega)$, is in matrix notation

$$\mathbf{x}(j\omega) = \overline{\mathbf{\Phi}}\mathbf{Z}(j\omega)^{-1}\overline{\mathbf{\Phi}}^{\mathrm{H}}\mathbf{f}(j\omega) \tag{6.1}$$

6.2.1 Using Strain

But, when measuring in water, the strain gauges are used, and the obtained data is therefore in strain. And because

$$\boldsymbol{\varepsilon} = r\frac{\partial^2}{\partial y^2}\mathbf{x}, \tag{6.2}$$

with r the inner radius of the tube, the strains can also be written as a modal sum. Thus, using matrix notation

$$\mathbf{x} = \mathbf{\Phi}\boldsymbol{\nu} \longleftrightarrow \boldsymbol{\varepsilon} = \mathbf{\Psi}\boldsymbol{\nu} \tag{6.3}$$

where $\mathbf{\Phi}$ contains the mode-shapes in displacement and $\mathbf{\Psi}$ contains the mode-shapes in strain. Therefore, in general terms, the built-up of the FRF as shown in appendix A.1 becomes

$$\left(-\omega^2\mathbf{M} + \mathbf{K}\right)\mathbf{x} = \mathbf{f} \tag{6.4}$$

$$\mathbf{\Phi}^{\mathrm{T}}\left(-\omega^2\mathbf{M} + \mathbf{K}\right)\mathbf{\Phi}\boldsymbol{\nu} = \mathbf{\Phi}^{\mathrm{T}}\mathbf{f}. \tag{6.5}$$

Because of the orthogonality property, the entries of $\boldsymbol{\nu}$, the modal coefficients, ν_i, become

$$\nu_i = \frac{\boldsymbol{\phi}_{(i)}^{\mathrm{T}}\mathbf{f}}{\mu_i\left(-\omega^2 + \omega_i^2\right)} \tag{6.6}$$

and thus, by substitution of equation 6.3 into equation 6.6, $\boldsymbol{\varepsilon}$ can be written as

$$\boldsymbol{\varepsilon}\left(j\omega\right) = \overline{\mathbf{\Psi}}\mathbf{Z}\left(j\omega\right)^{-1}\overline{\mathbf{\Phi}}^{\mathrm{H}}\mathbf{f}\left(j\omega\right). \tag{6.7}$$

6.2.2 Spectral Density Matrix

The measured data in the time domain, for the submerged tube induced by the fluid flow, are transferred to spectral densities by the LMS-measurement system, to take into account the convolution between the different measurement positions. In the Fourier transforms of the signals this can be seen as the phase-shift between the two signals. The spectral densities, can be built from the Fourier transposes of the strain gauge-signals as

$$\begin{aligned}
\mathbf{G}_{\varepsilon\varepsilon}\left(j\omega\right) &= \boldsymbol{\varepsilon}_{yy}\left(j\omega\right)\boldsymbol{\varepsilon}_{yy}^{\mathrm{H}}\left(j\omega\right) \\
&= \begin{bmatrix}
\varepsilon_{yy}|_1\left(j\omega\right)\varepsilon_{yy}^*|_1\left(j\omega\right) & \varepsilon_{yy}|_1\left(j\omega\right)\varepsilon_{yy}^*|_2\left(j\omega\right) & \dots & \varepsilon_{yy}|_1\left(j\omega\right)\varepsilon_{yy}^*|_N\left(j\omega\right) \\
\varepsilon_{yy}|_2\left(j\omega\right)\varepsilon_{yy}^*|_1\left(j\omega\right) & \varepsilon_{yy}|_2\left(j\omega\right)\varepsilon_{yy}^*|_2\left(j\omega\right) & \dots & \varepsilon_{yy}|_2\left(j\omega\right)\varepsilon_{yy}^*|_N\left(j\omega\right) \\
\vdots & \vdots & \ddots & \vdots \\
\varepsilon_{yy}|_N\left(j\omega\right)\varepsilon_{yy}^*|_1\left(j\omega\right) & \varepsilon_{yy}|_N\left(j\omega\right)\varepsilon_{yy}^*|_2\left(j\omega\right) & \dots & \varepsilon_{yy}|_N\left(j\omega\right)\varepsilon_{yy}^*|_N\left(j\omega\right)
\end{bmatrix}
\end{aligned}$$

where $(^{\mathrm{H}})$ is the hermitian transposed and $\varepsilon_{yy}^*|_1\left(j\omega\right)$ is the complex conjugate of $\varepsilon_{yy}|_1\left(j\omega\right)$. The diagonal shows the power spectral densities (PSD), equal to the Fourier transform of the auto-covariance of a time signal. And the off-diagonal terms are the cross spectral densities (CSD), equal to the Fourier transform of the cross-covariance between two time signals. These cross spectral densities are complex numbers describing the dynamic behaviour between different measurement points. The PSD's on the diagonal are not complex numbers because of the way they are formed mathematically. From a physical point of view this can also be explain, because between two sensors on one point there should not be any delay in the time domain, and thus no phase shift in the frequency domain. The CSD's can be complex and this complex part describes this behaviour. The CSD's on the lower triangular part of the matrix are the conjugate of their counterparts in the upper triangular matrix. So

$$g_{\varepsilon\varepsilon,ij}\left(j\omega\right) = g_{\varepsilon\varepsilon,ji}^*\left(j\omega\right) \tag{6.8}$$

and thus

$$\mathbf{G}_{\varepsilon\varepsilon}\left(j\omega\right)=\begin{bmatrix} g_{\varepsilon\varepsilon,11}\left(j\omega\right) & g_{\varepsilon\varepsilon,12}\left(j\omega\right) & \dots & g_{\varepsilon\varepsilon,1N}\left(j\omega\right) \\ g^{*}_{\varepsilon\varepsilon,12}\left(j\omega\right) & g_{\varepsilon\varepsilon,22}\left(j\omega\right) & \dots & g_{\varepsilon\varepsilon,2N}\left(j\omega\right) \\ \vdots & \vdots & \ddots & \vdots \\ g^{*}_{\varepsilon\varepsilon,1N}\left(j\omega\right) & g^{*}_{\varepsilon\varepsilon,2N}\left(j\omega\right) & \dots & g_{\varepsilon\varepsilon,NN}\left(j\omega\right) \end{bmatrix} \tag{6.9}$$

therefore the data-files are stored using just half the space compared with the full matrix. The obtained mode-shapes from the measurement systems, are non-complex, transferring the hermitian transpose into an ordinary transpose in equation 6.7 when handling the measurements. And, when using the power spectral density matrix, the transfer function can be described by

$$\mathbf{G}_{\varepsilon\varepsilon}\left(j\omega\right)=\mathbf{C}\overline{\mathbf{\Psi}}\mathbf{Z}\left(j\omega\right)^{-1}\overline{\mathbf{\Phi}}^{\mathrm{T}}\mathbf{B}\mathbf{G}_{ff}\left(j\omega\right)\mathbf{B}^{\mathrm{T}}\overline{\mathbf{\Phi}}\mathbf{Z}\left(j\omega\right)^{-1}\overline{\mathbf{\Psi}}^{\mathrm{T}}\mathbf{C}^{\mathrm{T}}. \tag{6.10}$$

6.3 Modelling the Fluid Force

There is a limited number of variables to model the fluid force, caused by the number of DoF in the measurements and the number of mode-shapes and frequencies used in the FRF-matrix. To keep the inversion to give a least square solution, the number of variables P, left to model the fluid force, should comply with

$$N \geqslant M \geqslant P \tag{6.11}$$

where N is the number of measurements DoF and M is the number of mode-shapes and frequencies. The fluid force is therefore modelled as a number of point loads, $\mathbf{f}^{\mathrm{model}}_{P\times 1}$, giving the same modal forces as a better discretized force, $\mathbf{f}_{n\times 1}$, or even the real fluid force, $f\left(y\right)$, in a continues analysis would do

$$\mathbf{q}_{M\times 1}=\overline{\mathbf{\Phi}}^{\mathrm{T}}\big|_{M\times n}\,\mathbf{B}_{n\times P}\,\mathbf{f}^{\mathrm{model}}_{P\times 1}=\overline{\mathbf{\Phi}}^{\mathrm{T}}\big|_{M\times n}\,\mathbf{f}_{n\times 1}=\begin{bmatrix} \int_L \phi_1\left(y\right) f\left(y\right)\,\mathrm{d}y \\ \int_L \phi_2\left(y\right) f\left(y\right)\,\mathrm{d}y \\ \vdots \\ \int_L \phi_M\left(y\right) f\left(y\right)\,\mathrm{d}y \end{bmatrix}. \tag{6.12}$$

Here, $\mathbf{B}$ is a Boolean matrix selecting the positions of the point loads. This is a $n \times P$ matrix filled with ones, for the numerical DoF for which a point force is to be identified and with zero, if there is no point force (see for an example the Notations section in the Nomenclature).
The vector $\mathbf{f}_{n\times 1}$ would represent the real fluid force much better, but it has too many DoF to be obtainable. The modelling of the fluid force is schematically depicted in figure 6.1, showing on the left side the tube in the continuous fluid flow and on the right hand side, the tube with the fluid force modelled by three point forces.
Next to the controllability Boolean matrix $\mathbf{B}$, there is a observability matrix $\mathbf{C}^{\mathrm{s}}$ specifying the strain gauges sensors positions along the tube. This is, similar to the controllability matrix $\mathbf{B}$, a matrix filled with ones and zeros. It has a one for each numerical DoF if there is a sensor measuring that DoF at that position and it has a zero if not.
The point forces $\mathbf{f}^{\mathrm{model}}_{p\times 1}$ can be obtained from the frequency response of the measured

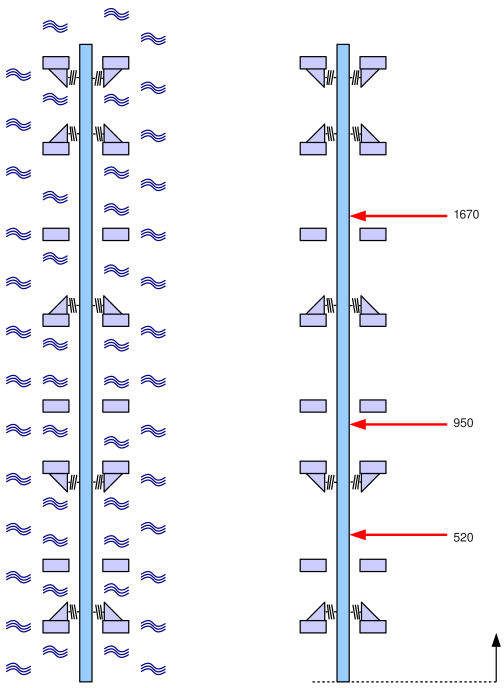


Figure 6.1: Schematic overview of of the tube in the continuous fluid flow on the left and with the fluid force modelled on the right.

strains. The FRF-matrix showing the relation between the measured response, $\boldsymbol{\varepsilon}^{\mathrm{ex}}$, and the obtainable load points, $\mathbf{f}^{\mathrm{model}}$, is in matrix notation

$$\boldsymbol{\varepsilon}^{\mathrm{ex}}(\jmath\omega) = \mathbf{C}^{\mathrm{s}}\overline{\boldsymbol{\Phi}}\mathbf{Z}^{-1}(j\omega)\,\overline{\boldsymbol{\Phi}}^{\mathrm{T}}\mathbf{B}\mathbf{f}^{\mathrm{model}}(j\omega) \tag{6.13}$$

To identify the point loads, the FRF-matrix is, together with the Boolean matrices, inverted using the pseudo inverse

$$\mathbf{f}^{\mathrm{model}}(\jmath\omega) = \left[\mathbf{C}^{\mathrm{s}}\overline{\boldsymbol{\Phi}}\mathbf{Z}^{-1}(j\omega)\,\overline{\boldsymbol{\Phi}}^{\mathrm{T}}\mathbf{B}\right]^{+}\boldsymbol{\varepsilon}^{\mathrm{ex}}(j\omega) \tag{6.14}$$

The experimental mode-shapes in water were obtained using the same sensors as the ones which are used to obtain the strain measurements in operation. Thus for the original method, method I, the mass-normalised, unexpanded experimental mode-shape data can be used, and thus there is no need for the Boolean matrix $\mathbf{C}^{\mathrm{s}}$. For the transposed mode-shapes, the mass normalised expanded experimental mode-shapes in water in displacement must be used. Equation 6.14 reduces for the original method to

$$\mathbf{f}^{\mathrm{model}}(\jmath\omega) = \left[\overline{\boldsymbol{\Psi}}_{\mathrm{w}}^{\mathrm{ex}}\mathbf{Z}_{\mathrm{w}}^{-1}(j\omega)\,\overline{\boldsymbol{\Phi}}_{\mathrm{w}}^{\mathrm{et\ T}}\mathbf{B}\right]^{+}\boldsymbol{\varepsilon}^{\mathrm{ex}}(j\omega) \tag{6.15}$$

For the new method, method II, using the re-normalised mode-shapes in air, for both the mode-shapes and the transposed mode-shapes, equation 6.14 becomes

$$\mathbf{f}^{\text{model}}(\jmath\omega) = \left[\mathbf{C}\overline{\mathbf{\Psi}}_{\text{a}}^{\text{et}}\mathbf{Z}_{\text{w}}^{-1}(j\omega)\,\overline{\mathbf{\Phi}}_{\text{a}}^{\text{et T}}\mathbf{B}\right]^{+}\varepsilon^{\text{ex}}(j\omega) \tag{6.16}$$

The identification of the spectral density force matrix, $\mathbf{G}_{ff}$, is done in two stages to be able to see the quality of both steps. Figure 6.2 shows a schematic representation of these steps.

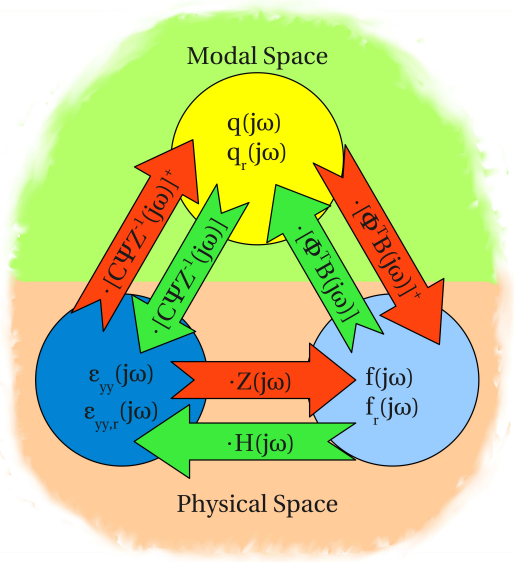


Figure 6.2: Schematic overview of transformations between strains, ε_{yy}, and force, $\mathbf{f}$, via the modal forces $\mathbf{q}$.

So first from the strains to the modal forces, then from the modal forces to the point forces. So, for method I, the spectral density modal force matrix, $\mathbf{G}_{qq}$, is obtained by

$$\begin{aligned}\mathbf{G}_{qq}(j\omega) &= \left[\overline{\mathbf{\Psi}}_{\text{w}}^{\text{ex}}\mathbf{Z}_{\text{w}}^{-1}(j\omega)\right]^{+}\mathbf{G}_{\varepsilon\varepsilon}(j\omega)\left[\mathbf{Z}_{\text{w}}^{-1}(j\omega)\,\overline{\mathbf{\Psi}}_{\text{w}}^{\text{ex T}}\right]^{+} &(6.17)\\ &= \left[\overline{\mathbf{\Psi}}_{\text{w}}^{\text{ex}}\mathbf{Z}_{\text{w}}^{-1}(j\omega)\right]^{+}\mathbf{G}_{\varepsilon\varepsilon}(j\omega)\left(\left[\overline{\mathbf{\Psi}}_{\text{w}}^{\text{ex}}\mathbf{Z}_{\text{w}}^{-1}(j\omega)\right]^{+}\right)^{\text{T}} &(6.18)\end{aligned}$$

then, the spectral density force matrix, $\mathbf{G}_{ff}$, is obtained from $\mathbf{G}_{qq}$

$$\begin{aligned}\mathbf{G}_{ff}(j\omega) &= \left[\overline{\mathbf{\Phi}}_{\text{w}}^{\text{et T}}\mathbf{B}\right]^{+}\mathbf{G}_{qq}(j\omega)\left[\mathbf{B}^{\text{T}}\overline{\mathbf{\Phi}}_{\text{w}}^{\text{et}}\right]^{+} &(6.19)\\ &= \left[\overline{\mathbf{\Phi}}_{\text{w}}^{\text{et T}}\mathbf{B}\right]^{+}\mathbf{G}_{qq}(j\omega)\left(\left[\overline{\mathbf{\Phi}}_{\text{w}}^{\text{et T}}\mathbf{B}\right]^{+}\right)^{\text{T}} &(6.20)\end{aligned}$$

For the new method, method II, a similar process is described. Again, the spectral density modal force matrix, $\mathbf{G}_{qq}$, is first obtained

$$\mathbf{G}_{qq}(j\omega) = \left[\mathbf{C}\overline{\mathbf{\Psi}}_{\mathrm{a}}^{\mathrm{et}}\mathbf{Z}_{\mathrm{w}}^{-1}(j\omega)\right]^{+}\mathbf{G}_{\varepsilon\varepsilon}(j\omega)\left[\mathbf{Z}_{\mathrm{w}}^{-1}(j\omega)\,\overline{\mathbf{\Psi}}_{\mathrm{a}}^{\mathrm{et\ T}}\mathbf{C}^{\mathrm{T}}\right]^{+} \tag{6.21}$$

$$= \left[\mathbf{C}\overline{\mathbf{\Psi}}_{\mathrm{a}}^{\mathrm{et}}\mathbf{Z}_{\mathrm{w}}^{-1}(j\omega)\right]^{+}\mathbf{G}_{\varepsilon\varepsilon}(j\omega)\left(\left[\mathbf{C}\overline{\mathbf{\Psi}}_{\mathrm{a}}^{\mathrm{et}}\mathbf{Z}_{\mathrm{w}}^{-1}(j\omega)\right]^{+}\right)^{\mathrm{T}} \tag{6.22}$$

then, the spectral density force matrix, $\mathbf{G}_{ff}$, is obtained from $\mathbf{G}_{qq}$

$$\mathbf{G}_{ff}(j\omega) = \left[\overline{\mathbf{\Phi}}_{\mathrm{a}}^{\mathrm{et\ T}}\mathbf{B}\right]^{+}\mathbf{G}_{qq}(j\omega)\left[\mathbf{B}^{\mathrm{T}}\overline{\mathbf{\Phi}}_{\mathrm{a}}^{\mathrm{et}}\right]^{+} \tag{6.23}$$

$$= \left[\overline{\mathbf{\Phi}}_{\mathrm{a}}^{\mathrm{et\ T}}\mathbf{B}\right]^{+}\mathbf{G}_{qq}(j\omega)\left(\left[\overline{\mathbf{\Phi}}_{\mathrm{a}}^{\mathrm{et\ T}}\mathbf{B}\right]^{+}\right)^{\mathrm{T}} \tag{6.24}$$

6.4 Regularisation

To cope with noise in the high frequency range and to realise properties of the fluid flow force known from literature [8], the singular values are subdued to an adjusted Tikhonov [11] regularisation.
For the higher frequency range (> 200 Hz) the experimentally obtained spectral density shows just noise, as to be seen in figure 6.3. This is due to the fact that the strain gauges measure with an accuracy of around 0.1 μE (the unit μE is 10^{-6} times the strain $\frac{\Delta L}{L}$). The effect of this noise in the calculation can be counteracted by applying a truncation to the singular value decomposition (TSVD), or by using a Tikhonov regularisation, both described in appendix A.3 and more extensively in [11]. This noise will generate a non-realistic high frequency force after the inversion. From literature, [8], is known that the PSD of the pressure should have a decreasing slope as shown in figure 6.4. To take care of the noise and at the same time enforce a slope similar to the slope in figure 6.4 an adjusted Tikhonov regularisation is applied. With this regularisation the pseudo inverse of the first stage, calculating the modal forces becomes

$$\left[\mathbf{C}\mathbf{\Psi}\mathbf{Z}^{-1}(j\omega)\right]^{+} = \mathbf{V}(j\omega)\,\mathbf{\Sigma}_{\alpha}^{-1}(j\omega)\,\mathbf{U}^{\mathrm{T}}(j\omega) \tag{6.25}$$

where $\mathbf{\Sigma}_{\alpha}^{-1}$ contains the regularisation. In stead of the usual inverse of the singular values $\frac{1}{\sigma_i}$ on the diagonal of $\mathbf{\Sigma}^{-1}$, it is now calculated as

$$\mathbf{\Sigma}_{\alpha}^{-1}(j\omega) = \begin{bmatrix}\mathbf{D}_{\alpha}(j\omega)\\ \mathbf{0}\end{bmatrix}_{N\times M} \tag{6.26}$$

where

$$\mathbf{D}_{\alpha}(j\omega) = \begin{bmatrix}\frac{\sigma_1}{\sigma_1^2+\alpha_1(\omega)} & & & \varnothing\\ & \frac{\sigma_2}{\sigma_2^2+\alpha_2(\omega)} & & \\ & & \ddots & \\ \varnothing & & & \frac{\sigma_M}{\sigma_M^2+\alpha_M(\omega)}\end{bmatrix}_{M\times M} \tag{6.27}$$

where $\alpha_i(j\omega)$ is the Tikhonov regulator. By making this regulator dependent on the frequency, the regularisation, and with that the slope of the FRF becomes controllable.

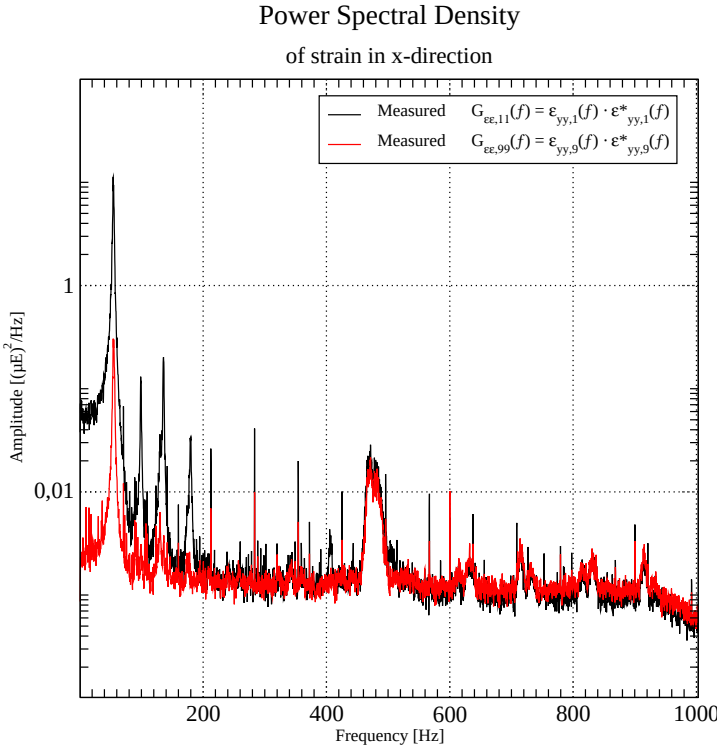


Figure 6.3: Graph showing the spectral density of the strain measurement on the first two sensors measuring equivalent displacement in two directions.

By adjusting the regularisation parameter $\alpha_i(j\omega)$ in the transformed space, it can be made dependant on frequency. By increasing the value of α for frequencies higher then the eigen frequency, being the cause of a large singular value, the noise in the high frequency range can be filtered out. By adjusting the power of a multiplier of α_0, the slope in the high frequency range becomes controllable.

$$\boldsymbol{\alpha}(\omega) = \alpha_0 \mathbf{V}^{\mathrm{H}} \left(\mathbf{V} \begin{bmatrix} \begin{cases} 1 & \text{if} \quad \omega \leqslant \omega_1 \\ 1 + (\omega - \omega_i)^m & \text{if} \quad \omega > \omega_1 \end{cases} \\ \vdots \\ \begin{cases} 1 & \text{if} \quad \omega \leqslant \omega_M \\ 1 + (\omega - \omega_M)^m & \text{if} \quad \omega > \omega_M \end{cases} \end{bmatrix} \right) \tag{6.28}$$

The parameter α_0 is obtained by observing the singular values in figure 6.5 and 6.6. The value of α_0 needs to be in the order of σ^2 to have any effect in the Tikhonov regularisation ($\frac{\sigma}{\sigma^2+\alpha_0}$). From both figures it is clear that the lowest singular value after the last eigen frequency is around $3 \cdot 10^{-7}$, so the α_0 is chosen to be $\sigma^2 \approx 1 \cdot 10^{-13}$.
Picturing the system in the frequency domain as a sum of singular DoF systems, the slope of the frequency response function after the highest eigen frequency in the system should have a downward angle of -40 dB corresponding to $\frac{1}{\omega^2}$. Under the assumption

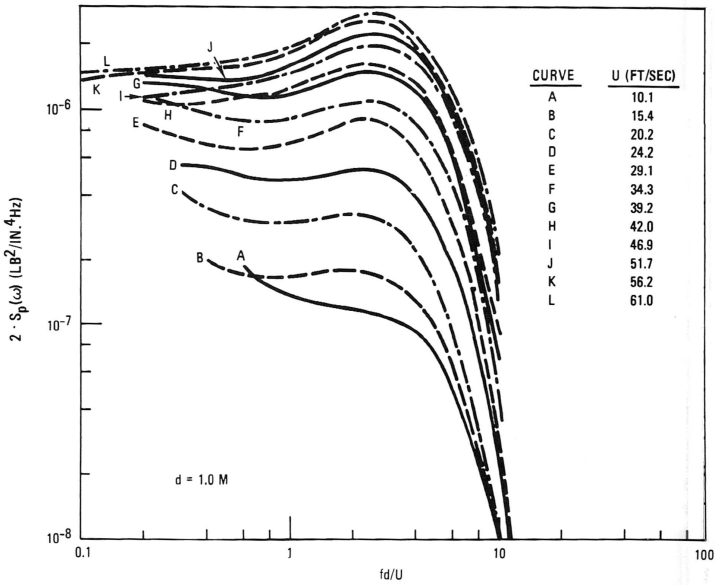


Figure 6.4: Graph showing a number of spectral densities of pressures induced by the fluid flow to a tube as a function of fd/U where f is the frequency, d is the diameter of the tube and U the speed of the flow[8].

that the fluid force function in the frequency domain for that frequency range can be seen as random noise, so a flat line, the response should also show this downward angle of -40 dB. Using the spectral densities, the response is squared and should therefore show a downward angle of -80 dB.
Using trail and error, the value for m is finally set to be 0.2, giving a slope of -80 dB, corresponding with the ω^{-4} prescription.

6.5 Results

After identifying the forces, the process can be reversed and the spectral density matrix in strain can be re-calculate. Comparing the measured PSD with the re-calculated one gives a first indication of the quality of the process. Simply speaking, it is like first dividing and then multiplying, with the same amount. It should therefore give a similar result at low frequency, and at high frequency the effect of the regulation should be visible.
The second criterion to see if the obtained forces model the real fluid force, is by comparing the shape of one of the PSD of the point forces with the PSD of the pressures along a tube in figure 6.4. These shapes should be comparable because the surface area on which the pressure acts along the tube is constant.

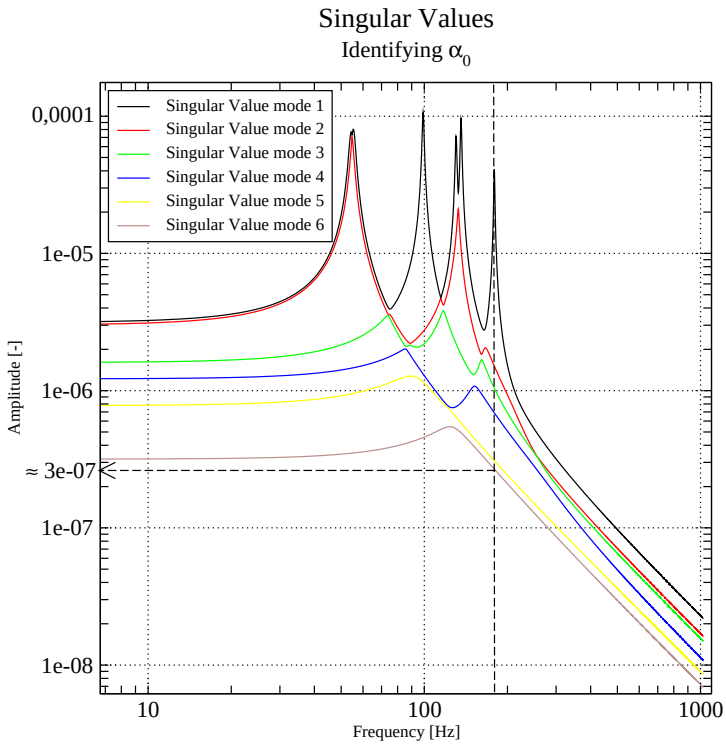


Figure 6.5: Graph showing the singular values depending on frequency in the SVD of $\left[\overline{\boldsymbol{\Psi}}_{\mathrm{w}}^{\mathrm{ex}}\mathbf{Z}_{\mathrm{w}}^{-1}\left(j\omega\right)\right]$ for method I and the choice of $\sqrt{\alpha_0}$.

Remark: *To handle strain measurements, the observability matrix* $\left[\boldsymbol{C\Psi Z}^{-1}\left(j\omega\right)\right]$ *must be used in stead of the displacement observability matrix* $\left[\boldsymbol{C\Phi Z}^{-1}\left(j\omega\right)\right]$. *This is not yet possible in* `CALC_ESSAI`, *but if the strain mode-shapes* $\boldsymbol{\Psi}$ *are exported in Code_Aster, using Python, and later on re-read as being 'displacement' mode-shapes,* `CALC_ESSAI` *can be tricked to use strain shapes anyway. For the original method, method I, only the experimental mode-shapes in strain have to be read-in as mode-shapes in displacement. But for the second method, method II, the expanded experimental PAK mode-shapes are used. These mode-shapes must first be written in strain to a data file using a python routine and later re-read as 'displacement' mode-shapes form this file.*

The orientation of the tube when submerged was not recorded during the measurements, so it is impossible to give a direction of movement relative to the fluid flow. The results will be presented for both methods in a similar fashion. For both methods, the measured PSD and the re-calculated PSD in strain for one of the two sensors at the bottom of the tube is shown in one figure. The PSD of the other strain sensor at the bottom of the tube, measuring strain due to a displacement in a perpendicular local z-direction is shown

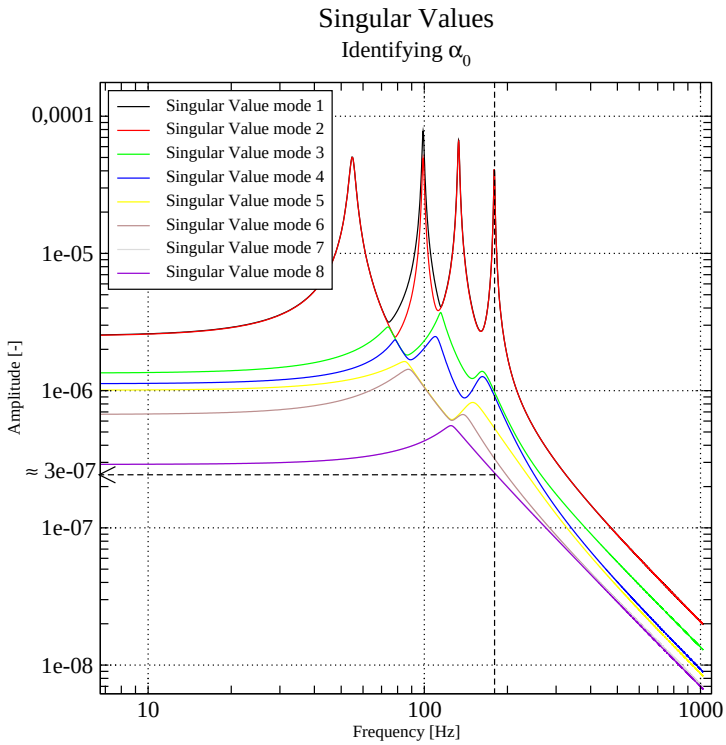


Figure 6.6: Graph showing the singular values depending on frequency in the SVD of $\left[\mathbf{C}^{\mathrm{s}}\overline{\mathbf{\Psi}}_{\mathrm{a}}^{\mathrm{et}}\mathbf{Z}_{\mathrm{w}}^{-1}\left(j\omega\right)\right]$ for method II the and choice of $\sqrt{\alpha_0}$.

in the next figure. These bottom sensors are selected because they are near the biggest amplitudes of the vibrating rod, thus showing the relative most accurate measurements. The PSD's of the identified forces on the different points is shown in the same local x-direction next figure. The last figure per method completes the set and shows the PSD's is the corresponding z-direction.

6.5.1 Method I: Using Mass Normalised Mode-shapes in Water

Figures showing the PSD's of strain

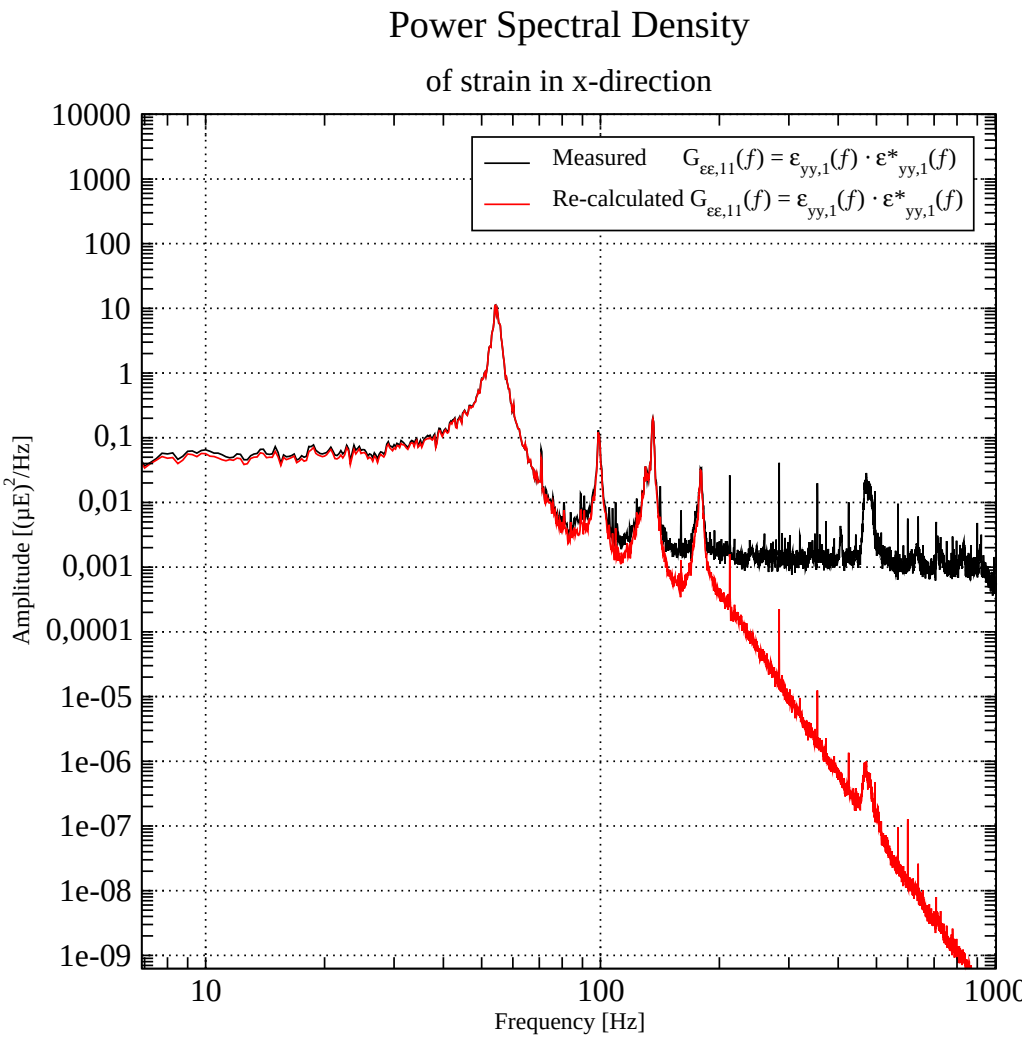


Figure 6.7: Power Spectral Densities in strain of measured data and re-calculated data for the sensor at the bottom of the tube measuring in a local x-direction. The re-calculated data is obtained using the mode-shapes normalised with method I.

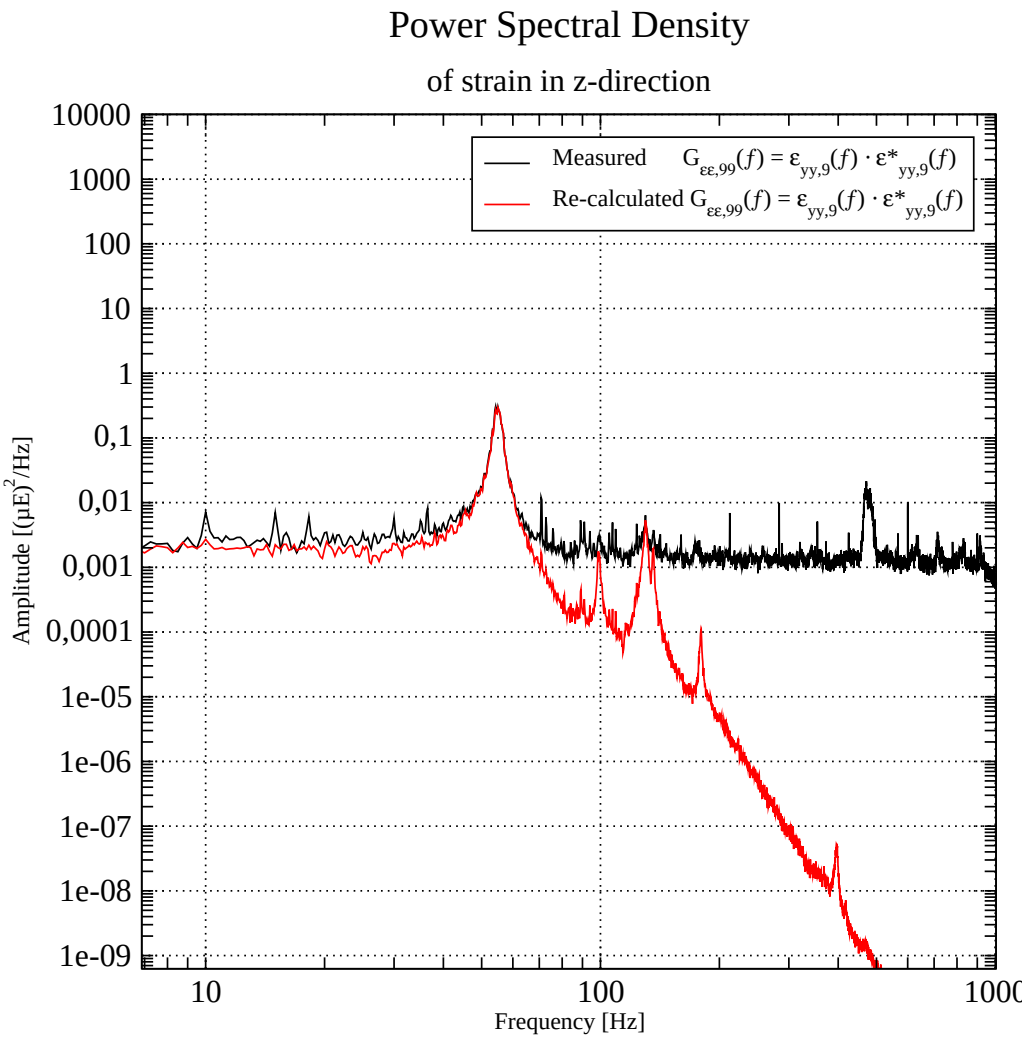


Figure 6.8: Power Spectral Densities in strain of measured data and re-calculated data for the sensor at the bottom of the tube measuring in a local z-direction. The re-calculated data is obtained using the mode-shapes normalised with method I.

Both figures show a really good re-calculability for this method, method I. They also show the desired slope in the re-calculated PSD.

Figures showing the PSD's of force

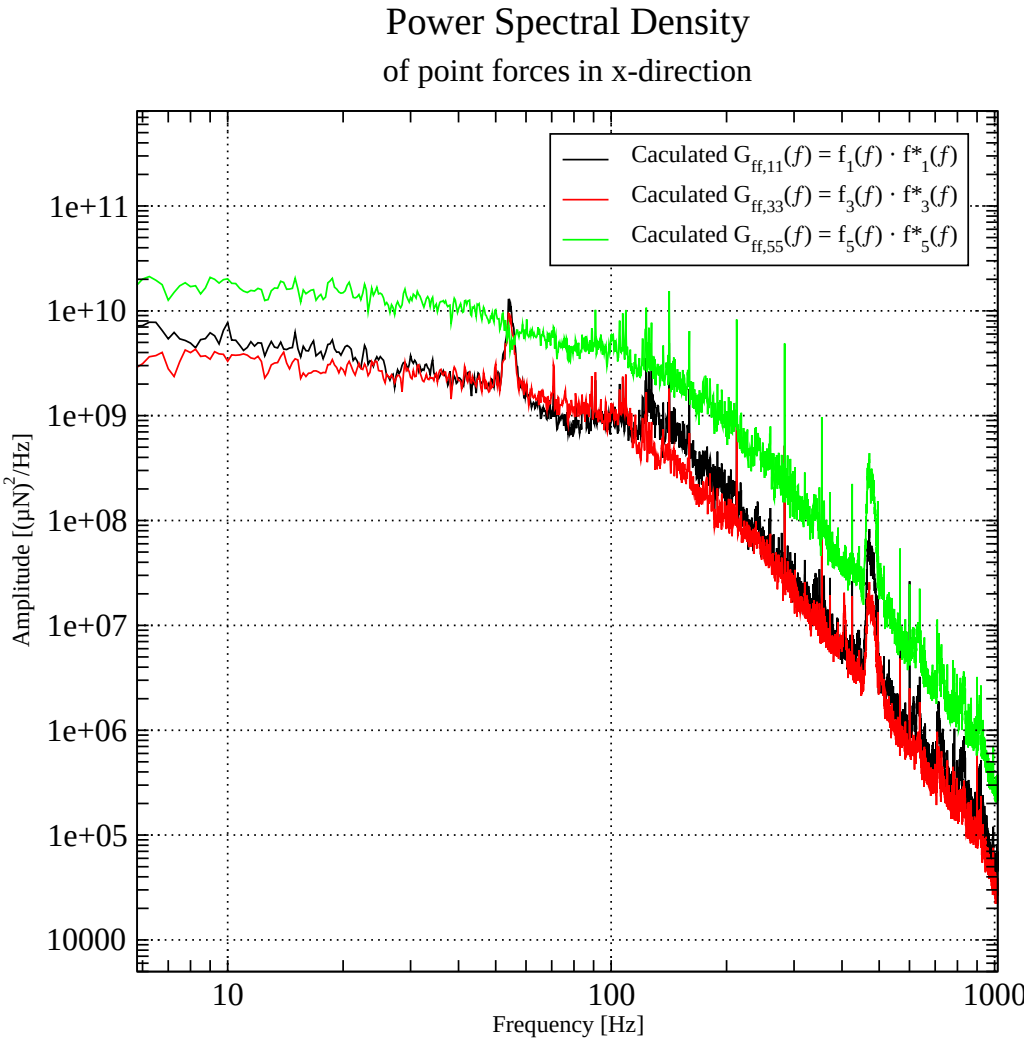


Figure 6.9: Power Spectral Densities of point forces of calculated data for the three different positions in a local x-direction, using the mode-shapes normalised with method I.

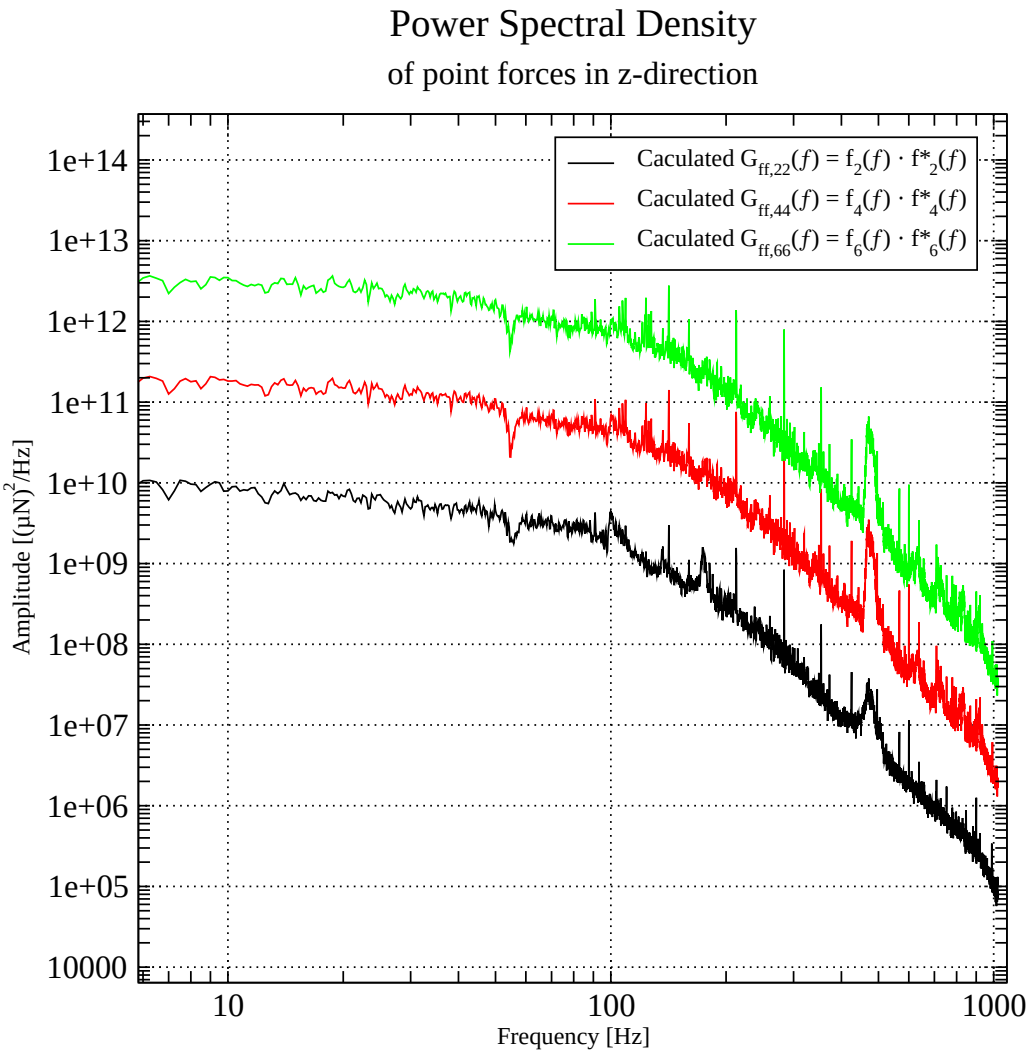


Figure 6.10: Power Spectral Densities of point forces of calculated data for the three different positions in a local z-direction, using the mode-shapes normalised with method I.

Both figures show a shape of the PSD of the identified force, which shows some similarity to the shape of the PSD of the pressures depicted in figure 6.4.

6.5.2 Method II: Using Re-normalised Mode-shapes in Air

Figures showing the PSD's of strain

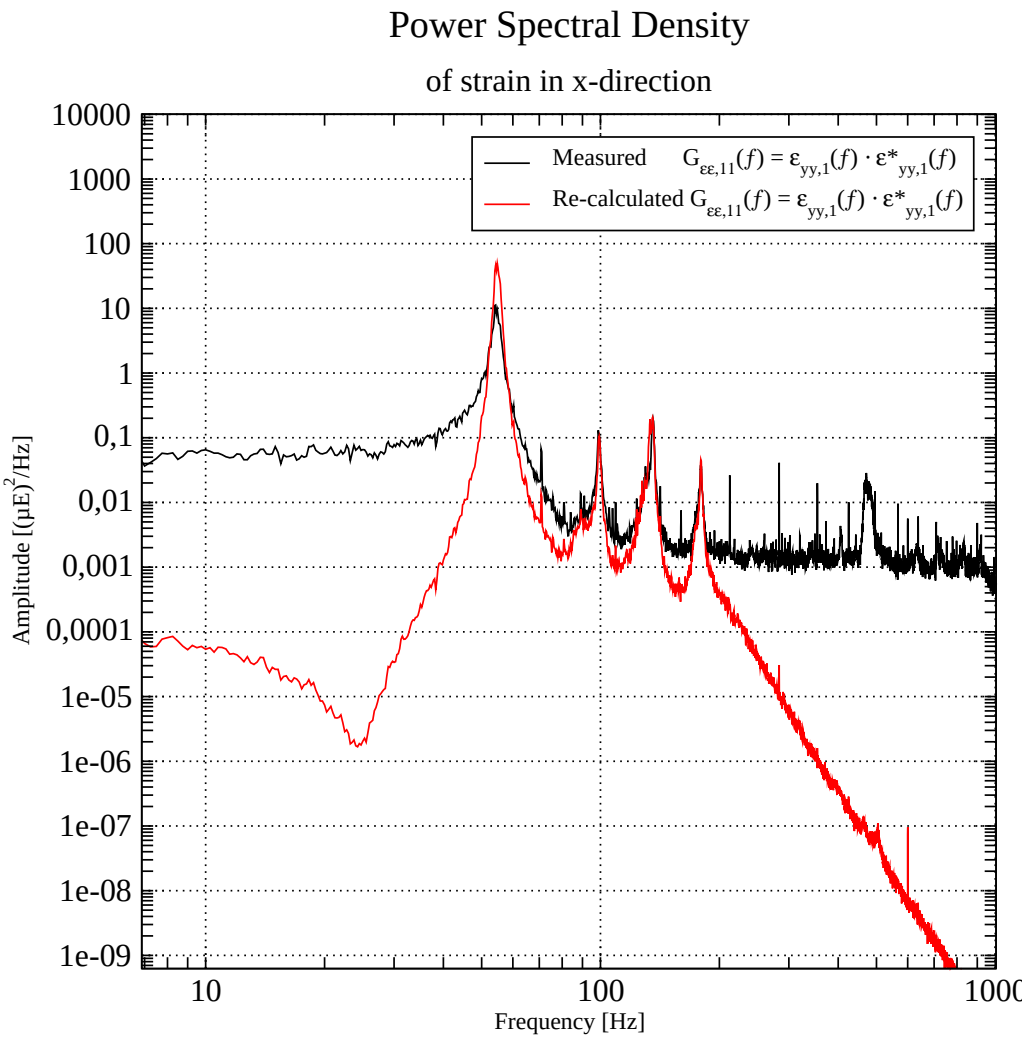


Figure 6.11: Power Spectral Densities in strain of measured data and re-calculated data for the sensor at the bottom of the tube measuring in a local x-direction. The re-calculated data is obtained using the mode-shapes normalised with method II.

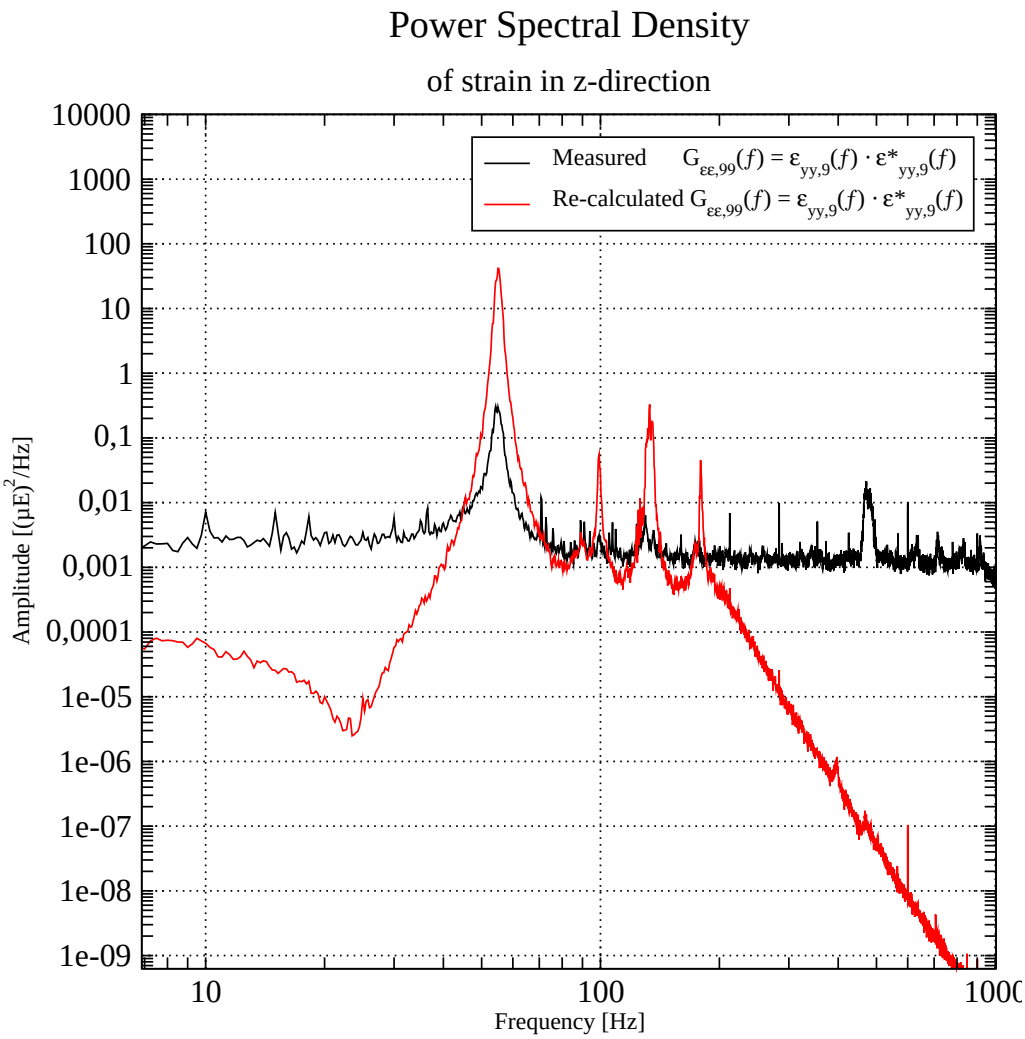


Figure 6.12: Power Spectral Densities in strain of measured data and re-calculated data for the sensor at the bottom of the tube measuring in a local z-direction. The re-calculated data is obtained using the mode-shapes normalised with method II.

Both figures, figures 6.11 and 6.12, show a bad re-calculated PSD. The second method, method II, which combines eigen frequencies in water with the mode shapes in air, gives a bad result compared to the figures of method I, in which all modal parameters are obtained in water.

Figures showing the PSD's of force

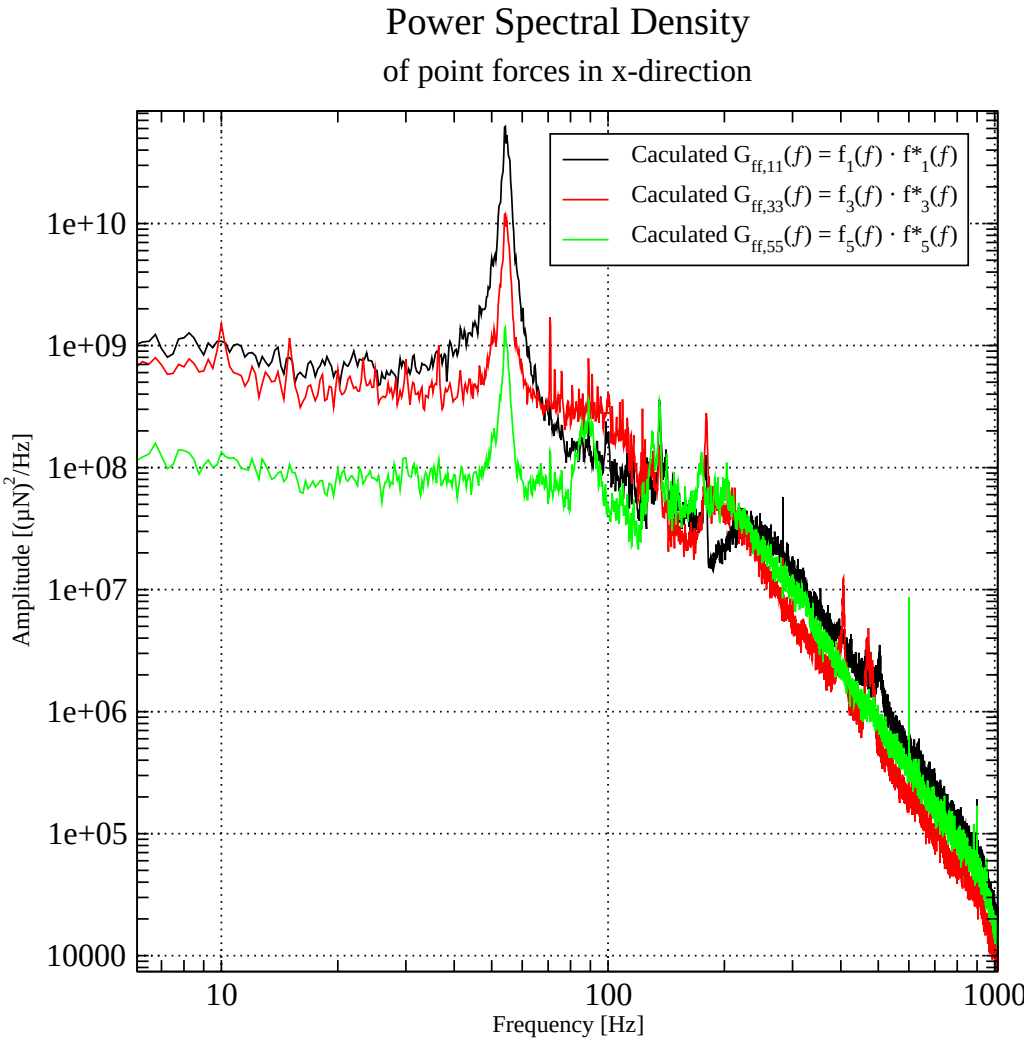


Figure 6.13: Power Spectral Densities of point forces of calculated data for the three different positions in a local x-direction, using the mode-shapes normalised with method II.

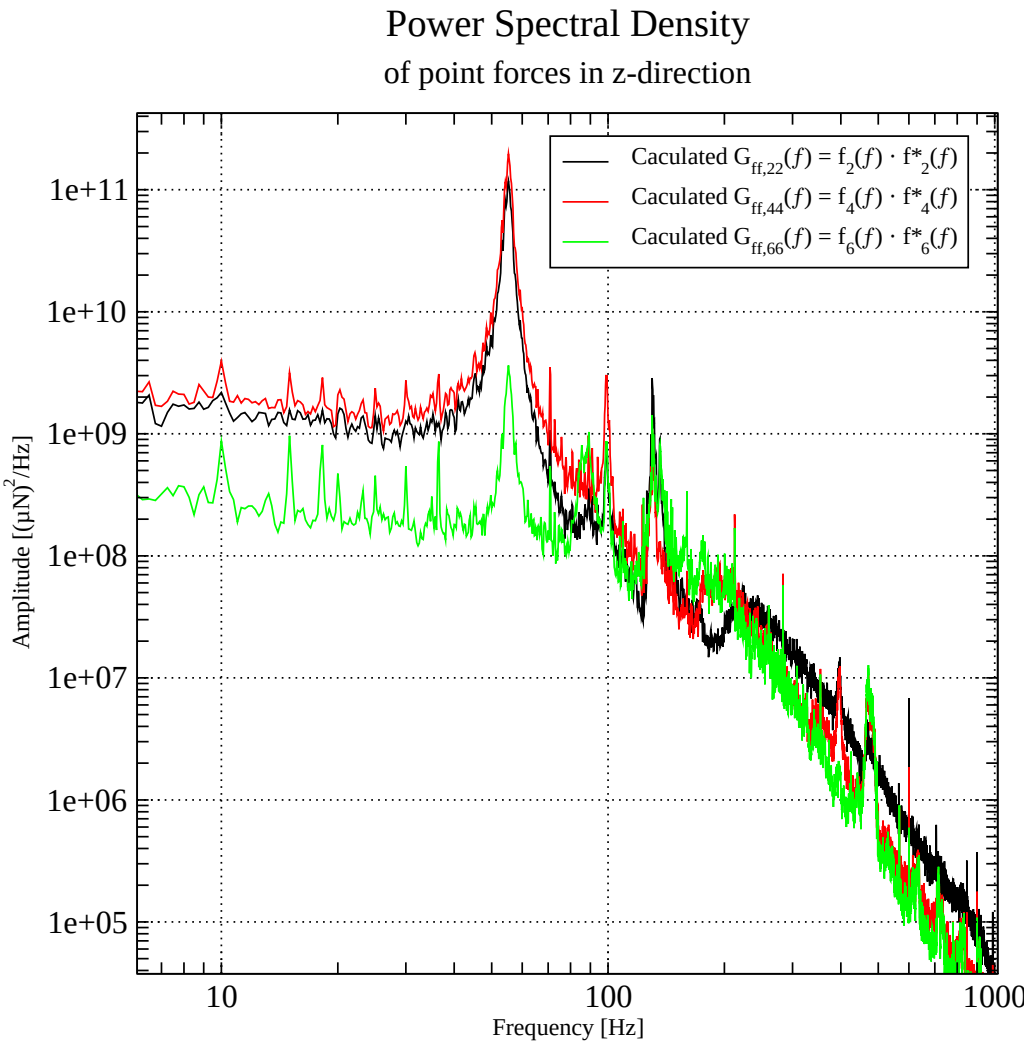


Figure 6.14: Power Spectral Densities of point forces of calculated data for the three different positions in a local z-direction, using the mode-shapes normalised with method II.

Both figures, figures 6.13 and 6.14, show again a shape of the PSD of the identified force, which is similar to the shape of the PSD of the pressures depicted in figure 6.4, except around the eigen frequencies of the system. This indicates that the system is not well described using the re-normalised mode-shapes in water, because the expected fluid force does not show any peak frequency. This result is therefore rather worthless, not only for these peaks at the eigen frequencies, but also because the recalculation of the PSDs in strain are not possible.

6.5.3 Comparing the Two Methods

The different methods to describe the system can easily be compared looking at the modal force. The calculated modal force and the re-calculated modal force should be quite the same because the calculated modal force is just multiplied by a pseudo-inverse independent of frequency $\left[\mathbf{\Phi}^{\mathrm{T}}\mathbf{B}\right]^{+}$ end afterwards multiplied by $\left[\mathbf{\Phi}^{\mathrm{T}}\mathbf{B}\right]$.

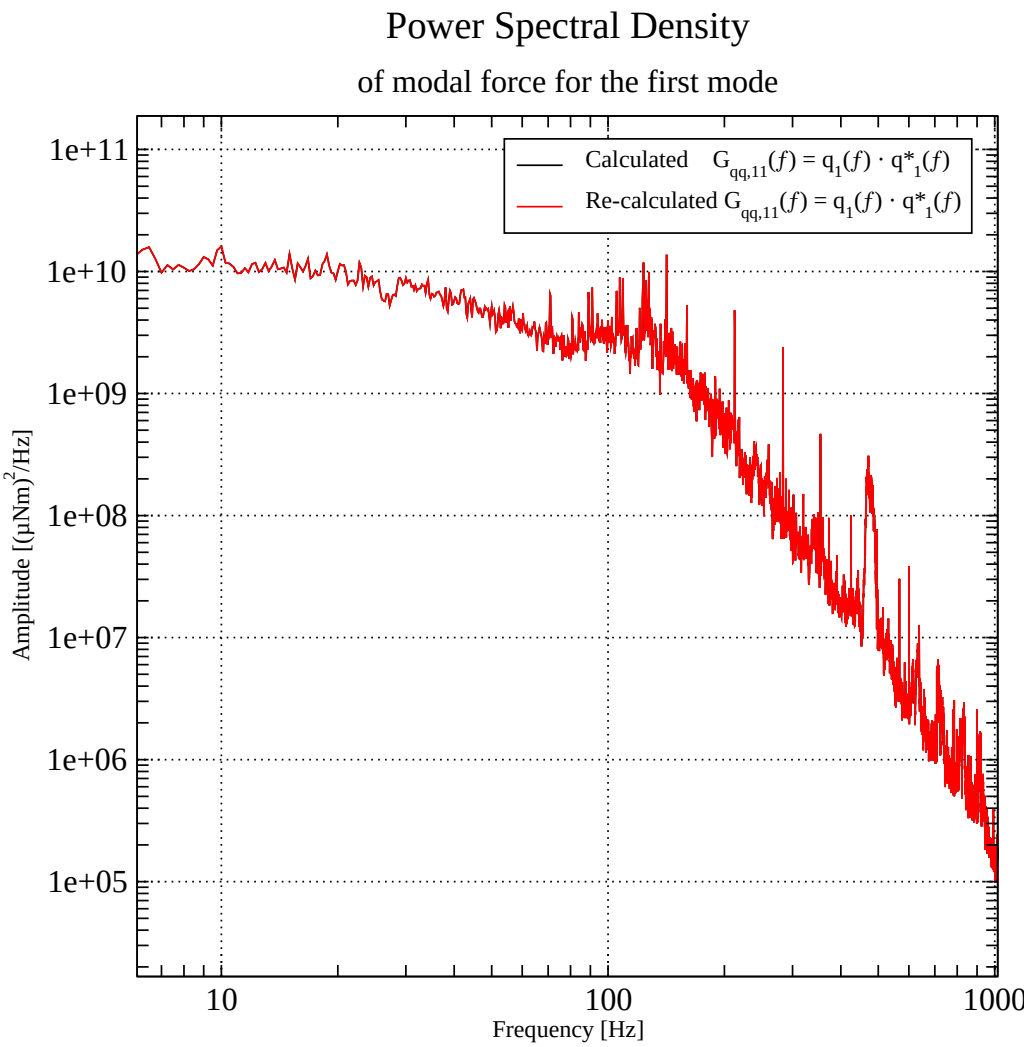


Figure 6.15: Power Spectral Densities of the calculated modal force and re-calculated modal force for the first eigen mode. The corresponding mode-shape was mass normalised with method I.

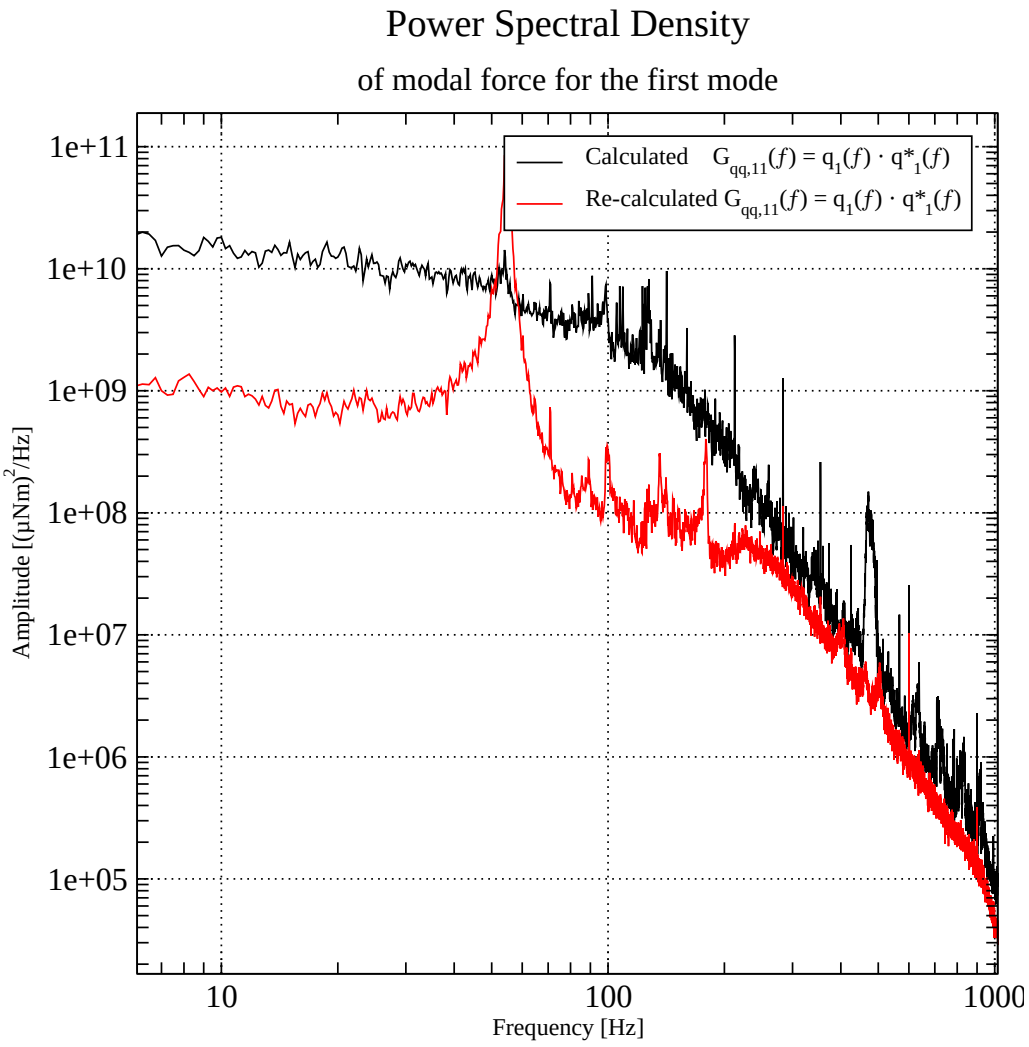


Figure 6.16: Power Spectral Densities of the calculated modal force and re-calculated modal force for the first eigen mode. The corresponding mode-shape was mass normalised with method II.

When comparing figure 6.15 with figure 6.16 it is easily seen that the modal force for the first mode in figure 6.16 shows a big discrepancy between the calculated and the re-calculated one. Because, when using method II, the re-normalised mode-shapes in air are used and these mode-shapes model the system worse than the mode-shapes in water. This difference shows itself by each least square approximation, when using the pseudo inverse. Between the calculated modal force $q\left(j\omega\right)$ and the re-calculated modal force $q_r\left(j\omega\right)$, there are two steps of which one is a least square approximation using the pseudo inverse as schematically shown in figure 6.2. It is obvious that the least square

approximation is a lot worse when the mode-shapes in air, mass normalised with method II, are used, than when the mode-shapes in water, mass normalised with method I, are used. Thus the system is far worse modelled using the mode-shapes in air as when the mode-shapes in water are used.
The hypothesis of chapter 5 cannot be applied as strictly as is necessary to deduced method II. Method I is, although in a lot less strict way, also dependent on the hypothesis. One could question if the mass normalisation is done correct and if the magnitudes of the point forces obtained using this method are therefore correct.

6.6 Comments, Conclusions & Recommendations

6.6.1 Comments

One of the strain gauge sensors, number 14, gave only measurement noise. It was evicted from the identification process.
In the study by Bodel [1] there is the appendix A1, which elaborates on the assumed difference in slope for the high frequency range between the plots of Fourier transfer of the signal in displacement and the Fourier transfer of the signal in strain. The plot containing the Fourier transfer of the strain signal is assumed to have a slope of ω^{-1} for high frequency range, in stead of the slope for displacement signal of ω^{-2}. This is supposed to be the case because

$$\varepsilon(y,t) = r\frac{\partial^2 X}{\partial y^2} \tag{6.29}$$

where $X(y,t)$ is defined as

$$X(y,t) = \cos(ky)\cos(\omega t) \tag{6.30}$$

and therefore $\varepsilon(y,t)$ becomes

$$\varepsilon(y,t) = -k^2 r X(y,t). \tag{6.31}$$

This is not true, because $X(y,t)$ is more like

$$X(y,t) = \{\sin(ky) + \cos(ky) + \sinh(ky) + \cosh(ky)\}\cos(\omega t) \tag{6.32}$$

and therefore

$$\varepsilon(y,t) \neq -k^2 r X(y,t). \tag{6.33}$$

The parameter $-k^2$ is then combined in the appendix with the homogeneous Euler-Bernoulli equation stating that

$$\frac{EI}{\rho A}\frac{\partial^4 X}{\partial y^4} + \frac{\partial^2 X}{\partial t^2} \tag{6.34}$$

giving

$$\frac{EI}{\rho A}k^4 - \omega^2 = 0 \tag{6.35}$$

which is correct. But it should only be used to calculate the eigen frequencies and eigen mode-shapes

$$\frac{EI}{\rho A}k_i^4 - \omega_i^2 = 0. \tag{6.36}$$

Equation 6.31 and 6.35 are then combined to give

$$\varepsilon(y,t) = -\omega r\sqrt{\frac{\rho A}{EI}}X(y,t) \sim \omega X(y,t) \tag{6.37}$$

and then the slope of the high frequency part of the FRF is adjusted from ω^{-2} to ω^{-1}. This is wrong, because the homogeneous Euler-Bernoulli equation should just be used to obtain the eigen frequencies ω_i in correspondence with their respective 'eigen'-shape

parameters k_i. Comparing the FRF-matrix for displacement with the one for strain, shows a similar dynamic behaviour

$$\mathbf{x}^{\mathrm{ex}}(j\omega) = \sum_i \frac{\mathbf{C}^{\mathrm{a}}\boldsymbol{\phi}_{(i)}\boldsymbol{\phi}_{(i)}^{\mathrm{T}}\mathbf{B}}{-\omega^2 + 2j\xi_i\omega\omega_i + \omega_i^2}\mathbf{f}(j\omega) \tag{6.38}$$

$$\boldsymbol{\varepsilon}^{\mathrm{ex}}(j\omega) = \sum_i \frac{\mathbf{C}^{\mathrm{s}}\boldsymbol{\psi}_{(i)}\boldsymbol{\phi}_{(i)}^{\mathrm{T}}\mathbf{B}}{-\omega^2 + 2j\xi_i\omega\omega_i + \omega_i^2}\mathbf{f}(j\omega)\,. \tag{6.39}$$

Both FRF-matrices should show a dynamic behaviour of ω^{-2} for their slopes at high frequency range in absence of more eigen frequencies.

6.6.2 Conclusions

Method I, in which the mass normalised mode-shapes in water are used in the FRF-matrix, gives a far better result in the inversion than method II, in which the re-normalised mode-shapes in air are used. The mode-shapes in water model the system better than the re-normalised mode-shapes in air do. The mode-shapes in water give a better result in the least square approximation than the mode-shapes in air. The error obtained in the process is easily seen when comparing the measured PSD with the re-calculated one. It is also shown when comparing the modal forces for the different methods. Apparently, the hypotheses from in chapter 5 cannot be used in such a stringent manner.
It shows however also one concern for method I, which is not directly visible. In method I the normalisation is done approximating the constant C_i by the equivalent length under the assumption that the mode-shapes are the same in air as in water (in the less stringent way). Although the approximation has an averaging character, if the mode-shapes are not similar enough, this method can still mass normalise the mode-shapes wrongly and therefore the obtained point forces could model the fluid force incorrectly. This cannot be seen when comparing the re-calculated with the measured PSD, because this is similar to multiplying and dividing as earlier explained. This is also the reason Bodel [1] did not noticed he had used non-normalised mode-shapes.

6.6.3 Recommendations

`CALC_ESSAI` should be adjusted to use strain measurements instead of only displacement measurements. This to make the routine more user friendly and to avoid the chance on errors using python routines to store strain data as displacement data.

Chapter 7

Validating the Method on the Magaly Model

7.1 Introduction

In this chapter the Magaly model is introduced. The experimental Magaly model is a model of the whole control rod assembly and therefore represents the real control rods a lot better, than the Phacetie model with its single control rod. This model is used to validate the identified forces obtained on the Phacetie model. The Magaly model is an experimental model built at Framatome (Areva) and was use to obtain fluid drag force data and vibrational amplitude data at the guide plate positions as described in [14].
To validate the identified forces obtained on the Phacetie model, they are applied on a numerical representation of the Magaly model in Code_Aster. Using this numerical model, the vibration at the position of the guide plates can be simulated and calculated. By comparing the experimental vibration data obtained by Areva, with the calculated vibrational data obtained on the numerical model, the identified fluid forces can be validated.
First the experimental Magaly model is described in section 7.2. Then the numerical representation of this model is shown in section 7.3. Section 7.4 shows how a time domain force function should be constructed from the power spectral density (PSD) of the forces. Because this has not been successfully accomplished and the numerical model does not model the experimental Magaly model in a way it can be used, there are so far no results obtained.
Furthermore, some insight show that by having modelled the fluid force the way it was done, no desirable results will be obtainable when it is applied to a different model with different mode-shapes. This is explained in section 7.5. The chapter is again finished with comments, conclusions and recommendations.

7.2 The Experimental MAGALY Model

In contrast to the Phacetie model, the Magaly model, contains a whole spider with its twenty-five control rods. The guide plates used to guide the control rods are truly similar

to the ones used in the real nuclear power plant, as depicted in figure 7.1. In figure 7.2 the control rod guide assembly is depicted showing the skeleton of the assembly and all the guide plates.

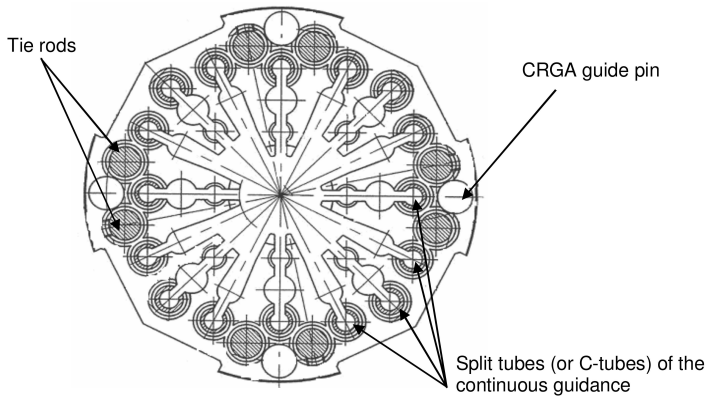


Figure 7.1: Top view of a guide plate in the experimental Magaly model. Ref [14].

7.3 The Numerical Magaly model

In the numerical representation of the Magaly model, the spider with the control rods, is modelled by two lines of elements. One of these two lines represents one control rod, and the other represents the other twenty-four control rods. Modelling all the twenty-five control rods connected to the spider separately cannot be done due to restrictions in computational speed. As in the numerical Phacetie model, the water in between the two tubes is not modelled, so the only interaction modelled between the single control rod and the representation of the twenty-four others is through the spider. The specifications of the model are obtained from a function in Code_Aster, with which 'emergency drop down' of the control rods is simulated. This function, `MACR_REPTIL_CALC`, obtains in turn the data from a database on different control rod designs. The function should be able to build the mesh of the control rods and the fuel assembly, calculate the deformations and eigen modes, calculate the depth of the control rod in the fuel assembly and the time it takes to drop down the control rod into the fuel assembly in case of an emergency. For the Magaly model only the mesh is used from this function. This mesh is depicted in figure 7.3
The point forces modelling the fluid force are applied on the same positions from the bottom of the tube for the Magaly model as where they were obtained on the Phacetie model.

7.4 Applying the Identified Forces

To apply the earlier identified forces, they must be extracted from the spectral density matrix $\mathbf{G}_{ff}$. This is done by using a Cholesky decomposition of the spectral density

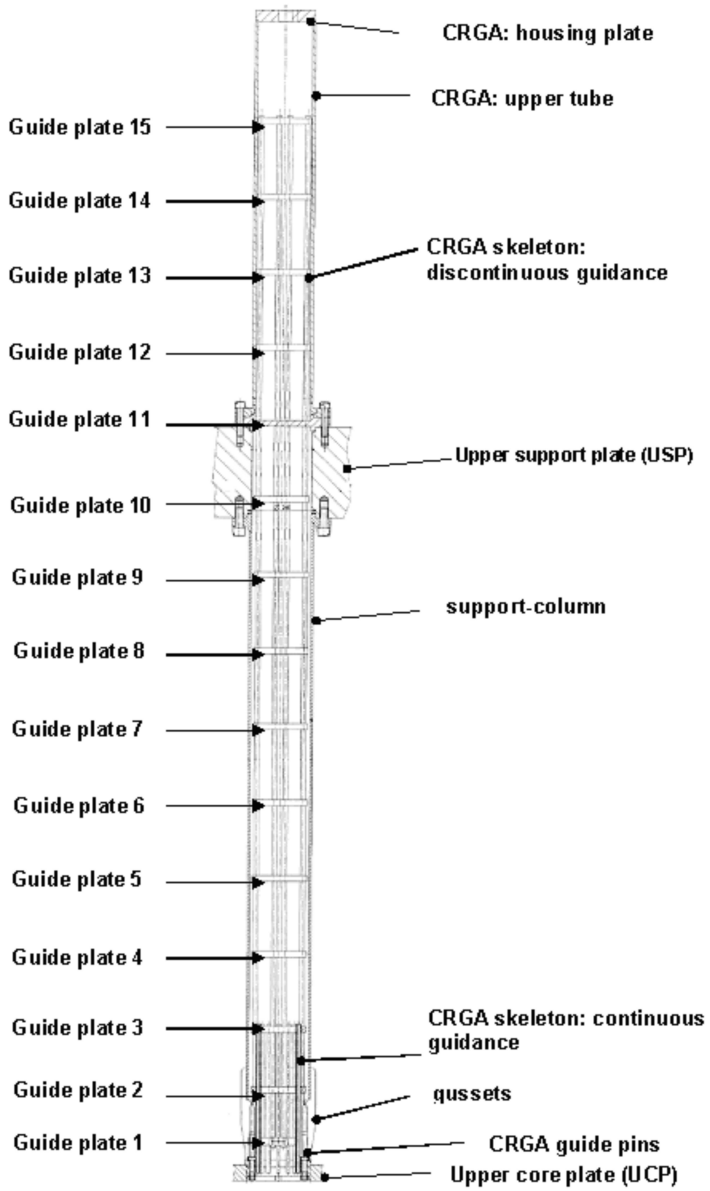


Figure 7.2: Overview of the control rod guide assembly with the guide plates in the experimental Magaly model. Ref [14].

matrix, because

$$\mathbf{G}_{ff}(j\omega) = \mathbf{F}(j\omega)\,\mathbf{F}^{\mathrm{H}}(j\omega) \tag{7.1}$$

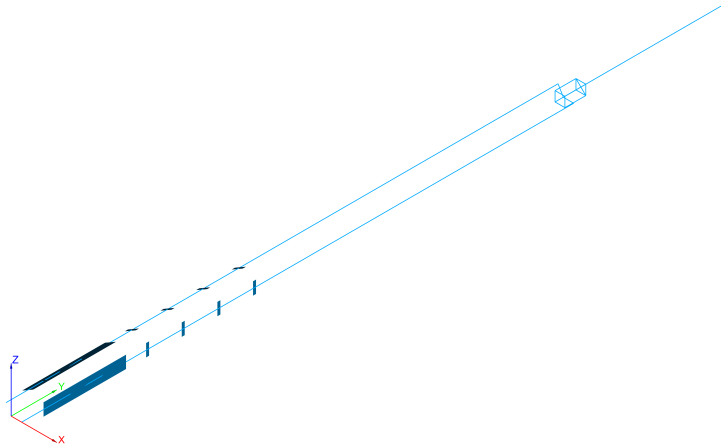


Figure 7.3: The mesh of the numerical Magaly model showing the two lines of elements, one modelling one control rod and the other one modelling the other twenty-four control rods.

Using $\mathbf{F}(j\omega)$ the force vector in the time domain can be simulated

$$\mathbf{f}(t) = \sqrt{\Delta\omega} \cdot \Re \left\{ \sum_{i=0}^{k-1} \mathbf{F}(j\omega_i) \boldsymbol{\chi}_{(i)} e^{j\omega_i t} \right\} \tag{7.2}$$

where $\Delta\omega$ is the frequency step and k is the number of steps in the frequency domain. The vector $\boldsymbol{\chi}_{(i)}$, used here in simplified form, contains just a random phase-shift. It is defined as

$$\chi_p = e^{j\theta_p} \tag{7.3}$$

where θ_p is the random phase shift between 0 and 2π.
This calculation of a time-domain signal is implemented in the Code_Aster function `GENE_FONC_ALEA`. This function does not work so far and needs to be improved.

7.5 A Critical Note Regarding the Modelling of the Fluid Force

The fluid force is modelled as a number of point forces, giving the same modal forces as the real fluid forces does. If these point forces are transferred to another structure, having different mode-shapes, then the obtained modal forces on this structure will only be equal to the modal forces of a real fluid force exciting the structure, if the mode-shapes are exactly the same. Thus, the slightest difference between a mode-shape of the Magaly model and of the Phacetie model, will have a great influence on the modal force. When the modelled force obtained at the Phacetie model is applied to the Magaly model, the obtained modal force on the Magaly model will only be a similar to the modal force of the real fluid force, if the mode-shapes of the Magaly model are exactly the same as the mode-shapes of the Phacetie model. The mode-shapes will never be exactly the same.

For example, the modelling of the non-linear interaction of the vibrating tube with the guide plates in the Phacetie model, using the springs, has such a great influence on the mode-shape, that, when compared to the mode-shapes of the Magaly model without the springs, the mode-shapes to these two systems will never be the same. So

$$\boldsymbol{\phi}_{(i)}^{\mathrm{Ph}}\mathbf{f}^{\mathrm{Ph}} = \int_L \phi_i^{\mathrm{Ph}}(y)\, f^{\mathrm{Ph}}(y)\ \mathrm{d}y \tag{7.4}$$

but, because the mode-shapes are not the same

$$\boldsymbol{\phi}_{(i)}^{\mathrm{Ma}}\mathbf{f}^{\mathrm{Ph}} \neq \int_L \phi^{\mathrm{Ma}}(y)\, f^{\mathrm{Ma}}(y)\ \mathrm{d}y. \tag{7.5}$$

The fluid force should therefore not be modelled as a number of point forces, but a more physical representation should be tried. For example, using a sort of 'force-shape'-approach in which the fluid force $\mathbf{f}$ is modelled as a sum of different shapes, as

$$\mathbf{f}(j\omega) = \sum_{i=1}^{P} \boldsymbol{\gamma}_{(i)}\beta_i(j\omega) \tag{7.6}$$

where $\boldsymbol{\gamma}_{(i)}$ are fluid shapes, containing shapes based on measured pressure distributions and $\beta_i(j\omega)$ are the participation factors. Then the number of shapes, P, is limited by the number of variables which can be identified, similar to the number of point forces which are now being identified.

7.6 Comments, Conclusions & Recommendations

7.6.1 Comments

Applying the point forces on different mode-shapes than the ones on which they were obtained, will not give a desirable result. Therefore it should be reconsidered using point forces to model the fluid force. A better result can probably be obtained using a form of expansion of the fluid force in 'fluid'-shapes. On the other hand, constructing a time signal from the fluid force, modelled this way, might turn out to be impossible.

7.6.2 Conclusions

With the function `GENE_FONC_ALEA` not functioning properly and the copied specifications from the drop down model not showing the desired behaviour, the validation of the identified fluid forces could so far not be done.

7.6.3 Recommendations

Logically, a recommendation is to get `GENE_FONC_ALEA` to work, obtaining a better numerical representation of the Magaly model to be able to finish this research and validate the identified fluid forces.
On a wider scope, the fluid force should be modelled differently and a method should be created to impose a more continuously modelled fluid forces on the numerical model.

Chapter 8

Conclusions & Recommendations

In this thesis a method to identify a fluid force inducing a vibration to the control rods of a nuclear power plant was researched. This method was initiated by Bodel [1] and that study was taken as a guideline through the process. During the research the method was compared with new insights and some important observations were made.
This chapter is split into four parts: Section 8.1 contains conclusions regarding the initial assignment, visualised in figure 1.3 in the Introduction. In section 8.2 considerations are described with respect to the observations made during the research.
Finally, some recommendations are made for future research in section 8.3 and for further improvement of Code_Aster concerning this study, section 8.4 gives a list of functions which should be improved.

8.1 Conclusions on the Thesis Assignment

In section 1.2 of the Introduction, four subjects were introduced, which were to be tested versus the existing study by Bodel. Except for the analysis being done in two directions simultaneously, non of the new concepts is chosen over the concepts already used in the study by Bodel, as depicted in figure 8.1.

- The 3D volume-element model is chosen over the 2D shell-element model. The two numerical models are quite similar according to the MAC-table between their eigen mode-shapes. The 2D shell-element model is slightly faster in computational time speaking and the 3D model gives slightly better MAC-numbers when they are calculated without using a mass matrix as a weighting matrix. The expanded measurements also needed to be checked without the mass matrix, because MAC-tables for the measurements in strain where made using Python and in Python it is not possible to handle the mass matrix. Therefore the 3D volume element matrix is chosen.

- In this thesis eigen mode-shape spaces and static shape spaces were compared when used as a expansion space. The eigen mode-shapes spaces were chosen, mainly

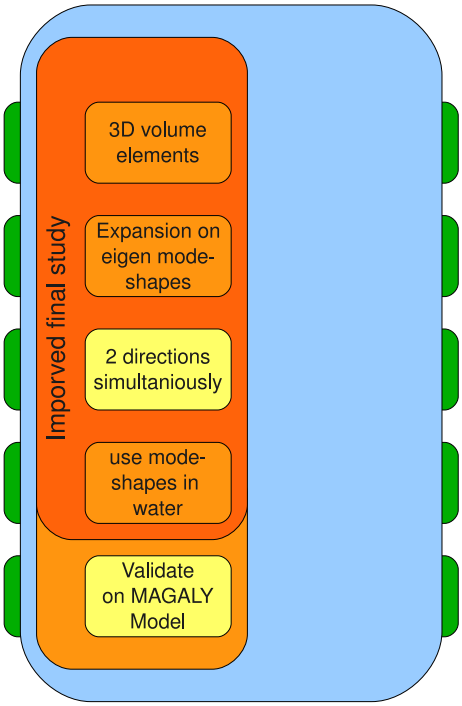


Figure 8.1: Overview of subjects which remain in the final study

because Code_Aster is not able to use static shape spaces in strain. And with the measurements done in water, and thus in strain, static shapes cannot be used for now.

- Doing the analysis in two directions simultaneously has shown to be an improvement. This way the slightly 'twisted' mode-shapes in water can be used in 3D and are not flattened to 2D mode-shapes as what was done in the study by Bodel [1].
- The original method, method I, gives a far better result when the PSDs of the strain measurements are re-calculated. Also the identified force PSDs show a better resembles with the PSDs in pressure obtained from literature. This is due to the fact that the mode-shapes in water, mass normalised and used in method I, describe the system a lot better than the re-normalised mode-shapes in air, used in method II.
- The final validation of the general method on the Magaly model has not yet been accomplished. But before that should be done, the modelling of the fluid force and

the general method should be reconsidered as explained in section 8.2.

8.2 Considerations & Observations

During the research several considerations regarding the fundamental methodology the study emerged as well as practical issues which had great implications on the quality of the study.

- First of all the observation that the obtained experimental mode-shapes were not well mass normalised. Efforts to mass normalise the FRF-data using ME-scope or with MATLAB SDTools had no desired result. The problems with the measurement are probably due to false calibration for some of the sensors. First, it must be researched what went wrong, and then the measurements need to be done again. The mass normalisation should be done using the measurement system or by using the MATLAB SDTools on the raw FRF-data. The practical measure, to 'hand'-normalise the mode-shapes is of a major influence on the quality of the research. It was done to be able to continue the study, but the results, the identified point forces, should therefore not be trusted.

- Secondly, the fluid force modelled as a number of point forces can not model the fluid force sufficiently well to be transferable to other models. The slightest difference between a mode-shape of the model on which the point forces are identified and the model on which they are applied, will have a great influence on the obtained modal force and with the modal force on the resulting vibration of the model. Application of the point forces will therefore not result in a similar vibration as when the model is excited with the true fluid force. Other methods to model the fluid force should therefore be researched.

- And third, the problem pointed out in chapter 2 is a fundamental problem in this method of identifying the fluid force. When Polymax does not has a measurement of the exciting force, the input signal reduces to the white noise signal it normally uses at the edges of the frequency range under consideration. Using only this random signal in time and space, the finally identified force will be this noise signal, if it had not been for the regularisational efforts. Due to this regularisation the slope at the high frequency range is imposed, forcing the signal to look like the obtain PSDs from literature.

8.3 Recommendations Considering the Research

These recommendations consist of a point wise summation of subjects, which can be look into during further research.

- When the cause of the problems regarding the normalisation of the measurements are understood, the measurements should be done again, to obtain well mass normalised experiments.

- As explained in chapter 5, using a two step expansion could maybe improve the expansion considerably. First, the numerical eigen mode-shapes in the frequency range of operation are used as an expansion space. And then the static shapes obtained using the sensor positions as DoFs, are used to expand the remaining differences. This combination of static-shapes and eigen mode-shapes as an expansion space may improve the expansion.
- The numerical representation of the Magaly model uses parameters of the function to simulate emergency drop-down in the nuclear power plant. Using these parameters for the model to simulate vibration in operation has so far not given the desired results. The numerical representation of the Magaly model should be reconsidered.
- In a wider scope, other paths through the possibilities in figure 1.5 can be considered. A more numerical approach could avoid some of the problems encounter during this research, for example regarding the modelled fluid force or with respect to the problem with Polymax. If a numerical model of the system in water can be obtained by parameter updating, the spectral density matrices can be applied to this numerical model and then there is no need to obtain the system using operational modal analysis.

8.4 Code_Aster Function Improvements

- `NORM_MODE`: This function mass normalises a specified result, using the modal mass within the result. It can now only process results in displacements and should be updated so it can also be used to normalise results in strain.
- `PROJ_MESU_MODAL`: This function calculates the participation factors for an expansion. It can operate on both strain and displacements, but not simultaneously.
- `CALC_ESSAI`: This routine is a graphical user interface callable from Code_Aster to do numerous operations. One of them is the force identification, making use of the theory explained in this thesis. It can only operate mode-shapes in displacement and should be updated to also be able to use mode-shapes in strain.
- `GENE_FONC_ALEA`: Function to calculate a time domain signal from a spectral density matrix in the frequency-domain. This function does not work properly and should be improved.

Appendices

Appendix A

Theory

A.1 Continuous vs. Discrete Modal Analysis

A.1.1 Introduction

In this chapter a continuous and a discrete modal analysis for a undamped beam will be done simultaneously. The analysis will be done without damping because the damping in this study will only be used to experimentally obtain the modal parameters. The full discrete analysis can be found in [9] and the full continuous analysis can be found in [3], [6] or [7].
By showing the beam theory in both continuous and discrete, the similarities and differences can be shown in a clear way. It gives more inside to the explanation in Appendix B.2 describing the way to test the mass normalisation of the mode-shapes of the tube. Next to that, it gives insight in the difference between the strain-field along the y-direction of the rod for continues (the double derivative of the displacement field with respect to the coordinate along the rod), in comparison to the normal mode-shape vector in displacement in the discrete version of the stiffness term.
The end of this analysis, can be used to underline the conclusions posted in the study. It can be clearly seen from the continuous analysis that modal forces are strongly based on the mode-shapes and therefore the fluid force modelled by a discrete amount of point forces is not transferable between different models. In the remainder of this appendix, the continuous equations will be posted on the left and the discrete on the right hand side.

A.1.2 Free Vibration of the Undamped Beam

As a start the equation of motion is stated for both continuous and discrete

$$\rho A\ddot{X}\left(y,t\right)\,\mathrm{d}y + EI\frac{\partial^4}{\partial y^4}X\left(y,t\right)\,\mathrm{d}y = 0 \qquad\qquad \mathbf{M}\ddot{\mathbf{x}}\left(t\right) + \mathbf{K}\mathbf{x}\left(t\right) = 0 \tag{A.1}$$

By using separation of variables, the displacement-field $X(y,t)$ and displacement-vector $\mathbf{x}(t)$, can be written as

$$X(y,t) = \phi(y)\,\nu(t) \qquad \mathbf{x}(t) = \boldsymbol{\phi}\,\nu(t) \tag{A.2}$$

Substitute and rewrite the equations, to find the well known

$$\rho A \frac{\ddot{\nu}(t)}{\nu(t)} + EI\frac{\phi''''(y)}{\phi(y)} = 0 \qquad \left(\mathbf{M}\frac{\ddot{\nu}(t)}{\nu(t)} + \mathbf{K}\right)\boldsymbol{\phi} = 0 \tag{A.3}$$

For a quasi-static solution $\frac{\ddot{\nu(t)}}{\nu(t)} = -\omega^2$ and is constant in time. And $\frac{\phi''''(y)}{\phi(y)} = \mu^4$ is constant in space. The system on the right hand side consists of n linear homogeneous equations which admits a non-trivial solution $\boldsymbol{\phi}_{(i)}$.

$$\begin{aligned} \omega_i^2\nu(t) + \ddot{\nu}(t) = 0 \qquad & \left(-\omega_i^2\mathbf{M} + \mathbf{K}\right)\boldsymbol{\phi}_{(i)} = 0 \\ -\mu^4\phi(y) + \phi''''(y) = 0 & \end{aligned} \tag{A.4}$$

Where the continuous equations have a solution in which the constants c_j are obtained from initial conditions and boundary conditions for the continuous analysis. On the discrete side, these conditions are already accounted for in the deduction of the mass and stiffness matrix, so the natural frequencies and mode-shapes are valued directly by the roots of the algebraic equation from the determinant.

$$\begin{aligned} \nu(t) &= c_0\sin\omega t + c_1\cos\omega t \\ \phi(y) &= C_{2i}\sin\mu y + c_3\cos\mu y\ldots \\ &\quad \cdots + c_4\sinh\mu y + c_5\cosh\mu y \end{aligned} \qquad \begin{aligned} &\det\left(-\omega_i^2\mathbf{M} + \mathbf{K}\right) = 0 \\ &\text{and } \boldsymbol{\phi}_{(i)} \text{ the respective eigen vector.} \end{aligned} \tag{A.5}$$

with

$$\mu^4 = \frac{\rho A}{EI}\omega^2$$

There are multiple solution possible for both continuous as well as for discrete. For continuous there are an infinite number are possibilities, for the discrete model there are n, the number of DoF, possibilities. The final undamped free vibration solutions can be written as

$$\begin{aligned} X_i(y,t) &= \phi_i(y)\,\nu_i(t) & \mathbf{x}_{(i)}(t) &= \boldsymbol{\phi}_{(i)}\nu_i(t) \\ \text{for } i &= 1,2,3,\ldots,\infty & \text{for } i &= 1,2,3,\ldots,n \end{aligned} \tag{A.6}$$

A.1.3 Orthogonality of the Eigen Mode-shapes

When the final solutions are substituted in equation A.1, an infinite number for the continuous and n for the discrete equations can be stated (leaving out the dependencies

on y and t for clarity)

$$-\omega_i^2 \rho A \phi_i \nu_i \, \mathrm{d}y + EI \frac{\partial^4}{\partial y^4} \phi_i \nu_i \, \mathrm{d}y = 0 \qquad -\omega_i^2 \mathbf{M} \boldsymbol{\phi}_{(i)} \nu_i + \mathbf{K} \boldsymbol{\phi}_{(i)} \nu_i = 0$$
$$\text{for } i = 1, 2, 3, \ldots, \infty \qquad \text{for } i = 1, 2, 3, \ldots, n \tag{A.7}$$

First dividing out ν_i and then pre-multiplying each eigen mode-shape or eigen mode-vector by a mode-shape ϕ_j or mode-vector $\boldsymbol{\phi}_{(j)}$ respectively, and integrating over the length of the beam for the continuous analysis (for the discrete analysis this integration is basically already done by the vector multiplication operation). Then pairs can be formed as

$$-\omega_i^2 \rho A \int_L \phi_j \phi_i \mathrm{d}y + EI \int_L \phi_j \frac{\partial^4}{\partial y^4} \phi_i \mathrm{d}y = 0 \qquad -\omega_i^2 \boldsymbol{\phi}_{(j)}^{\mathrm{T}} \mathbf{M} \boldsymbol{\phi}_{(i)} + \boldsymbol{\phi}_{(j)}^{\mathrm{T}} \mathbf{K} \boldsymbol{\phi}_{(i)} = 0$$

$$-\omega_j^2 \rho A \int_L \phi_i \phi_j \mathrm{d}y + EI \int_L \phi_i \frac{\partial^4}{\partial y^4} \phi_j \mathrm{d}y = 0 \qquad -\omega_j^2 \boldsymbol{\phi}_{(i)}^{\mathrm{T}} \mathbf{M} \boldsymbol{\phi}_{(j)} + \boldsymbol{\phi}_{(i)}^{\mathrm{T}} \mathbf{K} \boldsymbol{\phi}_{(j)} = 0 \tag{A.8}$$

The aim is to subtract the equations and show orthogonality, but first, for the continuous analysis, it needs to be shown that the stiffness terms of the two equations are the same. Using integration by parts

$$\int_L \phi_i \frac{\partial^4 \phi_j}{\partial y^4} \, \mathrm{d}y = \left[\phi \frac{\partial^3 \phi_j}{\partial y^3} \right]_0^L - \int_L \frac{\partial \phi_i}{\partial y} \frac{\partial^3 \phi_j}{\partial y^3} \, \mathrm{d}y \tag{A.9}$$

$$= \left[\phi_i \frac{\partial^3 \phi_j}{\partial y^3} \right]_0^L - \left[\frac{\partial \phi_i}{\partial y} \frac{\partial^2 \phi_j}{\partial y^2} \right]_0^L - \int_L \frac{\partial^2 \phi_i}{\partial y^2} \frac{\partial^2 \phi_j}{\partial y^2} \, \mathrm{d}y \tag{A.10}$$

and this, with any boundary condition at $y = 0$ and $y = L$ (fixed, free, simple and clamped with a vertical degree of freedom, but no springs) makes the first and second term of the right-hand part of the expression zero. Thus

$$\int_L \phi_i \frac{\partial^4 \phi_j}{\partial y^4} \, \mathrm{d}y = \int_L \frac{\partial^2 \phi_i}{\partial y^2} \frac{\partial^2 \phi_j}{\partial y^2} \, \mathrm{d}y = \int_L \frac{\partial^2 \phi_j}{\partial y^2} \frac{\partial^2 \phi_i}{\partial y^2} \, \mathrm{d}y = \int_L \phi_j \frac{\partial^4 \phi_i}{\partial y^4} \, \mathrm{d}y \tag{A.11}$$

Now the counterparts in equation A.8 can be subtracted showing

$$\left(\omega_j^2 - \omega_i^2\right) \rho A \int_L \phi_j \phi_i \, \mathrm{d}y = 0 \qquad \left(\omega_j^2 - \omega_i^2\right) \boldsymbol{\phi}_{(j)}^{\mathrm{T}} \mathbf{M} \boldsymbol{\phi}_{(i)} = 0 \tag{A.12}$$

Which shows that for two different mode-shapes, with eigen frequencies well apart, the mode-shapes must be orthogonal for the continues, and mass and stiffness matrix orthogonal, for the discrete analysis. And also the eigen mode-shapes in strain $\psi_i = R \frac{\partial^2 \phi_i}{\partial y^2}$, can be shown to be orthogonal. By rewriting equation A.8 a similar result is shown for non-multiple eigen frequencies

$$\left(\frac{1}{\omega_j^2} - \frac{1}{\omega_i^2} \right) \frac{EI}{R^2} \int_L \psi_j \psi_i \, \mathrm{d}y = 0. \tag{A.13}$$

A.1.4 Modal Mass and Mass Normalisation of the Mode-Shapes

The modal mass μ and the modal stiffness κ of a specific mode i is defined as

$$\mu_i = \rho A \int_L \phi_i^2 \, \mathrm{d}y \qquad \mu_i = \boldsymbol{\phi}_{(i)}^{\mathrm{T}} \mathbf{M} \boldsymbol{\phi}_{(i)}$$
$$\kappa_i = \frac{EI}{R^2} \int_L \psi_i^2 \, \mathrm{d}y \qquad \kappa_i = \boldsymbol{\phi}_{(i)}^{\mathrm{T}} \mathbf{K} \boldsymbol{\phi}_{(i)}. \tag{A.14}$$

As a mode-shape is just a shape and has no amplitude, it can be normalised, giving it an amplitude. It can be normalised in different ways (see appendix A.2). Normalising the mode-shapes with a unity modal mass, is called mass normalising and is most commonly used.

$$\overline{\phi_i} = \frac{\phi_i}{\sqrt{\mu_i}} \qquad \overline{\boldsymbol{\phi}}_{(i)} = \frac{1}{\sqrt{\mu_i}} \boldsymbol{\phi}_{(i)}$$
$$\rho A \int_L \overline{\phi_i}^2 \, \mathrm{d}y = 1 \qquad \overline{\boldsymbol{\phi}}_{(i)}^{\mathrm{T}} \mathbf{M} \overline{\boldsymbol{\phi}}_{(i)} = 1 \tag{A.15}$$

The modal stiffness κ_i is then equal to ω_i^2 which follows from the Rayleigh equation

$$\omega_i^2 = \frac{\frac{EI}{R^2} \int_L \psi_i^2 \, \mathrm{d}y}{\rho A \int_L \phi_i^2 \, \mathrm{d}y} = \frac{\kappa_i}{\mu_i} \qquad \omega_i^2 = \frac{\boldsymbol{\phi}_{(i)}^{\mathrm{T}} \mathbf{K} \boldsymbol{\phi}_{(i)}}{\boldsymbol{\phi}_{(i)}^{\mathrm{T}} \mathbf{M} \boldsymbol{\phi}_{(i)}} = \frac{\kappa_i}{\mu_i} \tag{A.16}$$

A.1.5 Forced Vibration of the Undamped Beam

The equation of motion for the forced, undamped, continuous and discrete system are stated as

$$\rho A \ddot{X}(y,t) \, \mathrm{d}y + EI \frac{\partial^4}{\partial y^4} X(y,t) \, \mathrm{d}y = f(y,t) \, \mathrm{d}y \qquad \mathbf{M}\ddot{\mathbf{x}}(t) + \mathbf{K}\mathbf{x}(t) = \mathbf{f}(t) \tag{A.17}$$

The displacement-field $X(y,t)$ and displacement-vector $\mathbf{x}(t)$ can be written as a sum of the eigen mode-shapes $\phi_i(y)$ and, $\boldsymbol{\phi}_{(i)}$ respectively, and the participation factors $\nu_i(t)$ as

$$X(y,t) = \sum_{i=1}^{\infty} \phi_i(y) \, \nu_i(t) \qquad \mathbf{x}(t) = \sum_{i=1}^{n} \boldsymbol{\phi}_{(i)} \nu_i(t) = \boldsymbol{\Phi} \boldsymbol{\nu}(t) \tag{A.18}$$
$$\text{for } i = 1, 2, 3, \ldots, \infty \qquad \text{for } i = 1, 2, 3, \ldots, n$$
$$\boldsymbol{\Phi} = \left[\boldsymbol{\phi}_{(1)} \; \boldsymbol{\phi}_{(2)} \; \ldots \; \boldsymbol{\phi}_{(n)} \right]$$
$$\boldsymbol{\nu}(t) = \left[\nu_1(t) \; \nu_2(t) \; \ldots \; \nu_n(t) \right]^{\mathrm{T}}$$

By substituting equation A.18 and pre-multiplied with the different eigen mode-shapes (and integrated for the continuous analysis) and using equation A.11, equation A.17

becomes (again, leaving out the dependencies for clarity)

$$\rho A \int_L \phi_i^2 \,\mathrm{d}y\, \ddot{\nu}_i + \frac{EI}{R^2} \int \psi_i^2 \,\mathrm{d}y\, \nu_i = \int_L \phi_i f \,\mathrm{d}y \qquad \boldsymbol{\phi}_{(i)}^{\mathrm{T}} \mathbf{M} \boldsymbol{\phi}_{(i)} \ddot{\nu}_i + \boldsymbol{\phi}_{(i)}^{\mathrm{T}} \mathbf{K} \boldsymbol{\phi}_{(i)} \nu_i = \boldsymbol{\phi}_{(i)}^{\mathrm{T}} \mathbf{f}$$

for $i = 1, 2, 3, \ldots, \infty$ for $i = 1, 2, 3, \ldots, n$ (A.19)

Now dividing by the modal mass $\mu_i = \rho A \int_L \phi_i^2 (y) \,\mathrm{d}y$ and using that for the steady state response $\ddot{\nu}(t) = -\omega^2 \nu(t)$ gives

$$\nu_i(t) = \frac{\int_L \phi_i(y) f(y,t) \,\mathrm{d}y}{\mu_i \left(-\omega^2 + \omega_i^2\right)} \qquad \nu_i(t) = \frac{\boldsymbol{\phi}_{(i)}^{\mathrm{T}} \mathbf{f}(t)}{\mu_i \left(-\omega^2 + \omega_i^2\right)}$$

for $i = 1, 2, 3, \ldots, \infty$ for $i = 1, 2, 3, \ldots, n$ (A.20)

And so, equation A.18 becomes

$$X(y,t) = \sum_{i=1}^{\infty} \phi_i(y) \frac{\int_L \phi_i(y) f(y,t) \,\mathrm{d}y}{\mu_i \left(-\omega^2 + \omega_i^2\right)} \qquad \mathbf{x}(t) = \sum_{i=1}^{n} \boldsymbol{\phi}_{(i)} \frac{\boldsymbol{\phi}_{(i)}^{\mathrm{T}} \mathbf{f}(t)}{\mu_i \left(-\omega^2 + \omega_i^2\right)}$$

in matrix form

$$\mathbf{x}(t) = \overline{\boldsymbol{\Phi}} \mathbf{Z}^{-1} \overline{\boldsymbol{\Phi}}^{\mathrm{T}} \mathbf{f}(t)$$

where

$$\mathbf{Z}(j\omega) = \left(-\omega^2 \mathbf{I} + \left[\backslash \omega_{i\backslash}^2\right]\right)$$

with

$$\left[\backslash \omega_{i\backslash}^2\right] = \begin{bmatrix} \omega_1^2 & & & \varnothing \\ & \omega_2^2 & & \\ & & \ddots & \\ \varnothing & & & \omega_n^2 \end{bmatrix}$$

thus

$$\mathbf{Z}^{-1}(j\omega) = \begin{bmatrix} \frac{1}{-\omega^2+\omega_1^2} & & & \varnothing \\ & \frac{1}{-\omega^2+\omega_2^2} & & \\ & & \ddots & \\ \varnothing & & & \frac{1}{-\omega^2+\omega_n^2} \end{bmatrix}$$

(A.21)

A.2 The Frequency Response Function

A.2.1 Introduction

The frequency response function (FRF) is the transfer function evaluated along the frequency, $j\omega$, axis. The transfer signal can be measured and converted into the frequency domain. In this appendix the marginal basics of the theory to obtain the frequency response function from measurements is described, to show, (1) on what, the different techniques to obtain the modal parameters are based, (2) how the mode-shapes can be real valued even though damping is taken into account, (3) why the mode-shapes obtained by the algorithm of the measurement system, can be mass normalised when the input signal is measured, and why, without the input signal, they cannot, (4) what kind of different normalisations there are and which should be used in this case.
The small theory shown here is mainly based on [9] and this work is also recommended for the full analysis.
The transfer function in the time-domain for a steady state response is shown in appendix A.1, equation A.21

$$\mathbf{x}(t) = \mathbf{H}(j\omega)\,\mathbf{f}(t) \tag{A.22}$$

$$\mathbf{H}(j\omega) = \sum_{i=1}^{n} \frac{\boldsymbol{\phi}_{(i)}\boldsymbol{\phi}_{(i)}^{\mathrm{T}}}{\mu_i\left(-\omega^2 + \omega_i^2\right)} \tag{A.23}$$

this basically already shows the transfer function along the frequency axis, but without damoing. Still, with this analysis a similar result should be obtained.

Damped System

The damping, which is present in every real structure, is used to obtain the mode-shapes from the measured frequency response function, thus the analysis needs to be done taken account for this damping. The general case is started by showing the damped system equations

$$\mathbf{M}\ddot{\mathbf{x}}(t) + \mathbf{C}\dot{\mathbf{x}}(t) + \mathbf{K}\mathbf{x}(t) = \mathbf{f}(t) \tag{A.24}$$

Using Laplace transformation, variable s, this becomes

$$\left(s^2\mathbf{M} + s\mathbf{C} + \mathbf{K}\right)\mathbf{x}(s) = \mathbf{f}(s) \tag{A.25}$$

$$\mathbf{Z}(s)\,\mathbf{x}(s) = \mathbf{f}(s) \tag{A.26}$$

where $\mathbf{Z}(s)$ is called the *dynamic stiffness matrix*. The *transfer function matrix*, $\mathbf{H}(s)$, is the inverse of $\mathbf{Z}(s)$, thus

$$\mathbf{H}(s) = \mathbf{Z}(s)^{-1} = \frac{\mathrm{adj}\left(\mathbf{Z}(s)\right)}{|\mathbf{Z}(s)|} \tag{A.27}$$

where $|\mathbf{Z}(s)|$ is the determinant of $\mathbf{Z}(s)$, the system characteristic equation. To calculate the poles, it is put in state space

$$(s\mathbf{A} + \mathbf{B})\,\mathbf{x}'(s) = \mathbf{f}'(s) \tag{A.28}$$

with

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{M} \\ \mathbf{M} & \mathbf{C} \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} -\mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{K} \end{bmatrix} \qquad \mathbf{x}'(s) = \begin{bmatrix} s\,\mathbf{x}(s) \\ \mathbf{x}(s) \end{bmatrix} \qquad \mathbf{f}'(s) = \begin{bmatrix} \mathbf{0} \\ \mathbf{f}(s) \end{bmatrix}. \quad \text{(A.29)}$$

By evaluating the homogeneous equation, $\mathbf{f}(s) = \mathbf{0}$, thus $\mathbf{f}'(s) = \mathbf{0}$, the poles of the characteristic equation can be identified

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & & & & & \\ & \ddots & & & \mathbf{0} & \\ & & \lambda_N & & & \\ & & & \lambda_1^* & & \\ & \mathbf{0} & & & \ddots & \\ & & & & & \lambda_N^* \end{bmatrix} \quad \text{(A.30)}$$

where $\lambda_i = \sigma_i + j\omega_{di}$ and $\lambda_i^* = \sigma_i - j\omega_{\mathrm{d}i}$, with σ_i is the *damping factor* and $\omega_{\mathrm{d}i}$ is the *damped natural frequency* of the i^{th} mode-shape. Equation A.27 can be rewritten as

$$\mathbf{H}(s) = \frac{\mathrm{adj}\left[\mathbf{Z}(s)\right]}{\prod_{i=1}^{N} C_1 (s-\lambda_i)(s-\lambda_i^*)} = \frac{\mathrm{adj}\left[\mathbf{Z}(s)\right]}{\prod_{i=1}^{2N} C_1 (s-\lambda_i)} \quad \text{(A.31)}$$

where C_1 is a constant and $\lambda_N + k = \lambda_k^*$ for $k = 1, 2, 3 \ldots, N$. Applying fraction expansion gives

$$\mathbf{H}(s) = \sum_{i=1}^{N} \left(\frac{\mathbf{R}_{(i)}}{(s-\lambda_i)} + \frac{\mathbf{R}_{(i)}^*}{(s-\lambda_i^*)} \right) \quad \text{(A.32)}$$

by multiplying the total equation subsequently by each $(s - \lambda_i)$ it is shown that for each corresponding $s = \lambda_i$ the *residue*, $\mathbf{R}_{(i)}$, is equal to

$$\mathbf{R}_{(i)} = \left(\mathbf{H}(s)(s-\lambda_i)\right)|_{s=\lambda_i} \quad \text{(A.33)}$$

By substitution the residues are

$$\mathbf{R}_{(i)} = \frac{\mathrm{adj}\left[\mathbf{Z}(\lambda_i)\right]}{\prod_{k=1,k\neq i}^{2N} C_1 (\lambda_i - \lambda_k)} \quad \text{(A.34)}$$

A closer look on $\mathrm{adj}\left(\mathbf{Z}(s=\lambda_i)\right)$ will clarify the relation to $\boldsymbol{\phi}_{(i)}$ because, from equation A.27

$$\mathbf{Z}(s) \cdot \mathrm{adj}\left(\mathbf{Z}(s)\right) = |\mathbf{Z}(s)| \cdot \mathbf{I} \quad \text{(A.35)}$$

evaluated on $s = \lambda_i$ gives

$$\mathbf{Z}(\lambda_i) \cdot \mathrm{adj}\left(\mathbf{Z}(\lambda_i)\right) = \mathbf{0} \quad \text{(A.36)}$$

therefore $\mathrm{adj}\left(\mathbf{Z}(\lambda_i)\right)$ must be build from different scalars times the eigenvector, $\boldsymbol{\phi}_{(i)}$, corresponding with λ_i. And because the mass, damping and stiffness matrix in the dynamic stiffness matrix, $\mathbf{Z}(s)$, are all symmetric (when no gyroscopic effects are present), the adjoint matrix $\mathrm{adj}\left(\mathbf{Z}(s)\right)$ must also be symmetric, thus

$$\mathrm{adj}\left(\mathbf{Z}(\lambda_i)\right) = C_{2i}\boldsymbol{\phi}_{(i)}\boldsymbol{\phi}_{(i)}^{\mathrm{T}} = C_{2i}\begin{bmatrix} \phi_1\phi_1 & \phi_1\phi_2 & \cdots & \phi_1\phi_N \\ \phi_2\phi_1 & \phi_2\phi_2 & \cdots & \phi_2\phi_N \\ \vdots & \vdots & \ddots & \vdots \\ \phi_N\phi_1 & \phi_N\phi_2 & \cdots & \phi_N\phi_N \end{bmatrix} \quad \text{(A.37)}$$

Now substituting equation A.37 in equation A.34 and equation A.34 in equation A.32 gives

$$\mathbf{H}(p) = \sum_{i=1}^{N} \left(\frac{C_{3i}\boldsymbol{\phi}_{(i)}\boldsymbol{\phi}_{(i)}^{\mathrm{T}}}{(p-\lambda_i)} + \frac{C_{3i}^{*}\boldsymbol{\phi}_{(i)}^{*}\boldsymbol{\phi}_{(i)}^{*\mathrm{T}}}{(p-\lambda_i^{*})} \right) \tag{A.38}$$

where

$$C_{3i} = \frac{C_{2i}}{\prod_{i=1, i\neq k}^{2N} C_1(\lambda_i - \lambda_k)} \tag{A.39}$$

The transfer function evaluated along the frequency axis gives the frequency response function

$$\mathbf{H}(j\omega) = \sum_{i=1}^{N} \left(\frac{C_{3i}\boldsymbol{\phi}_{(i)}\boldsymbol{\phi}_{(i)}^{\mathrm{T}}}{(j\omega-\lambda_i)} + \frac{C_{3i}^{*}\boldsymbol{\phi}_{(i)}^{*}\boldsymbol{\phi}_{(i)}^{*\mathrm{T}}}{(j\omega-\lambda_i^{*})} \right) \tag{A.40}$$

Proportionally Damped System

For the real structure, the damping is coupling the mode-shapes, but in case of small damping, the system can be modelled with proportional damping and the mode-shapes can be decoupled making the eigen vectors real. The damping matrix $\mathbf{C}$ is proportional, thus

$$\mathbf{C} = C_4\mathbf{M} + C_5\mathbf{K} \tag{A.41}$$

Therefore, the system equations become

$$\left(s^2\mathbf{M} + C_4 s\mathbf{M} + C_5 s\mathbf{K} + \mathbf{K}\right)\mathbf{x}(s) = \mathbf{f}(s) \tag{A.42}$$

and thus considering the homogeneous equation

$$\left(\frac{s^2 + C_4 s}{1 + C_5 s}\mathbf{M} + \mathbf{K}\right)\mathbf{x}(s) = \mathbf{0} \tag{A.43}$$

This equations will have complex poles, λ_i, complying with the totally imaginary poles, $j\omega_i$, obtained in the analysis of the undamped system (not shown here)

$$\frac{\lambda_i^2 + C_4\lambda_i}{1 + C_5\lambda_i} = -\omega_i^2 \tag{A.44}$$

and will therefore have the same real eigenvectors $\boldsymbol{\phi}_{(i)}$ as the undamped system. (The obtained modes-shapes can be turned in the complex plane by the complex constant C_{3i} to become real valued.) Equation A.40 will therefore simplify to

$$\mathbf{H}(j\omega) = \sum_{i=1}^{N} \frac{j2\omega_{\mathrm{d}i}C_{3i}\boldsymbol{\phi}_{(i)}\boldsymbol{\phi}_{(i)}^{\mathrm{T}}}{-\omega^2 + j2\omega\omega_i\zeta_i + \omega_i^2} \tag{A.45}$$

where $\omega_i = \sqrt{\sigma_i^2 + \omega_{\mathrm{d}i}^2}$ and $\zeta_i = -\frac{\sigma_i}{\sqrt{\sigma_i^2+\omega_{\mathrm{d}i}^2}}$.

Modal Vector Scaling

Comparing equation A.32 with equation A.38 shows

$$\mathbf{R}_{(i)} = C_{3i}\boldsymbol{\phi}_{(i)}\boldsymbol{\phi}_{(i)}^{\mathrm{T}} \tag{A.46}$$

For element $r_{pq\,(i)}$ of matrix $\mathbf{R}_{(i)}$, with response DoF p and input DoF q

$$r_{pq\,(i)} = C_{3i}\phi_{p\,(i)}\phi_{q\,(i)} \tag{A.47}$$

By choosing the factor C_{3i} the mode-shapes are scaled, normalised. The mode-shapes can be normalised in different ways:

- Unity normalisation, in which the highest entry in the obtained mode-shape is scaled to one.
- Unit length normalisation, where entries are scale such, that the length of the eigen vector is unity.
- Mass normalisation, where the modal mass $\mu_i = \overline{\boldsymbol{\phi}}_{(i)}^{\mathrm{T}}\mathbf{M}^{\mathrm{ex}}\overline{\boldsymbol{\phi}}_{(i)} = 1$ is unity.

Comparing equation A.45 with equation A.23 shows that $C_{3i} = \frac{\mu_i}{j2\omega_{\mathrm{d}i}}$ to have a normalisation of the mode-shapes for which the analysis gives similar results as for the analysis done from a model prospective in appendix A.1. For the experimental modal analysis the mass matrix, $\mathbf{M}^{\mathrm{ex}}$, and therefore the modal masses are unknown. Therefore the mode-shapes are normalised for mass, having a modal mass of unity. Equation A.47 then becomes

$$r_{pq\,(i)} = \frac{1}{j2\omega_{\mathrm{d}i}}\overline{\phi}_{p\,(i)}\overline{\phi}_{q\,(i)} \tag{A.48}$$

Still, the mode-shapes entries cannot be decided upon for different input and output positions. By looking at the driving-point FRF residue, $r_{qq\,(i)}$, the mode-shape entries for the driving point can be obtained

$$\overline{\phi}_{q\,(i)} = \sqrt{j2\omega_{\mathrm{d}i}r_{qq\,(i)}} \tag{A.49}$$

and then also the other mode-shapes entries.

$$\overline{\phi}_{p\,(i)} = \frac{j2\omega_{\mathrm{d}i}r_{qq\,(i)}}{\overline{\phi}_{q\,(i)}} \tag{A.50}$$

Thus, when no drivingpoint FRF is present the modeshapes cannot be normalised. The final FRF is then described by

$$\mathbf{H}\left(j\omega\right) = \sum_{i=1}^{N}\frac{\overline{\boldsymbol{\phi}}_{(i)}\overline{\boldsymbol{\phi}}_{(i)}^{\mathrm{T}}}{-\omega^2 + j2\omega\omega_i\zeta_i + \omega_i^2} \tag{A.51}$$

A.3 Singular Value Decomposition and the Pseudo Inverse

A.3.1 Introduction

In this appendix the Moore-Penrose pseudo inverse is explained, as well as a way to obtain it using the singular value decomposition. It is the theory beneath the inversion of the FRF-matrix done in chapter 6. When the matrix is ill conditioned, like the FRF-matrix near resonance, the inverse becomes unstable. To overcome these instabilities, two regularisation filters will be presented. The Truncated Singular Value Decomposition and the Tikhonov filter as obtained from [11].

A.3.2 Pseudo Inverse

When searching for a solution for $\mathbf{Ax} = \mathbf{b}$ the inverse of $\mathbf{A}$ needs to be calculated (or other techniques can be applied). For full rank square matrices the inverse can be easily obtained, for example by

$$\mathbf{A}^{-1} = \frac{\text{adj}\,(\mathbf{A})}{|\mathbf{A}|} \tag{A.52}$$

but when the determinant is zero, which is the case for any non-square or non full rank matrix, the inverse cannot be calculated. However, there exist another technique that calculates the least square approximation, the Moore-Penrose pseudo inverse. The pseudo-inverse, $\mathbf{A}^+$, of a $m \times n$ matrix $\mathbf{A}$ is unique and satisfies all of the following criteria.

$$\mathbf{A}\mathbf{A}^+\mathbf{A} = \mathbf{A} \tag{A.53}$$

$$\mathbf{A}^+\mathbf{A}\mathbf{A}^+ = \mathbf{A}^+ \tag{A.54}$$

$$\left(\mathbf{A}\mathbf{A}^+\right)^{\mathrm{H}} = \mathbf{A}\mathbf{A}^+ \tag{A.55}$$

$$\left(\mathbf{A}^+\mathbf{A}\right)^{\mathrm{H}} = \mathbf{A}^+\mathbf{A} \tag{A.56}$$

If $\mathbf{A}$ consists of only non-complex numbers, the Hermitian transpose ($^{\mathrm{H}}$), becomes just an ordinary transpose ($^{\mathrm{T}}$). If $\mathbf{A}$ is a square, full rank matrix then

$$\mathbf{A}^+ = \mathbf{A}^{-1}. \tag{A.57}$$

This Pseudo inverse can be obtained by a reduced singular value decomposition, as explained in the next section.

A.3.3 Singular Value Decomposition

A $m \times n$ matrix $\mathbf{A}$ can be decomposed as

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{H}} \tag{A.58}$$

where $\mathbf{U}$ contains the left singular vectors of A, being orthonormal and spanning the column-space of $\mathbf{A}$ and $\mathbf{V}$ contains the right singular vectors of $\mathbf{A}$, being orthonormal and spanning the row-space of $\mathbf{A}$[2]. $\mathbf{\Sigma}$ is diagonal matrix consisting of the singular values

σ_i. These singular values are the square-root of the eigen values of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$.
The inverse of the matrix is

$$\mathbf{A}^{-1} = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^{\mathrm{H}} \tag{A.59}$$

if the matrix $\mathbf{A}$ is not of full rank, thus the rank r is smaller then n and m, $\mathbf{\Sigma}$ turns into

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{D} & 0 \\ 0 & 0 \end{bmatrix} \tag{A.60}$$

where $\mathbf{D}$ is an $r \times r$ matrix containing the singular values σ_i in descending order, then the inverse turns into the pseudo inverse

$$\mathbf{A}^{+} = \mathbf{V}_r\mathbf{D}^{-1}\mathbf{U}_r^{\mathrm{H}} \tag{A.61}$$

where $\mathbf{U}_r$ is $\begin{bmatrix}\mathbf{u}_1 \dots \mathbf{u}_r\end{bmatrix}$ and $\mathbf{V}_r$ is $\begin{bmatrix}\mathbf{v}_1 \dots \mathbf{v}_r\end{bmatrix}$. To show that this is the least square solution, the pseudo-inverse is used to find a optimal solution $\hat{\mathbf{x}}$, for $\mathbf{A}\mathbf{x} = \mathbf{b}$.

$$\hat{\mathbf{x}} = \mathbf{A}^{+}\mathbf{b} = \mathbf{V}_r\mathbf{D}^{-1}\mathbf{U}_r^{\mathrm{H}}\mathbf{b} \tag{A.62}$$

by multiplying $\hat{\mathbf{x}}$ by the singular value decomposition for $\mathbf{A}$

$$\mathbf{A} = \mathbf{U}_r\mathbf{D}\mathbf{V}_r^{\mathrm{H}} \tag{A.63}$$

it is shown that

$$\begin{aligned}
\mathbf{A}\hat{\mathbf{x}} &= \left(\mathbf{U}_r\mathbf{D}\mathbf{V}_r^{\mathrm{H}}\right)\left(\mathbf{V}_r\mathbf{D}^{-1}\mathbf{U}_r^{\mathrm{H}}\right)\mathbf{b} && && \text{(A.64)}\\
&= \mathbf{U}_r\mathbf{D}\mathbf{D}^{-1}\mathbf{U}_r^{\mathrm{H}}\mathbf{b} && \text{because } \mathbf{V}_r^{\mathrm{H}}\mathbf{V}_r = \mathbf{I}_r && \text{(A.65)}\\
&= \mathbf{U}_r\mathbf{U}_r^{\mathrm{H}}\mathbf{b} && \text{because } \mathbf{D}\mathbf{D}^{-1} = \mathbf{I} && \text{(A.66)}
\end{aligned}$$

And $\mathbf{U}\mathbf{U}^{\mathrm{H}}\mathbf{b}$ is the projection, $\hat{\mathbf{b}}$, of $\mathbf{b}$ on the space spanned by $\mathbf{U}$. The vector $\|\mathbf{b} - \hat{\mathbf{b}}\|$ is orthogonal to $\hat{\mathbf{b}}$ and thus is closest to the real solution $\mathbf{b}$. Thus $\hat{\mathbf{x}}$ is the least-square solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$.

A.3.4 Truncated Single Value Decomposition & Tikhonov Regularisation

When the matrix is ill-conditions, having a great discrepancy between the smallest and biggest singular value, the matrix should be subdue to some form of regularisation. Next to that, noise in measured signals can be counteracted using regularisation. If the $\mathbf{b}$ is a measured signal, it can be polluted with some measurement noise, $\mathbf{b}^{\mathrm{noise}}$. This noise then appears in the solution $\mathbf{x}$ as

$$\mathbf{x} = \mathbf{A}^{+}\mathbf{b}^{\mathrm{exact}} + \mathbf{A}^{+}\mathbf{b}^{\mathrm{noise}} \tag{A.67}$$

$$\mathbf{x} = \mathbf{A}^{+}\mathbf{b}^{\mathrm{exact}} + \sum_i \sigma^{-1}\mathbf{u}_i^{\mathrm{H}}b_i^{\mathrm{noise}}\mathbf{v}_i \tag{A.68}$$

where $\mathbf{u}_{(i)}$ is the i^{th} column of $\mathbf{U}$ and $\mathbf{v}_i$ is the i^{th} column of $\mathbf{V}$. Now it is easily seen that for relative small singular values, σ_i, the error is amplified relative to the desired exact

value of $\mathbf{x}$. By truncating the singular value matrix $\mathbf{\Sigma}$, for singular values smaller then a certain value, this amplification can be avoided. Using a filter $W_\varepsilon\left(\sigma_i^2\right)$

$$\mathbf{x} = \mathbf{A}^+_{\text{trunc}}\mathbf{b}^{\text{exact}} + \sum_i W_\varepsilon\left(\sigma_i^2\right)\sigma^{-1}\mathbf{u}_i^{\text{H}} b_i^{\text{noise}}\mathbf{v}_i \tag{A.69}$$

with

$$W_\varepsilon\left(\sigma_i^2\right) = \begin{cases} 1 & \text{if} \quad \sigma^2 > \varepsilon \\ 0 & \text{if} \quad \sigma^2 \leqslant \varepsilon \end{cases} \tag{A.70}$$

If the matrix to be inverted is dependant (on frequency for example), discontinuities can arise using this method of regularisation. Therefore a continues regularisation was developed, the Tikhonov regularisation

$$W_\alpha\left(\sigma_i^2\right) = \frac{\sigma_i}{\sigma_i^2 + \alpha} \tag{A.71}$$

transforming equation A.69 into

$$\mathbf{x} = \mathbf{A}^+_{\text{Tikh}}\mathbf{b}^{\text{exact}} + \sum_i \frac{\sigma_i}{\sigma_i^2 + \alpha}\mathbf{u}_i^{\text{H}} b_i^{\text{noise}}\mathbf{v}_i \tag{A.72}$$

Appendix B

Tests

B.1 Modal Assurance Criterion Number

The Modal Assurance Criterion (MAC) number is a mathematical tool to compare two vectors. It can measure the orthogonality between two mode-shapes. It gives unity when the mode-shapes are the same and zero if they are truly orthogonal. It is calculated as follows

$$\mathrm{MAC}_{ij} = \frac{\left[\boldsymbol{\phi}_i^{\mathrm{T}} \mathbf{W} \boldsymbol{\phi}_j\right]^2}{\left[\boldsymbol{\phi}_i^{\mathrm{T}} \mathbf{W} \boldsymbol{\phi}_i\right]\left[\boldsymbol{\phi}_j^{\mathrm{T}} \mathbf{W} \boldsymbol{\phi}_j\right]} \tag{B.1}$$

where $\mathbf{W}$ is the weighting matrix. The numerical mass $\mathbf{M}$ or stiffness $\mathbf{K}$ matrix can be used as a weighting matrix, giving the numerical eigen mode-shapes a MAC-table like a unity matrix by definition. If the mass and stiffness matrix are 'lumped', that is diagonally orientated, the MAC-table without any matrices as weighting matrix gives a good near unity result. For experimentally obtained mode-shapes, when there is no mass matrix, this is also used. Because, when the sensors are evenly distributed, it represents a 'lumped'-mass matrix, and thus the MAC-table can still be used.
The mode-shapes do not need to be normalised. Also, it does not have to be an eigen mode-shape, any mode-shape can make a MAC number. The mode-shapes do not have to be in displacement, also mode-shapes with rotational degrees of freedom or even in strain are possible to make MAC numbers.
The tests are published in the thesis as a table. See for an example table B.1

Table B.1: *Example of a MAC-table containing the MAC-numbers*

vs.			Mode-shapes of Second system				
		nr.	1	2	3	...	M
	nr.	[freq.]	$[f_1]$	$[f_2]$	$[f_3]$	...	$[f_M]$
Mode-shapes of First system	1	$[f_1]$	MAC_{11}	MAC_{12}	MAC_{13}	...	MAC_{1M}
	2	$[f_2]$	MAC_{21}	MAC_{22}	MAC_{23}	...	MAC_{2M}
	3	$[f_3]$	MAC_{31}	MAC_{32}	MAC_{33}	...	MAC_{3M}
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	m	$[f_m]$	MAC_{m1}	MAC_{m2}	MAC_{m3}	...	MAC_{mM}

B.2 Mass Normalisation Criterion

The mass normalisation criterion is based on the similarity between the continuous and discrete analyses. It can thus only be calculated when the continuous analysis is known. In appendix A.1, equation A.15 states

$$\overline{\phi_i} = \frac{\phi_i}{\sqrt{\mu_i}} \qquad \overline{\boldsymbol{\phi}}_{(i)} = \frac{1}{\sqrt{\mu_i}}\boldsymbol{\phi}_{(i)}$$

$$\rho A \int_L \overline{\phi_i}^2 \,\mathrm{d}y = 1 \qquad \overline{\boldsymbol{\phi}}_{(i)}^{\mathrm{T}} \mathbf{M} \overline{\boldsymbol{\phi}}_{(i)} = 1 \tag{B.2}$$

For a well discretized numerical eigen mode-shapes, the analytical statement can be used to test the mass normalisation. The numerical eigen mode-shapes need to be well discretisized to get an approximation close to the analytical calculation. Because from the analytical statement it follows that the equivalent length is

$$L_{eq} = \int_L \overline{\phi_i}^2 \,\mathrm{d}y = \frac{1}{\rho A} \tag{B.3}$$

First the mode-shape data must be sorted in ascending order with respect to the y-component of the node. Now, by computing the equivalent length of the numerical eigen mode-shapes with a 'pseudo' trapezium rule, as similar value should emerge. The trapezium-rule used by `CALC_ESSAI` is

$$\tilde{L}_{eq} = \sum_{j=2}^{N} \frac{1}{2} \left(\phi_{j-1}^2 + \phi_j^2\right) \Delta y \approx \frac{1}{\rho A} \tag{B.4}$$

This method builds, in a way, 'blocks' approximating the volume enclosed by the mode-shapes squared as depicted in figure B.1(b). As a second option, a summation of the squared frusta shown in figure B.1(c), can be calculated using the formula by Heron of Alexandrië [16]

$$V_{\text{frustrum}} = \frac{h}{3}\left(A_1 + A_2 + \sqrt{A_1 A_2}\right) \tag{B.5}$$

where h is the height of a frustrum and, A_1 and A_2, are the surface areas of the top and bottom plane. Adjusted for the mode-shape values and substituted in the summation, the approximation becomes

$$\tilde{L}_{eq} = \sum_{j=2}^{N} \frac{1}{3} \left(\overline{\phi}_{i,j-1}^2 + \overline{\phi}_{i,j}^2 + \overline{\phi}_{i,j-1} \overline{\phi}_{i,j} \right) \Delta y \approx \frac{1}{\rho A} \tag{B.6}$$

The first approximation overestimates the equivalent length, while the second approximation underestimates it. The second approximation is a lot better when the next mode-shape value is negative and the data is from a not well discretized mode-shape (like the raw experimental data), but when the data is well discretized this only has a small influence. Therefore the first approximation is used and `CALC_ESSAI` is not changed.
The finer the model is discretized, the closer the result should be to the analytical value of $\frac{1}{\rho A}$. For experimental mode-shape data, which is not well discretized, the values will be worse, but still should be in the order of the analytical value. After expansion, when the discretization is improved, the value should improve, but keep in mind that the experimental model is not exactly the same as the numerical models. Still it should give a value at least in the order of the analytical value. Figure B.2 visualises the analytical calculation as well as the two experimental approximations.

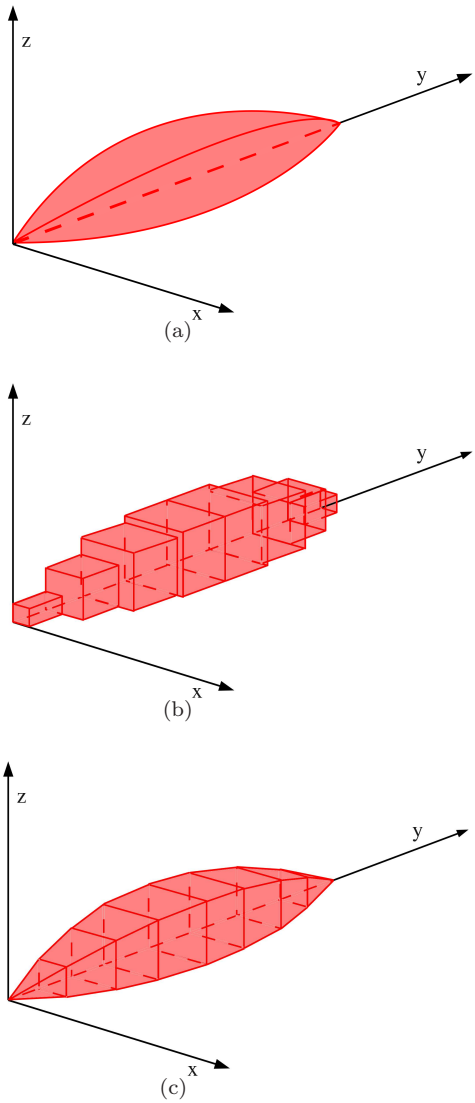


Figure B.1: (a) Visualisation using a continuous analytical mass normalisation of a mode-shape $\int_L \overline{\phi}^2 \, \mathrm{d}y = \frac{1}{\rho A}$, (b) visualisation of the 'block' approximation of mass normalisation of a mode-shapes $\sum_{j=2}^{N} \frac{1}{2} \left(\overline{\phi}_{i,j-1}^2 + \overline{\phi}_{i,j}^2 \right) \Delta y \approx \frac{1}{\rho A}$, and (c) visualisation of the 'frustrum' approximation of mass normalisation of a mode-shapes $\sum_{j=2}^{N} \frac{1}{3} \left(\overline{\phi}_{i,j-1}^2 + \overline{\phi}_{i,j}^2 + \overline{\phi}_{i,j-1} \overline{\phi}_{i,j} \right) \Delta y \approx \frac{1}{\rho A}$.

Appendix C

Modelling Practice

C.1 Contemplating for Added Stiffness due to Modelling

When using a FEM to model a real structure, the structure is basically divided into a number of small pieces for which the behaviour is determined at the interfaces. The pieces itself are left rigid and therefore stiffness is added to the system. Because, compared to the real thing, which can be viewed as an infinite number of infinite small pieces, the model consist of a low number of considerable sized pieces. When the number of pieces is increased, and the seize is thus reduced, the added stiffness goes towards zero. Now, to be able to account for this added stiffness (and in case of this study, be able to find the desired spring stiffness by hand iteration) the free-free numerical model (in case of this study, of the tube with the spring stiffness set to zero) is compared to the free-free analytical solution for the tube.
As an example for this method, the tube of this study is chosen, obviously. The free-free solution is to be found in any general dynamics book or lecture notes, [3], [7] or [6] and was also done in the internship report [15].
The first four eigen solutions are

$$\mu_0 = \frac{0}{L} \qquad \mu_1 = \frac{4.730041}{L} \qquad \mu_2 = \frac{7.853205}{L} \qquad \mu_3 = \frac{10.995608}{L} \qquad \mu_4 = \frac{14.137165}{L} \tag{C.1}$$

where μ_i gives information over the eigen mode-shape with respect to it corresponding eigen frequency ω_i as

$$\mu_i = \sqrt[4]{\frac{\rho A}{EI}\omega_i^2} \tag{C.2}$$

the first answer, μ_0, corresponds to the riged body mode, in which there is no shape distortion. The corresponding frequency (and radial frequency ω_0) is zero, no vibration, but just constant acceleration. The other radial frequencies ω_i can be calculated

$$\omega_i = \sqrt{\frac{EI}{\rho A}}\mu_i^2 \tag{C.3}$$

By now comparing the frequencies of the numerical free-free model with these frequencies of analytical analysis a linear factor is obtained which represents the frequency change due to the added stiffness. This factor is for each frequency pair almost the same, but slightly decreases for higher frequencies. By applying this factor to the experimental frequencies, new aiming frequencies for the numerical model can be set. In table C.1 the process is shown. Starting from the upper left side going down, the factors p are being calculated by dividing the numerical eigen frequencies for the free-free structure by the eigen frequencies of the analytical solution. Then, continuing from the bottom left corner to the right, multiplying the factors p with their respective experimental frequencies gives the numerical 'aiming' frequencies. These frequencies are depicted in bold in table C.1.
The factors slightly decrease for each higher eigen frequency, this is probably due to the fact that higher mode-shapes participate with less displacement and therefore the influence of the added stiffness is less. On the other hand, higher mode-shapes have stronger curvature and will thus suffer more from the added stiffness. This slight decrease is not visual due to rounding. Using some iteration method to approach these newly set

Table C.1: Table showing the process of factorisation of the 'added stiffness due to modelling' and the calculation of the numerical 'aiming' frequencies (in bold) for the iteration process.

1	2	3	4	Mode-shape number		
$f_{\text{num}}^{\text{free-free}}$	$f_{\text{num}}^{\text{free-free}}$	$f_{\text{num}}^{\text{free-free}}$	$f_{\text{num}}^{\text{free-free}}$	Numerical frequencies [Hz]		
$f_{\text{analytic}}^{\text{free-free}}$	$f_{\text{analytic}}^{\text{free-free}}$	$f_{\text{analytic}}^{\text{free-free}}$	$f_{\text{analytic}}^{\text{free-free}}$	Real frequencies [Hz]		
$p_{\text{mode 1}}$				$f_{\text{exp}}^{\text{springs}}$	$\mathbf{f_{num,aim}^{springs}}$	1
	$p_{\text{mode 2}}$			$f_{\text{exp}}^{\text{springs}}$	$\mathbf{f_{num,aim}^{springs}}$	2
		$p_{\text{mode 3}}$		$f_{\text{exp}}^{\text{springs}}$	$\mathbf{f_{num,aim}^{springs}}$	3
			$p_{\text{mode 4}}$	$f_{\text{exp}}^{\text{springs}}$	$\mathbf{f_{num,aim}^{springs}}$	4

frequencies, the numerical model can be updated taking into account the added stiffness. This method is of course only applicable if the analytical solution can be found. In case of a straight forward tube, as the case in this study, it is not a problem, but for any more complicated shape it soon becomes impossible to obtain the frequencies in an analytical matter and thus the method is not useful.
Second comment on this method is that the algorithm to obtain the frequencies for the numerical model is different for free-free situation as for a situation in which there are more boundary conditions. This difference could jeopardise the method.
The fact that the finally found spring stiffness for the two different models are the same, is a strong indication that the method works and gives good results.

C.2 Numerical Mode-Shapes Obtained by Code_Aster

Eigen mode-shapes have the property that they are orthogonal, and, in the numerical case, orthogonal with respect to the mass and stiffness matrix as shown in appendix A.1.

When Code_Aster calculates eigen mode-shapes, it uses a algorithm which is primarily based on this eigen mode-shape property. When the dynamic properties (the mass and stiffness) for the tube are the same in two directions (the x and z-direction in case of this study), there will be two mode-shapes on a ninety degree angle over the y-axis, to represent basically one mode-shape and its angle of orientation. The offset angle of this numerical pair of mode-shapes will be random. The next mode-shape pair is orthogonal to the mode-shape represented by the first mode-shape pair, for any angle of orientation. So, the first numerical mode-shape of the second pair has again a random offset angle. The second numerical shape is aligned at a ninety degree angle again, and so on. A pattern as depicted in figure C.1 will emerge.
If the 3D volume-element model is now compared to the 2D shell-element model the corresponding mode-shape pairs have different orientations, and therefore only a partial influence. But one pair of numerical mode-shapes of one model can describe the mode-shape of the other model, so the MAC-numbers for corresponding real mode-shapes do add up to around unity. This is quite obvious, because the projection changes the shape. The MAC-numbers show a correlation without taking amplitude into account, and thus the shape change due to the this projection of the mode-shape, gives exactly the reduction form one.
The measurements in air, both the ones with the use of the PAK-system as well as the ones obtained with the use of the LMS-system, have a specific direction. Namely the direction in which the hammer was struck. Because the measurement data was hand adjusted as explained in chapter 2, section 2.4, the measurements and the expansions have all mode-shape pairs in the same direction.

C.3 Simple Model of the Spring

To have an indication of the ratio between the stiffness of the spring in z- (or x-) direction (perpendicular to the rod) and in y-direction (along the rod) a simple analysis is made. The rotational stiffness is not modelled and set to zero because this is taken care of by the stiffness in y-direction time the radius of the cross section.
To get an indication of the ratio between the spring stiffness in the direction perpendicular to the tube and the stiffness along the tube, a simple calculation is set up using figure C.2. This figure shows four schematic representations of the spring. The shaded part of the spring of the upper right part scheme is repeated in the lower left, with a representation of the forces. In this representation the sum of the moments on A gives

$$\sum M_A = 0 \tag{C.4}$$

which becomes

$$F \cdot \frac{h_t}{2} - \int_0^{2\pi} \left[\underbrace{\frac{k_y}{2\pi\left(r-\frac{t}{2}\right)}\left(r-\frac{t}{2}\right)}_{\text{'stiffness'}} \underbrace{\sin\theta\sin\psi\left(r-\frac{t}{2}\right)}_{\text{'displacement'}} \underbrace{\left(r-\frac{t}{2}\right)\sin\psi}_{\text{'moment arm'}} \right] d\psi = 0 \tag{C.5}$$

The 'stiffness' is the stiffness in axial direction, along the tube. This is build up from k_y divided by the circumference times the increments over the circumference $\left(r-\frac{t}{2}\right)\,\mathrm{d}\psi$. This division is done so that when this part is multiplied and integrated over, it becomes

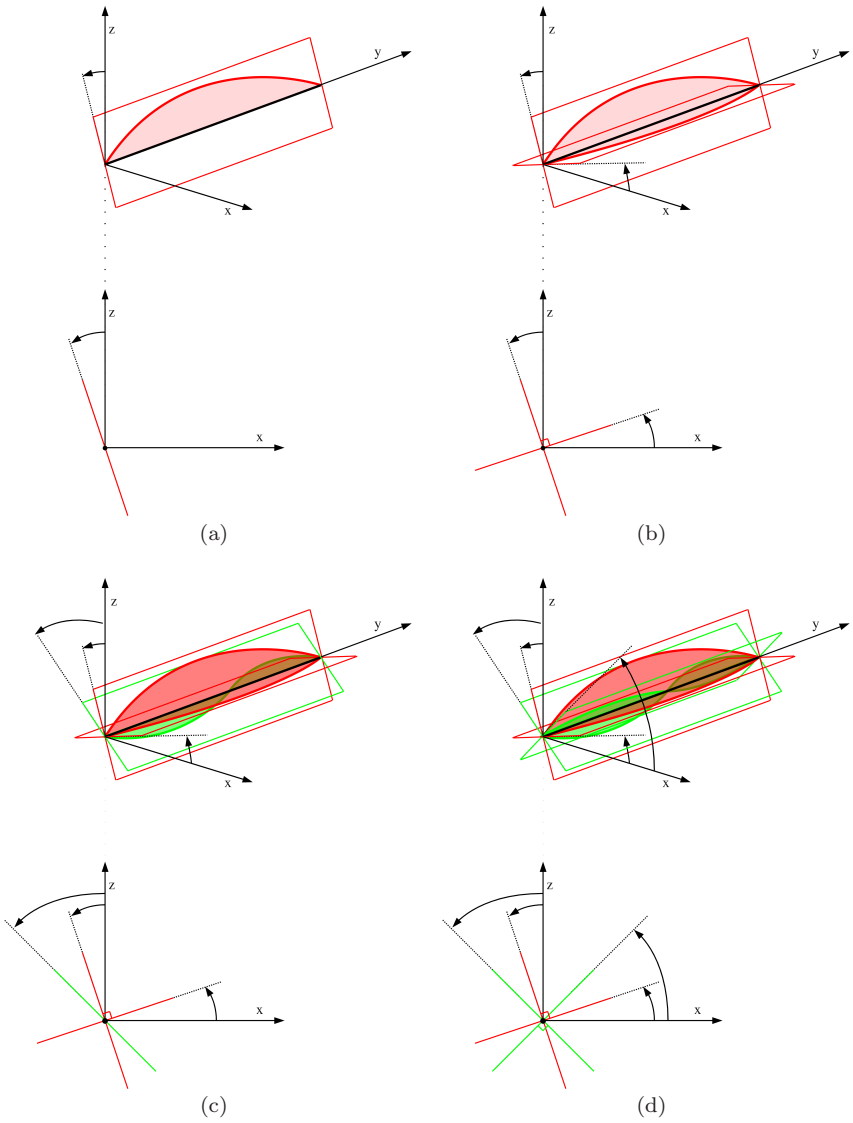


Figure C.1: (a) the first mode-shapes is found and orientated randomly, (b) the second numerical mode-shapes completes the two dimensional space and, to be orthogonal, is placed on a ninety degree angle. (c) the third numerical mode-shape is orthogonal to the first pair and is thus again randomly placed. (d) So a random scatter of mode-shape pairs arises.

the stiffness.
The 'displacement' part is build form the radius $\left(r - \frac{t}{2}\right)$ times the sine of the integration angle ψ to correct for the fact that an arm is needed with the length from the bending axis to the place on the circumference and times the sine of θ for the amount of bending

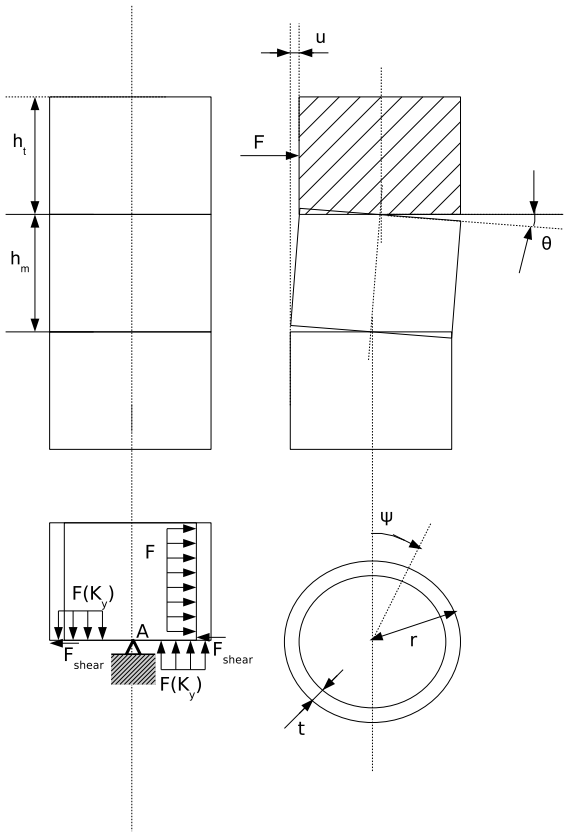


Figure C.2: Scheme for the simple analysis of the spring

around the bending axis.
The 'moment arm' is the radius $\left(r - \frac{t}{2}\right)$ times the sine of the integration angle ψ to correct for the fact that we need as an arm the length from the bending axis to the place on the circumference. With

$$F = k_z u \qquad \text{and} \qquad u = h_m \sin\theta \tag{C.6}$$

this becomes

$$k_z \frac{h_m h_t}{2} \sin\theta - \frac{k_y}{2\pi} \left(r - \frac{t}{2}\right)^2 \sin\theta \int_0^{2\pi} \sin^2\psi d\psi = 0 \tag{C.7}$$

which gives

$$\left(k_z \frac{h_m h_t}{2} - \frac{k_y}{2\pi} \left(r - \frac{t}{2}\right)^2 [\pi]\right) \sin\theta = 0 \tag{C.8}$$

and

$$\sin\theta \neq 0 \tag{C.9}$$

so

$$k_z h_m h_t - k_y \left(r - \frac{t}{2} \right)^2 = 0 \rightarrow k_z = k_y \frac{(2r-t)^2}{4 h_m h_t} \tag{C.10}$$

This shows that as an indication $k_{x,z}$ should be around 11.3% of k_y.

C.4 Spring Positions

The spring positions and the four corresponding nodes used to model the spring behaviour are depicted for both numerical models in table C.2.

Table C.2: Table showing the spring position and the corresponding nodes for two numerical models. where R is the radius of the tube.

Spring position along y axis [mm]	Node numbers on cross section							
	3D Volume-element model				2D Shell-element model			
	$x = R$ $z = 0$	$x = -R$ $z = 0$	$x = 0$ $z = R$	$x = 0$ $z = -R$	$x = R$ $z = 0$	$x = -R$ $z = 0$	$x = 0$ $z = R$	$x = 0$ $z = -R$
297	N303	N1410	N204	N5543	N720	N3118	N34	N250
797	N353	N1460	N154	N5993	N1170	N3568	N84	N300
1294	N402	N1509	N105	N6443	N1611	N4009	N133	N349
1883	N461	N1784	N46	N6974	N2142	N4540	N192	N408
2089	N482	N1805	N25	N7163	N2331	N4729	N213	N429

Appendix D

Software & Programs

D.1 Software Packages

During this research three software packages were extensively used:

- Salomé_Méca: To make the numerical models and to visualise the mode-shapes of the expanded experiments.

- Code_Aster: To make the numerical calculations. From reading the measurement data, expanding it on the numerical models, to using the routine `CALC_ESSAI` to identify the forces.

- Python: To built subroutine callable from Code_Aster to be able to do the research which cannot yet be done by the routines in Code_Aster.

D.2 Code_Aster & Python Programs

The research is structured fairly similar to the structure in the thesis. The programs in Code_Aster are divided into five parts:

- Building the numerical models. In this program the 3D volume-element model and the 2D shell-element model are built.

- Reading the measurement data. In this program the measurements in air, obtained using the PAK-system and the LMS-system, are read-in from the data-files. Also the measurements in water, in strain, are read.

- Expanding the measurement data on the numerical models. There are two similar programs both doing the expansion, but one operates the 'projecting' method and the other one the 'reduction' method, as explained in chapter 4. The command file of the 'reduction' method will be shown.

- Identifying the forces. In this program a model is built, which is to be used as and controllability matrix **B**. This model consists of the points, which model the fluid force. Then `CALC_ESSAI` is started to do the force identification.
- Applying the forces to the Magaly model. The numerical representation of the Magaly model is built in this program. Next to that, a time signal, representing the force should be built using `GENE_FONC_ALEA`. Because this does not work and the numerical representation of the Magaly model should be built in a different way, this program is omitted.

The mass normalisation is done calling Python-routines from Code_Aster, which rewrite the data-files. Python is also used to calculate the MAC-tables. The Code_Aster command file are shown as well as the Python routines.

```
#####################################################################################################
#                                                                                                   #
#   Building the 2D and 3D Model in both the x- and z-direction                                     #
#                                                                                                   #
#   This Code_Aster program will build the two models (2D and 3D). This model will have no re-      #
#   tricted movement in the z-direction. So the obtained mode-shapes will be in both directions.    #
#                                                                                                   #
#   Joost de Jong                                                                                   #
#                                                                                                   #
#####################################################################################################

#====================================================================================================#
#   Begin
#====================================================================================================#
import sys
import Numeric
sys.path.append('/home/joost/Bureaublad/Study_EDF/python_files/')
from modes import *
from nodes_2D_vs_3D import *

DEBUT(PAR_LOT = 'NON');

#   Obtaining the nodes to be able to compare the 2D and 3D model later on...
noeuds3D, noeuds2D, index3D, index2D = nodes_2D_vs_3D();
#----------------------------------------------------------------------------------------------------#
#   Defining the nodes which model the five springs along the rod
#----------------------------------------------------------------------------------------------------#
noeu3D = ('N303', 'N353', 'N402','N461','N482','N204','N154','N105','N46','N25','N1410','N1460','N1509…
noeu2D = ('N34','N84','N133','N192','N213','N250', 'N300', 'N349','N408','N429','N720','N1170','N1611'…
#----------------------------------------------------------------------------------------------------#
#   Defining the nodes where the accelerometer sensors are
#----------------------------------------------------------------------------------------------------#
# for r= 4mm;   nsensa = ('N4199','N4216','N4237','N4249','N4259','N4289','N4305','N4318','N4334','N43…
nsensa3D = ('N1382','N1399','N1420','N1432','N1442','N1472','N1488','N1501','N1517','N1530','N1547','N…
# not necessary nsensaz = ('N275','N292','N313','N325','N335','N365','N381','N394','N410','N423','N440…
nsensa2D = ('N468','N630','N810','N927','N1017','N1278','N1422','N1539','N1683','N1800','N1953','N2061…

#====================================================================================================#
#   Importing the Nummerical model 'maille' to project the 3D elements on, in 3D.
#====================================================================================================#
MAINtmp3    = LIRE_MAILLAGE(    FORMAT          = 'ASTER',
                                UNITE           = 20,
                                );

MAINtmp3    = DEFI_GROUP(       reuse           = MAINtmp3,
                                MAILLAGE        = MAINtmp3,
                                CREA_GROUP_MA   = _F(   NOM             = 'TU_MA3D',
                                                        TOUT            = 'OUI',
                                                        ),
                                CREA_GROUP_NO   = _F(   NOM             = 'NOEUD3D',
                                                        NOEUD           = noeuds3D,
                                                        ),
                               );

MAIN3D      = CREA_MAILLAGE(    MAILLAGE        = MAINtmp3,
                                CREA_POI1       = _F(   NOEUD           = noeu3D,
                                                        NOM_GROUP_MA    = 'SOU_MA3D',
                                                        ),
                                );

#DETRUIRE(                       CONCEPT         = _F(   NOM             = MAINtemp,
#                                                        ),
#                                INFO            = 1,
#                                );
#
MODELN3D    = AFFE_MODELE(      MAILLAGE        = MAIN3D,
                                VERIF           = 'MAILLE',
                                AFFE            = ( _F( GROUP_MA        = 'TU_MA3D',
                                                        PHENOMENE       = 'MECANIQUE',
                                                        MODELISATION    = '3D',
                                                        ),
                                                    _F( GROUP_MA        = 'SOU_MA3D',
```

```
                                                    PHENOMENE       = 'MECANIQUE',
                                                    MODELISATION    = 'DIS_T',
                                                    ),
                                                ),
                                );

MATCRAY     = DEFI_MATERIAU(    ELAS            = _F(   E               = 210.E9,
                                                        NU              = 0.3,
                                                        RHO             = 7800.,
                                                        ),
                                );

#-------------------------------------------------------------------------------------------------#
#   Assigning the springstiffness values
#-------------------------------------------------------------------------------------------------#
K1__        = 2.1E7/4                       #1.7E7/4; #Ky2.1E7/4#
K2__        = 1.8E6/4                       #2.4E6/4; #Kx, Kz1.8E6/4#
#-------------------------------------------------------------------------------------------------#

CARAN3D     = AFFE_CARA_ELEM(   MODELE          = MODELN3D,
                                DISCRET         = (     _F(     GROUP_MA        = 'SOU_MA3D',
                                                                CARA            = 'K_T_D_N',
                                                                VALE            = (K2__, K1__, K2__),
                                                                ),
#                                                       _F(     GROUP_MA        = 'SOU_MA3D',
#                                                               CARA            = 'M_T_D_N',
#                                                               VALE            = 0.0001,
#                                                               ),
                                                        ),
                                );

MATN3D      = AFFE_MATERIAU(    MAILLAGE        = MAIN3D,
                                MODELE          = MODELN3D,
                                AFFE            = _F(   GROUP_MA        = 'TU_MA3D',
                                                        MATER           = MATCRAY,
                                                        ),
                                );

KELN3D      = CALC_MATR_ELEM(   OPTION          = 'RIGI_MECA',
                                MODELE          = MODELN3D,
                                CHAM_MATER      = MATN3D,
                                CARA_ELEM       = CARAN3D,
                                );

MELN3D      = CALC_MATR_ELEM(   OPTION          = 'MASS_MECA',
                                MODELE          = MODELN3D,
                                CHAM_MATER      = MATN3D,
                                CARA_ELEM       = CARAN3D,
                                );

NUMEN3D     = NUME_DDL(         MATR_RIGI       = KELN3D,
                                );

KASN3D      = ASSE_MATRICE(     MATR_ELEM       = KELN3D,
                                NUME_DDL        = NUMEN3D,
                                );

MASN3D      = ASSE_MATRICE(     MATR_ELEM       = MELN3D,
                                NUME_DDL        = NUMEN3D,
                                INFO            = 2,
                                );

MODEA3Dt    = MODE_ITER_SIMULT( MATR_A          = KASN3D,
                                MATR_B          = MASN3D,
                                METHODE         = 'SORENSEN',
                                TYPE_RESU       = 'DYNAMIQUE',
                                OPTION          = 'SANS',
                                CALC_FREQ       = _F(   OPTION          = 'BANDE',
                                                        APPROCHE        = 'REEL',
                                                        FREQ            = (10.,2000.),
                                                        ),
                                VERI_MODE       = _F(   STOP_ERREUR     = 'NON',
```

```
                                                    SEUIL           = 1e-03,
                                                    ),
                                );

MODEA3Dt    = NORM_MODE(        reuse           = MODEA3Dt,
                                MODE            = MODEA3Dt,
                                NORME           = 'MASS_GENE',
                                );

#-----------------------------------------------------------------------------------------------#
# To pick specific modeshapes from the first 14 determinted,change the sequence of 'NUME_MODE'
#-----------------------------------------------------------------------------------------------#
MODEA3D     = EXTR_MODE(        FILTRE_MODE     = ( _F( MODE            = MODEA3Dt,
                                                        NUME_ORDRE      = (1,2,3,4,5,6,7,8,),
                                                        ),
#                                                    _F( MODE            = MODEA3Dt,
#                                                        NUME_MODE       = (2,4,6,8,),
#                                                        ),
                                                    ),
                                );

#===============================================================================================#
#   Calculating the strain from the displacements of the MODESHAPES of MODEAIR!!!
#   (using first the strain in the elements: EPSI_ELNO_DEPL and than the strain at the
#   nodes: EPSI_NOEU_DEPL.
#===============================================================================================#

MODEA3D     = CALC_ELEM(        reuse           = MODEA3D,
                                RESULTAT        = MODEA3D,
                                GROUP_MA        = 'TU_MA3D',
                                OPTION          = 'EPSI_ELNO_DEPL'
                                );

MODEA3D     = CALC_NO(          reuse           = MODEA3D,
                                RESULTAT        = MODEA3D,
                                GROUP_MA        = 'TU_MA3D',
                                OPTION          = 'EPSI_NOEU_DEPL'
                                );
#===============================================================================================#
#   Building the statique mode-shapes on 3D model
#   for the accelerometers (PAK) positions
#===============================================================================================#

#-----------------------------------------------------------------------------------------------#
#   in x-direction
#-----------------------------------------------------------------------------------------------#
LIAISO3a    = AFFE_CHAR_MECA(   MODELE          = MODELN3D,
                                DDL_IMPO        = _F(   NOEUD           = nsensa3D,
                                                        DX              = 0.0,
                                                        DZ              = 0.0,
                                                        ),
                                );

KELN3Da     = CALC_MATR_ELEM(   OPTION          = 'RIGI_MECA',
                                MODELE          = MODELN3D,
                                CHAM_MATER      = MATN3D,
                                CARA_ELEM       = CARAN3D,
                                CHARGE          = LIAISO3a,
                                );

NUMEN3Da    = NUME_DDL(         MATR_RIGI       = KELN3Da,
                                );


KASN3Da     = ASSE_MATRICE(     MATR_ELEM       = KELN3Da,
                                NUME_DDL        = NUMEN3Da,
                                );

MODSTA3a    = MODE_STATIQUE(    #INFO            =  2,
                                MATR_RIGI       =  KASN3Da,
                                MODE_STAT       = _F(   NOEUD           = nsensa3D,
                                                        SANS_CMP        = ( 'DY',
```

```
                                                                    ),
                                                ),
                              );

#=============================================================================================#
#   Building the statique mode-shapes
#   for the spring positions
#=============================================================================================#

#---------------------------------------------------------------------------------------------#
#   in x-direction
#---------------------------------------------------------------------------------------------#
LIAISO3s    = AFFE_CHAR_MECA(   MODELE          = MODELN3D,
                                DDL_IMPO        = _F(   NOEUD           = noeu3D[:5],
                                                        DX              = 0.0,
                                                        DZ              = 0.0,
                                                        ),
                              );

KELN3Ds     = CALC_MATR_ELEM(   OPTION          = 'RIGI_MECA',
                                MODELE          = MODELN3D,
                                CHAM_MATER      = MATN3D,
                                CARA_ELEM       = CARAN3D,
                                CHARGE          = LIAISO3s,
                                );

NUMEN3Ds    = NUME_DDL(         MATR_RIGI       = KELN3Ds,
                                );


KASN3Ds     = ASSE_MATRICE(     MATR_ELEM       = KELN3Ds,
                                NUME_DDL        = NUMEN3Ds,
                                );

MODSTA3s    = MODE_STATIQUE(    #INFO            =  2,
                                MATR_RIGI       =  KASN3Ds,
                                MODE_STAT       = _F(   NOEUD           = noeu3D[:5],
                                                        SANS_CMP        = ( 'DY',
                                                            ),
                                                        ),
                              );
#=============================================================================================#
#   Importing the Nummerical model 'maille' to project the shell-elements on, in 2D.
#=============================================================================================#
MAINtmp2    = LIRE_MAILLAGE(    FORMAT          = 'ASTER',
                                UNITE           = 21,
                                );

MAINtmp2    = DEFI_GROUP(       reuse           = MAINtmp2,
                                MAILLAGE        = MAINtmp2,
                                CREA_GROUP_MA   = _F(   NOM             = 'TU_MA2D',
                                                        TOUT            = 'OUI',
                                                        ),
                                CREA_GROUP_NO   = _F(   NOM             = 'NOEUD2D',
                                                        NOEUD           = noeuds2D,
                                                        ),
                                 );


MAIN2D      = CREA_MAILLAGE(    MAILLAGE        = MAINtmp2,
                                CREA_POI1       = _F(   NOEUD           = noeu2D,
                                                        NOM_GROUP_MA    = 'SOU_MA2D'
                                                        ),
                              );

#DETRUIRE(                       CONCEPT         = _F(   NOM             = MAINtemp,
#                                                        ),
#                                INFO            = 1,
#                                );

MODELN2D    = AFFE_MODELE(      MAILLAGE        = MAIN2D,
                                VERIF           = 'MAILLE',
```

```
                              AFFE            = ( _F( GROUP_MA        = 'TU_MA2D',
                                                      PHENOMENE       = 'MECANIQUE',
                                                      MODELISATION    = 'DKT',
                                                      ),
                                                  _F( GROUP_MA        = 'SOU_MA2D',
                                                      PHENOMENE       = 'MECANIQUE',
                                                      MODELISATION    = 'DIS_T',
                                                      ),
                                                ),
                              );

CARAN2D     = AFFE_CARA_ELEM(   MODELE          = MODELN2D,
                                DISCRET         = _F(   GROUP_MA        = 'SOU_MA2D',
                                                        CARA            = 'K_T_D_N',
                                                        VALE            = (K2__, K1__, K2__),
                                                        ),
                                COQUE           = _F(   GROUP_MA         = 'TU_MA2D',
                                                        EPAIS           = 1E-3,
                                                        ),
                                );

MATN2D      = AFFE_MATERIAU(    MAILLAGE        = MAIN2D,
                                MODELE          = MODELN2D,
                                AFFE            = _F(   GROUP_MA        = 'TU_MA2D',
                                                        MATER           = MATCRAY,
                                                        ),
                                );

KELN2D      = CALC_MATR_ELEM(   OPTION          = 'RIGI_MECA',
                                MODELE          = MODELN2D,
                                CHAM_MATER      = MATN2D,
                                CARA_ELEM       = CARAN2D,
                                );

MELN2D      = CALC_MATR_ELEM(   OPTION          = 'MASS_MECA',
                                MODELE          = MODELN2D,
                                CHAM_MATER      = MATN2D,
                                CARA_ELEM       = CARAN2D,
                                );

NUMEN2D     = NUME_DDL(         MATR_RIGI       = KELN2D,
                                );

KASN2D      = ASSE_MATRICE(     MATR_ELEM       = KELN2D,
                                NUME_DDL        = NUMEN2D,
                                );

MASN2D      = ASSE_MATRICE(     MATR_ELEM       = MELN2D,
                                NUME_DDL        = NUMEN2D,
                                );

MODEA2Dt    = MODE_ITER_SIMULT( MATR_A          = KASN2D,
                                MATR_B          = MASN2D,
                                METHODE         = 'SORENSEN',
                                TYPE_RESU       = 'DYNAMIQUE',
                                OPTION          = 'SANS',
                                CALC_FREQ       = _F(   OPTION          = 'BANDE',
                                                        APPROCHE        = 'REEL',
                                                        FREQ            = (10.0,2000.0),
                                                        ),
                                VERI_MODE       = _F(   STOP_ERREUR     = 'NON',
                                                        SEUIL           = 1e-03,
                                                        ),
                                );

MODEA2Dt    = NORM_MODE(        reuse           = MODEA2Dt,
                                MODE            = MODEA2Dt,
                                NORME           = 'MASS_GENE',
                                );
#-------------------------------------------------------------------------------------------------#
# To pick specific modeshapes from the first 14 determinted,change the sequence of 'NUME_MODE'
#-------------------------------------------------------------------------------------------------#
```

```
MODEA2D     = EXTR_MODE(        FILTRE_MODE     = ( _F( MODE            = MODEA2Dt,
                                                      NUME_ORDRE      = (1,2,3,4,5,6,7,8,),
                                                      ),
#                                                   _F( MODE            = MODEA2Dt,
#                                                       NUME_MODE       = (2,4,6,8,),
#                                                       ),
                                                  ),
                                );

#=================================================================================================#
#   Calculating the strain from the displacements of the MODESHAPES of MODEAIR!!!
#   (using first the strain in the elements: EPSI_ELNO_DEPL and than the strain at the
#   nodes: EPSI_NOEU_DEPL.
#=================================================================================================#

MODEA2D     = CALC_ELEM(        reuse           = MODEA2D,
                                RESULTAT        = MODEA2D,
                                GROUP_MA        = 'TU_MA2D',
                                OPTION          = 'EPSI_ELNO_DEPL'
                                );

MODEA2D     = CALC_NO(          reuse           = MODEA2D,
                                RESULTAT        = MODEA2D,
                                GROUP_MA        = 'TU_MA2D',
                                OPTION          = 'EPSI_NOEU_DEPL'
                                );

#=================================================================================================#
#   Building the statique mode-shapes on 2D model
#   for the accelerometers (PAK) positions
#=================================================================================================#

#-------------------------------------------------------------------------------------------------#
#   in x-direction
#-------------------------------------------------------------------------------------------------#
LIAISO2a    = AFFE_CHAR_MECA(   MODELE          = MODELN2D,
                                DDL_IMPO        = _F(   NOEUD           = nsensa2D,
                                                        DX              = 0.0,
                                                        DZ              = 0.0,
                                                        ),
                                );

KELN2Da     = CALC_MATR_ELEM(   OPTION          = 'RIGI_MECA',
                                MODELE          = MODELN2D,
                                CHAM_MATER      = MATN2D,
                                CARA_ELEM       = CARAN2D,
                                CHARGE          = LIAISO2a,
                                );

NUMEN2Da    = NUME_DDL(         MATR_RIGI       = KELN2Da,
                                );


KASN2Da     = ASSE_MATRICE(     MATR_ELEM       = KELN2Da,
                                NUME_DDL        = NUMEN2Da,
                                );

MODSTA2a    = MODE_STATIQUE(    #INFO            =  2,
                                MATR_RIGI       =  KASN2Da,
                                MODE_STAT       = _F(   NOEUD           = nsensa2D,
                                                        SANS_CMP        = ( 'DY','DRX','DRY','DRZ',
                                                            ),
                                                        ),
                                );

#=================================================================================================#
#   Building the statique mode-shapes
#   for the spring positions
#=================================================================================================#

#-------------------------------------------------------------------------------------------------#
#   in x-direction
```

```
#-------------------------------------------------------------------------------------------------#
LIAISO2s    = AFFE_CHAR_MECA(   MODELE          = MODELN2D,
                                DDL_IMPO        = _F(   NOEUD           = noeu2D[:5],
                                                        DX              = 0.0,
                                                        DZ              = 0.0,
                                                        ),
                                );

KELN2Ds     = CALC_MATR_ELEM(   OPTION          = 'RIGI_MECA',
                                MODELE          = MODELN2D,
                                CHAM_MATER      = MATN2D,
                                CARA_ELEM       = CARAN2D,
                                CHARGE          = LIAISO2s,
                                );

NUMEN2Ds    = NUME_DDL(         MATR_RIGI       = KELN2Ds,
                                );


KASN2Ds     = ASSE_MATRICE(     MATR_ELEM       = KELN2Ds,
                                NUME_DDL        = NUMEN2Ds,
                                );

MODSTA2s    = MODE_STATIQUE(    #INFO            =  2,
                                MATR_RIGI       =  KASN2Ds,
                                MODE_STAT       = _F(   NOEUD           = noeu2D[:5],
                                                        SANS_CMP        = ( 'DY','DRX','DRY','DRZ',
                                                            ),
                                                        ),
                                );
#=================================================================================================#
#   Building the 1D model, which will be used to compare the 2D and 3D model by projection.
#=================================================================================================#

MAYA_1D     = LIRE_MAILLAGE(    UNITE           = 22,
                                );

MAYA_1D     = DEFI_GROUP(       reuse           = MAYA_1D,
                                MAILLAGE        = MAYA_1D,
                                CREA_GROUP_MA   = _F(   NOM             = 'MC',
                                                        TOUT            = 'OUI',
                                                        ),
                                );

MODEL_BA    = AFFE_MODELE(      MAILLAGE        = MAYA_1D,
                                VERIF           = 'MAILLE',
                                AFFE            = _F(   PHENOMENE       = 'MECANIQUE',
                                                        MODELISATION    = 'BARRE',
                                                        GROUP_MA        = 'MC'
                                                        ),
                                );

epaisseur   = 0.001
rayon       = 0.004
CARA_BA     = AFFE_CARA_ELEM(   MODELE          =  MODEL_BA,
                                BARRE           = _F(   GROUP_MA        = 'MC',
                                                        SECTION         = 'CERCLE',
                                                        CARA            = ('EP','R',),
                                                        VALE            = (epaisseur, rayon),
                                                        ),
                                );

MAT_BA      = AFFE_MATERIAU(    MAILLAGE        = MAYA_1D,
                                MODELE          = MODEL_BA,
                                AFFE            = _F(   GROUP_MA        = 'MC',
                                                        MATER           = MATCRAY,
                                                        ),
                                );

KEL_BA      = CALC_MATR_ELEM(   OPTION          = 'RIGI_MECA',
                                MODELE          = MODEL_BA,
                                CHAM_MATER      = MAT_BA,
```

```
                                CARA_ELEM       = CARA_BA,
                                );


MEL_BA      = CALC_MATR_ELEM(   OPTION          = 'MASS_MECA',
                                MODELE          = MODEL_BA,
                                CHAM_MATER      = MAT_BA,
                                CARA_ELEM       = CARA_BA,
                                );

NUME_BA     = NUME_DDL(         MATR_RIGI       = KEL_BA,
                                );

KAS_BA      = ASSE_MATRICE(     MATR_ELEM       = KEL_BA,
                                NUME_DDL        = NUME_BA,
                                );

MAS_BA      = ASSE_MATRICE(     MATR_ELEM       = MEL_BA,
                                NUME_DDL        = NUME_BA,
                                );

#=================================================================================================#
#   Comparing the 3D and 2D models
#
#   This is done by making a MAC in two ways:
#   - by obtaining the corresponding nodes in the two models and extracting just the translational
#     DOF from both corresponding modeshapes. And then MAC-ing these 'reduced' modeshapes against
#     each other
#   - by first projecting the mode-shapes of the models on a 1D bar model and then MAC-ing these
#     two models against each other. This can only be done in displacement, because the 1D model
#     is a bar model and can therefore not calculate any strain. That the 1D model is a bar model
#     is because the 3D model has only got translational DOF and can therefore only be projected
#     on a bar model.
#
#=================================================================================================#


#=================================================================================================#
#   Auto MACs full
#=================================================================================================#

base3D = extract_modes(MODEA3D, '3D Model Full', 'displacement', '','')
mac_modes(base3D,base3D)
base2D = extract_modes(MODEA2D, '2D Model Full', 'displacement', '','')
mac_modes(base2D,base2D)
MAC3DRED    = MAC_MODES(        BASE_1          = MODEA3D,
                                BASE_2          = MODEA3D,
                                INFO            = 2,
                                );
MAC2DRED    = MAC_MODES(        BASE_1          = MODEA2D,
                                BASE_2          = MODEA2D,
                                INFO            = 2,
                                );
base3D = extract_modes(MODEA3D, '3D Model Full', 'strain', '','')
mac_modes(base3D,base3D)
base2D = extract_modes(MODEA2D, '2D Model Full', 'strain', '','')
mac_modes(base2D,base2D)

#=================================================================================================#
#   Reduced Method
#=================================================================================================#
base3D = extract_modes(MODEA3D, '3D Model Reduced', 'displacement', "'NOEUD3D'", index3D)
mac_modes(base3D,base3D)
base2D = extract_modes(MODEA2D, '2D Model Reduced', 'displacement', "'NOEUD2D'", index2D)
mac_modes(base2D,base2D)
mac_modes(base3D,base2D)

base3Ds = extract_modes(MODEA3D, '3D Model Reduced', 'strain', "'NOEUD3D'", index3D)
mac_modes(base3Ds,base3Ds)
base2Ds = extract_modes(MODEA2D, '2D Model Reduced', 'strain', "'NOEUD2D'", index2D)
mac_modes(base2Ds,base2Ds)
mac_modes(base3Ds,base2Ds)
```

```
#=======================================================================================================#
#   Projection Method
#=======================================================================================================#

RE3DNUM     = PROJ_CHAMP(       RESULTAT        = MODEA3D,
                                METHODE         = 'ELEM',
                                MODELE_1        = MODELN3D,
                                MODELE_2        = MODEL_BA,
                                NOM_CHAM        = 'DEPL',
                                NUME_DDL        = NUME_BA,
                                );
RE2DNUM     = PROJ_CHAMP(       RESULTAT        = MODEA2D,
                                METHODE         = 'ELEM',
                                MODELE_1        = MODELN2D,
                                MODELE_2        = MODEL_BA,
                                NOM_CHAM        = 'DEPL',
                                NUME_DDL        = NUME_BA,
                                );

proj3D = extract_modes(RE3DNUM, '3D Model Projected', 'displacement', '', '')
mac_modes(proj3D,proj3D)
proj2D = extract_modes(RE2DNUM, '2D Model Projected', 'displacement', '', '')
mac_modes(proj2D,proj2D)
mac_modes(proj3D,proj2D)

MAC3DNUM    = MAC_MODES(        BASE_1          = RE3DNUM,
                                BASE_2          = RE3DNUM,
                                MATR_ASSE       = MAS_BA,
                                INFO            = 2,
                                );
MAC2DNUM    = MAC_MODES(        BASE_1          = RE2DNUM,
                                BASE_2          = RE2DNUM,
                                MATR_ASSE       = MAS_BA,
                                INFO            = 2,
                                );
MAC2DN3D    = MAC_MODES(        BASE_1          = RE3DNUM,
                                BASE_2          = RE2DNUM,
                                MATR_ASSE       = MAS_BA,
                                INFO            = 2,
                                );

RE2DN3D     = PROJ_CHAMP(       RESULTAT        = MODEA2D,
                                METHODE         = 'ELEM',
                                MODELE_1        = MODELN2D,
                                MODELE_2        = MODELN3D,
                                NOM_CHAM        = 'DEPL',
                                NUME_DDL        = NUMEN3D,
                                );

MAC3DP3D    = MAC_MODES(        BASE_1          = RE2DN3D,
                                BASE_2          = MODEA3D,
                                MATR_ASSE       = MASN3D,
                                INFO            = 2,
                                );
CALC_ESSAI( UNITE_RESU = 8,
            INTERACTIF = 'OUI',
            );

#=======================================================================================================#
#   Kill the to many concepts
#=======================================================================================================#

DETRUIRE(                       CONCEPT         = _F(   NOM             = MODEA3D,
                                                        ),
                                INFO            = 1,
                                );
DETRUIRE(                       CONCEPT         = _F(   NOM             = MODEA2D,
                                                        ),
                                INFO            = 1,
```

```
                                );

#=========================================================================================#
FIN();
#=========================================================================================#
```

```
#####################################################################################################
#                                                                                                   #
#   Processing the Experimental Data                                                  27 may 2009   #
#                                                                                                   #
#   This Code_Aster Program will process the Experimental data. It loads a mesh with the posi-     #
#   tions of the sensors and then it distributes the modeshapes over it. It does this for both     #
#   PAK and LMS. It uses 'POURSUITE' to increase the data in the datafiles, to be able to do the   #
#   expansion later on.                                                                             #
#                                                                                                   #
#   Joost de Jong                                                                                   #
#                                                                                                   #
#####################################################################################################

#===================================================================================================#
#   Begin
#===================================================================================================#

import sys
import Numeric
sys.path.append('/home/joost/Bureaublad/Study_EDF/python_files/')
from modes import *
#from to_file_print_new_LMS_data import *
from to_file_print_new_PAK_data import *
POURSUITE(PAR_LOT = 'NON');

#===================================================================================================#
#   Importing the expirimental 'maille' for accelerometers and creating the model
#   for the values gives by the accelerometers (PAK)
#===================================================================================================#

MAYAPAK     = LIRE_MAILLAGE(    UNITE           = 20,
                                FORMAT          = 'ASTER'
                                );



MAYAPAK     = DEFI_GROUP(       reuse           = MAYAPAK,
                                MAILLAGE        = MAYAPAK,
                                CREA_GROUP_MA   = _F(   NOM             = 'MCPAK',
                                                        TOUT            = 'OUI',
                                                        ),
                                );

MODELPAK    = AFFE_MODELE(      MAILLAGE        = MAYAPAK,
                                VERIF           = 'MAILLE',
                                AFFE            = _F(   PHENOMENE       = 'MECANIQUE',
                                                        MODELISATION    = 'DIS_T',
                                                        GROUP_MA        = 'MCPAK',
                                                        ),
                                );

CARAPAK     = AFFE_CARA_ELEM(   MODELE          = MODELPAK,
                                DISCRET         = _F(   GROUP_MA        = 'MCPAK',
                                                        CARA            = 'K_T_D_L',
                                                        VALE            = (10000.,10000.,10000.),
                                                        ),
                                );

KELPAK      = CALC_MATR_ELEM(   OPTION          = 'RIGI_MECA',
                                MODELE          = MODELPAK,
                                CARA_ELEM       = CARAPAK,
                                );

MELPAK      = CALC_MATR_ELEM(   OPTION          = 'MASS_MECA',
                                MODELE          = MODELPAK,
                                CARA_ELEM       = CARAPAK,
                                );


NUMPAK      = NUME_DDL(         MATR_RIGI       = KELPAK
                                );
```

```
KASSPAK     = ASSE_MATRICE(     MATR_ELEM       = KELPAK,
                                NUME_DDL        = NUMPAK,
                                );


MASSPAK     = ASSE_MATRICE(     MATR_ELEM       = MELPAK,
                                NUME_DDL        = NUMPAK,
                                );


MODEXPAK    = LIRE_RESU(        TYPE_RESU       = 'MODE_MECA',
                                FORMAT          = 'IDEAS',
                                MODELE          = MODELPAK,
                                UNITE           = 30,
                                NOM_CHAM        = 'DEPL',
                                MATR_A          = KASSPAK,
                                MATR_B          = MASSPAK,
                                FORMAT_IDEAS    = _F(   NOM_CHAM        = 'DEPL',
                                                        NUME_DATASET    = 55,
                                                        RECORD_6        = (1,2,2,8,2,3,),
                                                        POSI_ORDRE      = (7,4,),
                                                        POSI_NUME_MODE  = (7,4),
                                                        POSI_FREQ       = (8,1,),
                                                        POSI_MASS_GENE  = (8,2,),
                                                        POSI_AMOR_GENE  = (8,3,),
                                                        NOM_CMP         = ('DX','DY','DZ'),
                                                        ),
                                TOUT_ORDRE      = 'OUI',
                                );

#MODEXPAK    = NORM_MODE(        reuse           = MODEXPAK,
#                                MODE            = MODEXPAK,
#                                NORME           = 'MASS_GENE',
#                                );
MODEXP3D = extract_modes(MODEXPAK, 'PAK Experimental Mode-shapes 3D full','displacement', '', '')
to_file_print(MODEXP3D,'MAYAPAK')

#=======================================================================================================#
#   Importing the expirimental 'maille' for strain gauges and creating the model
#   for the values gives by the strain gauges (LMS)
#=======================================================================================================#

MAYALMS     = LIRE_MAILLAGE(    UNITE           = 21,
                                FORMAT          = 'ASTER'
                                );



MAYALMS     = DEFI_GROUP(       reuse           = MAYALMS,
                                MAILLAGE        = MAYALMS,
                                CREA_GROUP_MA   = _F(   NOM             = 'MCLMS',
                                                        TOUT            = 'OUI',
                                                        ),
                                CREA_GROUP_NO   = _F(   NOM             = 'MDSLMS',
                                                        NOEUD           = ('N1','N2','N3','N4','N5','N…
                                                        ),
                                );

MODELLMS    = AFFE_MODELE(      MAILLAGE        = MAYALMS,
                                VERIF           = 'MAILLE',
                                AFFE            = _F(   PHENOMENE       = 'MECANIQUE',
                                                        MODELISATION    = 'DIS_T',
                                                        GROUP_MA        = 'MCLMS',
                                                        ),
                                );

CARALMS     = AFFE_CARA_ELEM(   MODELE          = MODELLMS,
                                DISCRET         = _F(   GROUP_MA        = 'MCLMS',
                                                        CARA            = 'K_T_D_L',
                                                        VALE            = (10000.,10000.,10000.),
                                                        ),
```

```
                              );

KELLMS      = CALC_MATR_ELEM(   OPTION          = 'RIGI_MECA',
                                MODELE          = MODELLMS,
                                CARA_ELEM       = CARALMS,
                                );

MELLMS      = CALC_MATR_ELEM(   OPTION          = 'MASS_MECA',
                                MODELE          = MODELLMS,
                                CARA_ELEM       = CARALMS,
                                );


NUMLMS      = NUME_DDL(         MATR_RIGI       = KELLMS
                                );


KASSLMS     = ASSE_MATRICE(     MATR_ELEM       = KELLMS,
                                NUME_DDL        = NUMLMS,
                                );


MASSLMS     = ASSE_MATRICE(     MATR_ELEM       = MELLMS,
                                NUME_DDL        = NUMLMS,
                                );


MODEXLMS    = LIRE_RESU(        TYPE_RESU       = 'MODE_MECA',
                                FORMAT          = 'IDEAS',
                                MODELE          = MODELLMS,
                                UNITE           = 31,
                                NOM_CHAM        = 'EPSI_NOEU_DEPL',
                                MATR_A          = KASSLMS,
                                MATR_B          = MASSLMS,
                                FORMAT_IDEAS    = _F(   NOM_CHAM        = 'EPSI_NOEU_DEPL',
                                                        NUME_DATASET    = 55,
                                                        RECORD_6        = (1,2,2,3,2,3,),
                                                        POSI_ORDRE      = (7,4,),
                                                        POSI_NUME_MODE  = (7,4),
                                                        POSI_FREQ       = (8,1,),
                                                        POSI_MASS_GENE  = (8,2),
                                                        POSI_AMOR_GENE  = (8,3),
                                                        NOM_CMP         = ('EPXX','EPYY','EPZZ',),#'EP…
                                                        ),
                                TOUT_ORDRE      = 'OUI',
                                );
#IMPR_CO        (       CONCEPT                         = _F(   NOM     = MAYALMS,
#                                                                       ),
#                       UNITE                           = 8,);
#
#MODEXL3D = extract_modes(MODEXLMS, 'LMS Experimental Mode-shapes 3D full', 'strain', '', '')
#to_file_print(MODEXL3D,'MAYALMS')

#=================================================================================================#
#   Importing the expirimental 'maille' in fluid so in 2-directions (x z) for strain gauges and
#   creating the model for the values gives by the strain gauges in fluid (in operation!) (LMS)
#=================================================================================================#

MODEXFLU    = LIRE_RESU(        TYPE_RESU       = 'MODE_MECA',
                                FORMAT          = 'IDEAS',
                                MODELE          = MODELLMS,
                                UNITE           = 32,
                                NOM_CHAM        = 'EPSI_NOEU_DEPL',
                                MATR_A          = KASSLMS,
                                MATR_B          = MASSLMS,
                                FORMAT_IDEAS    = _F(   NOM_CHAM        = 'EPSI_NOEU_DEPL',
                                                        NUME_DATASET    = 55,
                                                        RECORD_6        = (1,2,2,3,2,3,),
                                                        POSI_ORDRE      = (7,4,),
                                                        POSI_NUME_MODE  = (7,4),
                                                        POSI_FREQ       = (8,1,),
                                                        POSI_MASS_GENE  = (8,2),
```

```
                                                POSI_AMOR_GENE  = (8,3),
                                                NOM_CMP         = ('EPYY','EPXX','EPZZ',)#'EPX…
                                                # Pay attention, in Data_LMS_Fluid.unv, the or…
                                                # is different!!!
                                                ),
                                TOUT_ORDRE      = 'OUI',
                                );

#=================================================================================================#
#   Now, to be able to trick CALC_ESSAI in using strain, we need to read the file as if it were
#   displacements...
#=================================================================================================#

MODEXFDX    = LIRE_RESU(        TYPE_RESU       = 'MODE_MECA',
                                FORMAT          = 'IDEAS',
                                MODELE          = MODELLMS,
                                UNITE           = 32,
                                NOM_CHAM        = 'DEPL',
                                MATR_A          = KASSLMS,
                                MATR_B          = MASSLMS,
                                FORMAT_IDEAS    = _F(   NOM_CHAM        = 'DEPL',
                                                        NUME_DATASET    = 55,
                                                        RECORD_6        = (1,2,2,3,2,3,),
                                                        POSI_ORDRE      = (7,4,),
                                                        POSI_NUME_MODE  = (7,4),
                                                        POSI_FREQ       = (8,1,),
                                                        POSI_MASS_GENE  = (8,2),
                                                        POSI_AMOR_GENE  = (8,3),
                                                        NOM_CMP         = ('DX','DY','DZ',)#'EPXY','EP…
                                                        # Pay attention, doing it like this will make …
                                                        # to be in the DX-coordinate.
                                                        ),
                                TOUT_ORDRE      = 'OUI',
                                );
#-------------------------------------------------------------------------------------------------#
#   And to make the nomalisation right, we need to ude NORM_MODE, thus we should be happy that
#   CALC_ESSAI needs to be tricked, because with strain measuments this is not possible...
#-------------------------------------------------------------------------------------------------#


MODEXFDX    = NORM_MODE(        reuse           = MODEXFDX,
                                MODE            = MODEXFDX,
                                NORME           = 'MASS_GENE',
                                );


#=================================================================================================#
#   MAC-ing the modeshapes, to see whether the results are orthogonal
#=================================================================================================#

basePAK = extract_modes(MODEXPAK, 'PAK Experimental Mode-shapes', 'displacement', '', '')
mac_modes(basePAK,basePAK)
baseLMS = extract_modes(MODEXLMS, 'LMS Experimental Mode-shapes', 'strain', '', '')
mac_modes(baseLMS,baseLMS)
baseLFL = extract_modes(MODEXFLU, 'LMS Fluid Experimental Mode-shapes', 'strain', '', '')
mac_modes(baseLFL,baseLFL)

MACPAK      = MAC_MODES(        BASE_1          = MODEXPAK,
                                BASE_2          = MODEXPAK,
                                INFO            = 2,
                                );
CALC_ESSAI( UNITE_RESU = 8,
            INTERACTIF = 'OUI',
            );

#=================================================================================================#
FIN();
#=================================================================================================#
```

```
#=================================================================================================#
#   Making the model expansion.
#
#   The model expamsions are made threefold; First the PAK experimental dta is expanded on both
#   the 3D and 2D model, then the LMS data is expanded, also on both models, and finally, the LMS
#   data in fluid is expanded on the 2D and 3D models. This last data is of course the only
#   non-mass normalised data.
#=================================================================================================#
#=================================================================================================#
#   Begin
#=================================================================================================#

import sys
import Numeric
sys.path.append('/home/joost/Bureaublad/Study_EDF/python_files/')
from modes import *
from nodes_2D_vs_3D import *
from to_file_print import *

POURSUITE(PAR_LOT = 'NON');

#   Obtaining the nodes to be able to compare the 2D and 3D model later on...
noeuds3D, noeuds2D, index3D, index2D = nodes_2D_vs_3D();

#-------------------------------------------------------------------------------------------------#
# To pick specific modeshapes from the first 14 determinted,change the sequence of 'NUME_MODE'
#-------------------------------------------------------------------------------------------------#
MODEA3D     = EXTR_MODE(        FILTRE_MODE     = ( _F( MODE            = MODEA3Dt,
                                                        NUME_ORDRE      = (1,2,3,4,5,6,7,8,),
#                                                       NUME_ORDRE      = (1,2,3,4,5,6,7,8,9,10,11,12,…
                                                        ),
#                                                    _F( MODE            = MODEA3Dt,
#                                                        NUME_MODE       = (2,4,6,8,),
#                                                        ),
                                                  ),
                                );
#-------------------------------------------------------------------------------------------------#

DETRUIRE(                       CONCEPT         = _F(   NOM             = MODEA3Dt,
                                                        ),
                                INFO            = 1,
                                );
#=================================================================================================#
#   Calculating the strain from the displacements of the MODESHAPES of MODEAIR!!!
#   (using first the strain in the elements: EPSI_ELNO_DEPL and than the strain at the
#   nodes: EPSI_NOEU_DEPL.
#=================================================================================================#

MODEA3D     = CALC_ELEM(        reuse           = MODEA3D,
                                RESULTAT        = MODEA3D,
                                GROUP_MA        = 'TU_MA3D',
                                OPTION          = 'EPSI_ELNO_DEPL'
                                );

MODEA3D     = CALC_NO(          reuse           = MODEA3D,
                                RESULTAT        = MODEA3D,
                                GROUP_MA        = 'TU_MA3D',
                                OPTION          = 'EPSI_NOEU_DEPL'
                                );


#-------------------------------------------------------------------------------------------------#
# To pick specific modeshapes from the first 14 determinted,change the sequence of 'NUME_MODE'
#-------------------------------------------------------------------------------------------------#
MODEA2D     = EXTR_MODE(        FILTRE_MODE     = ( _F( MODE            = MODEA2Dt,
                                                        NUME_ORDRE      = (1,2,3,4,5,6,7,8,),
#                                                       NUME_ORDRE      = (1,2,3,4,5,6,7,8,9,10,11,12…
                                                        ),
#                                                    _F( MODE            = MODEA2Dt,
#                                                        NUME_MODE       = (1,3,5,7,),
#                                                        ),
                                                  ),
```

```
                              );
#-------------------------------------------------------------------------------------------------#

DETRUIRE(   CONCEPT             = _F(           NOM     = MODEA2Dt,
                                                ),
                                  INFO          = 1,
            );
#=================================================================================================#
#   Calculating the strain from the displacements of the MODESHAPES of MODEAIR!!!
#   (using first the strain in the elements: EPSI_ELNO_DEPL and than the strain at the
#   nodes: EPSI_NOEU_DEPL.
#=================================================================================================#

MODEA2D     = CALC_ELEM(        reuse           = MODEA2D,
                                RESULTAT        = MODEA2D,
                                GROUP_MA        = 'TU_MA2D',
                                OPTION          = 'EPSI_ELNO_DEPL'
                                );

MODEA2D     = CALC_NO(          reuse           = MODEA2D,
                                RESULTAT        = MODEA2D,
                                GROUP_MA        = 'TU_MA2D',
                                OPTION          = 'EPSI_NOEU_DEPL'
                                );

#=================================================================================================#
#   Making the model expansion of the PAK data to creat a nice shape of the expirimental data on
#   the 3D numerical model and afterwards on the 2D numerical model.
#   First on the 3D model.
#=================================================================================================#

REGENP3D    = PROJ_MESU_MODAL(  MODELE_CALCUL   = _F(   BASE            = MODEA3D,
                                                        MODELE          = MODELN3D,
                                                        ),
                                MODELE_MESURE   = _F(   MESURE          = MODEXPAK,
                                                        MODELE          = MODELPAK,
                                                        NOM_CHAM        = 'DEPL',
                                                        ),
                                RESOLUTION      = _F(   METHODE         = 'SVD',
                                                        EPS             = 1.0E-5,
                                                        ),
                                NOM_PARA        = (     'AMOR_GENE','MASS_GENE','AMOR_REDUIT'
                                                        ),
                                );

RESETP3D    = REST_GENE_PHYS(   RESU_GENE       = REGENP3D,
                                TOUT_ORDRE      = 'OUI',
                                TOUT_CHAM       = 'OUI',
                                );
#=================================================================================================#
#   Static mode-shapes on the 3D model.
#=================================================================================================#

#-------------------------------------------------------------------------------------------------#
#   On accelerometer positions
#-------------------------------------------------------------------------------------------------#

REGSTP3a    = PROJ_MESU_MODAL(  MODELE_CALCUL   = _F(   BASE            = MODSTA3a,
                                                        MODELE          = MODELN3D,
                                                        ),
                                MODELE_MESURE   = _F(   MESURE          = MODEXPAK,
                                                        MODELE          = MODELPAK,
                                                        NOM_CHAM        = 'DEPL',
                                                        ),
                                RESOLUTION      = _F(   METHODE         = 'SVD',
                                                        EPS             = 1.0E-5,
                                                        ),
                                NOM_PARA        = (     'AMOR_GENE','MASS_GENE','AMOR_REDUIT'
                                                        ),
                                );

RESSTP3a    = REST_GENE_PHYS(   RESU_GENE       = REGSTP3a,
```

```
                              TOUT_ORDRE      = 'OUI',
                              TOUT_CHAM       = 'OUI',
                              );
#-----------------------------------------------------------------------------------------------------#
#   On spring positions
#-----------------------------------------------------------------------------------------------------#

REGSTP3s    = PROJ_MESU_MODAL(  MODELE_CALCUL   = _F(   BASE            = MODSTA3s,
                                                        MODELE          = MODELN3D,
                                                        ),
                              MODELE_MESURE   = _F(   MESURE          = MODEXPAK,
                                                        MODELE          = MODELPAK,
                                                        NOM_CHAM        = 'DEPL',
                                                        ),
                              RESOLUTION      = _F(   METHODE         = 'SVD',
                                                        EPS             = 1.0E-5,
                                                        ),
                              NOM_PARA        = (     'AMOR_GENE','MASS_GENE','AMOR_REDUIT'
                                                        ),
                              );

RESSTP3s    = REST_GENE_PHYS(   RESU_GENE       = REGSTP3s,
                              TOUT_ORDRE      = 'OUI',
                              TOUT_CHAM       = 'OUI',
                              );

#=====================================================================================================#
#   Calculating the strain from the displacements (using first the strain in the
#   elements: EPSI_ELNO_DEPL and than the strain at the nodes: EPSI_NOEU_DEPL.
#=====================================================================================================#

RESETP3D    = CALC_ELEM(        reuse           = RESETP3D,
                              RESULTAT        = RESETP3D,
                              GROUP_MA        = 'TU_MA3D',
                              OPTION          = 'EPSI_ELNO_DEPL'
                              );

RESETP3D    = CALC_NO(          reuse           = RESETP3D,
                              RESULTAT        = RESETP3D,
                              GROUP_MA        = 'TU_MA3D',
                              OPTION          = 'EPSI_NOEU_DEPL'
                              );

#IMPR_GENE(                     UNITE           = 8,
#                               GENE            = _F(   RESU_GENE   = REGENP3D,
#                                                       ),
#                               );
#
IMPR_RESU(                    UNITE           = 80,
                              FORMAT          = 'MED',
                              RESU            = _F(   MAILLAGE    = MAIN3D,
                                                        RESULTAT    = RESETP3D,
                                                        ),
                              );

IMPR_RESU(                    UNITE           = 81,
                              FORMAT          = 'MED',
                              RESU            = _F(   MAILLAGE    = MAIN3D,
                                                        RESULTAT    = RESSTP3s,
                                                        ),
                              );

IMPR_RESU(                    UNITE           = 82,
                              FORMAT          = 'MED',
                              RESU            = _F(   MAILLAGE    = MAIN3D,
                                                        RESULTAT    = RESSTP3a,
                                                        ),
                              );

#=====================================================================================================#
#   And, as last, on the 2D numerical model.
#=====================================================================================================#
```

```
REGENP2D    = PROJ_MESU_MODAL(  MODELE_CALCUL   = _F(   BASE            = MODEA2D,
                                                        MODELE          = MODELN2D,
                                                        ),
                                MODELE_MESURE   = _F(   MESURE          = MODEXPAK,
                                                        MODELE          = MODELPAK,
                                                        NOM_CHAM        = 'DEPL',
                                                        ),
                                RESOLUTION      = _F(   METHODE         = 'SVD',
                                                        EPS             = 1.0E-5,
                                                        ),
                                NOM_PARA        = (     'AMOR_GENE','MASS_GENE','AMOR_REDUIT'
                                                        ),
                                );

RESETP2D    = REST_GENE_PHYS(   RESU_GENE       = REGENP2D,
                                TOUT_ORDRE      = 'OUI',
                                TOUT_CHAM       = 'OUI',
                                );
#=================================================================================================#
#   Static mode-shapes on the 2D model.
#=================================================================================================#

#-------------------------------------------------------------------------------------------------#
#   On accelerometer positions
#-------------------------------------------------------------------------------------------------#

REGSTP2a    = PROJ_MESU_MODAL(  MODELE_CALCUL   = _F(   BASE            = MODSTA2a,
                                                        MODELE          = MODELN2D,
                                                        ),
                                MODELE_MESURE   = _F(   MESURE          = MODEXPAK,
                                                        MODELE          = MODELPAK,
                                                        NOM_CHAM        = 'DEPL',
                                                        ),
                                RESOLUTION      = _F(   METHODE         = 'SVD',
                                                        EPS             = 1.0E-5,
                                                        ),
                                NOM_PARA        = (     'AMOR_GENE','MASS_GENE','AMOR_REDUIT'
                                                        ),
                                );

RESSTP2a    = REST_GENE_PHYS(   RESU_GENE       = REGSTP2a,
                                TOUT_ORDRE      = 'OUI',
                                TOUT_CHAM       = 'OUI',
                                );
#-------------------------------------------------------------------------------------------------#
#   On spring positions
#-------------------------------------------------------------------------------------------------#

REGSTP2s    = PROJ_MESU_MODAL(  MODELE_CALCUL   = _F(   BASE            = MODSTA2s,
                                                        MODELE          = MODELN2D,
                                                        ),
                                MODELE_MESURE   = _F(   MESURE          = MODEXPAK,
                                                        MODELE          = MODELPAK,
                                                        NOM_CHAM        = 'DEPL',
                                                        ),
                                RESOLUTION      = _F(   METHODE         = 'SVD',
                                                        EPS             = 1.0E-5,
                                                        ),
                                NOM_PARA        = (     'AMOR_GENE','MASS_GENE','AMOR_REDUIT'
                                                        ),
                                );

RESSTP2s    = REST_GENE_PHYS(   RESU_GENE       = REGSTP2s,
                                TOUT_ORDRE      = 'OUI',
                                TOUT_CHAM       = 'OUI',
                                );

#=================================================================================================#
#   Calculating the strain from the displacements (using first the strain in the
#   elements: EPSI_ELNO_DEPL and than the strain at the nodes: EPSI_NOEU_DEPL.
#=================================================================================================#
```

```
RESETP2D    = CALC_ELEM(        reuse           = RESETP2D,
                                RESULTAT        = RESETP2D,
                                GROUP_MA        = 'TU_MA2D',
                                OPTION          = 'EPSI_ELNO_DEPL'
                                );

RESETP2D    = CALC_NO(          reuse           = RESETP2D,
                                RESULTAT        = RESETP2D,
                                GROUP_MA        = 'TU_MA2D',
                                OPTION          = 'EPSI_NOEU_DEPL'
                                );
#IMPR_GENE(                     UNITE           = 9,
#                               GENE            = _F(   RESU_GENE   = REGENP2D,
#                                                       ),
#                               );
#
#IMPR_RESU(                     UNITE           = 80,
#                               FORMAT          = 'MED',
#                               RESU            = _F(   MAILLAGE    = MAIN2D,
#                                                       RESULTAT    = RESETP2D,
#                                                       ),
#                               );

#=================================================================================================#
#   Making the model expansion of the LMS data to creat a nice shape of the expirimental data on
#   the 3D numerical model and afterwards on the 2D numerical model.
#   First on the 3D model.
#=================================================================================================#

REGENL3D    = PROJ_MESU_MODAL(  MODELE_CALCUL   = _F(   BASE            = MODEA3D,
                                                        MODELE          = MODELN3D,
                                                        ),
                                MODELE_MESURE   = _F(   MESURE          = MODEXLMS,
                                                        MODELE          = MODELLMS,
                                                        NOM_CHAM        = 'EPSI_NOEU_DEPL',
                                                        ),
                                RESOLUTION      = _F(   METHODE         = 'SVD',
                                                        EPS             = 1.0E-5,
                                                        ),
                                NOM_PARA        = (     'AMOR_GENE','MASS_GENE','AMOR_REDUIT'
                                                        ),
                                );

#-------------------------------------------------------------------------------------------------
#   Using the just calculated generilised modeshape factors (alpha) to calculated the shape in
#   displacement. Afterwards these ones are recalculated in strains again. This indirect
#   methode is the only posibility!
#-------------------------------------------------------------------------------------------------

RESETL3D    = REST_GENE_PHYS(   RESU_GENE       = REGENL3D,
                                TOUT_ORDRE      = 'OUI',
                                TOUT_CHAM       = 'OUI',
                                );

RESETL3D    = CALC_ELEM(        reuse           = RESETL3D,
                                RESULTAT        = RESETL3D,
                                GROUP_MA        = 'TU_MA3D',
                                OPTION          = 'EPSI_ELNO_DEPL'
                                );

RESETL3D     = CALC_NO(         reuse           = RESETL3D,
                                RESULTAT        = RESETL3D,
                                GROUP_MA        = 'TU_MA3D',
                                OPTION          = 'EPSI_NOEU_DEPL'
                                );

IMPR_RESU(                      UNITE           = 83,
                                FORMAT          = 'MED',
                                RESU            = _F(   MAILLAGE    = MAIN3D,
                                                        RESULTAT    = RESETL3D,
                                                        ),
```

```
                                );

#=================================================================================================#
#   And, as last, on the 2D numerical model.
#=================================================================================================#

REGENL2D    = PROJ_MESU_MODAL(  MODELE_CALCUL   = _F(   BASE            = MODEA2D,
                                                        MODELE          = MODELN2D,
                                                        ),
                                MODELE_MESURE   = _F(   MESURE          = MODEXLMS,
                                                        MODELE          = MODELLMS,
                                                        NOM_CHAM        = 'EPSI_NOEU_DEPL',
                                                        ),
                                RESOLUTION      = _F(   METHODE         = 'SVD',
                                                        EPS             = 1.0E-5,
                                                        ),
                                NOM_PARA        = (     'AMOR_GENE','MASS_GENE','AMOR_REDUIT'
                                                        ),
                                );

#-------------------------------------------------------------------------------------------------
#   Using the just calculated generilised modeshape factors (alpha) to calculated the shape in
#   displacement. After 'MAC-ing' these ones are re-writen as strains again. This indirect
#   methode is the only posibility!
#-------------------------------------------------------------------------------------------------

RESETL2D    = REST_GENE_PHYS(   RESU_GENE       = REGENL2D,
                                TOUT_ORDRE      = 'OUI',
                                TOUT_CHAM       = 'OUI',
                                );

RESETL2D    = CALC_ELEM(        reuse           = RESETL2D,
                                RESULTAT        = RESETL2D,
                                GROUP_MA        = 'TU_MA2D',
                                OPTION          = 'EPSI_ELNO_DEPL'
                                );

RESETL2D     = CALC_NO(         reuse           = RESETL2D,
                                RESULTAT        = RESETL2D,
                                GROUP_MA        = 'TU_MA2D',
                                OPTION          = 'EPSI_NOEU_DEPL'
                                );
#=================================================================================================#
#   Making the model expansion of the LMS data in fluid, to creat a nice shape of the expirimental
#   data on the 3D numerical model and afterwards on the 2D numerical model.
#   First on the 3D model.
#=================================================================================================#

REGENF3D    = PROJ_MESU_MODAL(  MODELE_CALCUL   = _F(   BASE            = MODEA3D,
                                                        MODELE          = MODELN3D,
                                                        ),
                                MODELE_MESURE   = _F(   MESURE          = MODEXFLU,
                                                        MODELE          = MODELLMS,
                                                        NOM_CHAM        = 'EPSI_NOEU_DEPL',
                                                        ),
                                RESOLUTION      = _F(   METHODE         = 'SVD',
                                                        EPS             = 1.0E-5,
                                                        ),
                                NOM_PARA        = (     'AMOR_GENE','MASS_GENE','AMOR_REDUIT'
                                                        ),
                                );

#-------------------------------------------------------------------------------------------------
#   Using the just calculated generilised modeshape factors (alpha) to calculated the shape in
#   displacement. Afterwards these ones are recalculated in strains again. This indirect
#   methode is the only posibility!
#-------------------------------------------------------------------------------------------------

RESETF3D    = REST_GENE_PHYS(   RESU_GENE       = REGENF3D,
                                TOUT_ORDRE      = 'OUI',
                                TOUT_CHAM       = 'OUI',
                                );
```

```
RESETF3D    = NORM_MODE(        reuse           = RESETF3D,
                                MODE            = RESETF3D,
                                NORME           = 'MASS_GENE',
                                );

RESETF3D    = CALC_ELEM(        reuse           = RESETF3D,
                                RESULTAT        = RESETF3D,
                                GROUP_MA        = 'TU_MA3D',
                                OPTION          = 'EPSI_ELNO_DEPL'
                                );

RESETF3D     = CALC_NO(         reuse           = RESETF3D,
                                RESULTAT        = RESETF3D,
                                GROUP_MA        = 'TU_MA3D',
                                OPTION          = 'EPSI_NOEU_DEPL'
                                );

IMPR_RESU(                      UNITE           = 84,
                                FORMAT          = 'MED',
                                RESU            = _F(   MAILLAGE    = MAIN3D,
                                                        RESULTAT    = RESETF3D,
                                                        ),
                                );

#=================================================================================================#
#   And, as last, on the 2D numerical model.
#=================================================================================================#

REGENF2D    = PROJ_MESU_MODAL(  MODELE_CALCUL   = _F(   BASE            = MODEA2D,
                                                        MODELE          = MODELN2D,
                                                        ),
                                MODELE_MESURE   = _F(   MESURE          = MODEXFLU,
                                                        MODELE          = MODELLMS,
                                                        NOM_CHAM        = 'EPSI_NOEU_DEPL',
                                                        ),
                                RESOLUTION      = _F(   METHODE         = 'SVD',
                                                        EPS             = 1.0E-5,
                                                        ),
                                NOM_PARA        = (     'AMOR_GENE','MASS_GENE','AMOR_REDUIT'
                                                        ),
                                );

#-------------------------------------------------------------------------------------------------
#   Using the just calculated generilised modeshape factors (alpha) to calculated the shape in
#   displacement. After 'MAC-ing' these ones are re-writen as strains again. This indirect
#   methode is the only posibility!
#-------------------------------------------------------------------------------------------------

RESETF2D    = REST_GENE_PHYS(   RESU_GENE       = REGENF2D,
                                TOUT_ORDRE      = 'OUI',
                                TOUT_CHAM       = 'OUI',
                                );

RESETF2D    = NORM_MODE(        reuse           = RESETF2D,
                                MODE            = RESETF2D,
                                NORME           = 'MASS_GENE',
                                );

RESETF2D    = CALC_ELEM(        reuse           = RESETF2D,
                                RESULTAT        = RESETF2D,
                                GROUP_MA        = 'TU_MA2D',
                                OPTION          = 'EPSI_ELNO_DEPL'
                                );

RESETF2D     = CALC_NO(         reuse           = RESETF2D,
                                RESULTAT        = RESETF2D,
                                GROUP_MA        = 'TU_MA2D',
                                OPTION          = 'EPSI_NOEU_DEPL'
                                );

#=================================================================================================#
```

```
#   MAC-ing the modeshapes, to see whether the results are orthogonal
#=============================================================================================================#
MACREP3D    = MAC_MODES(        BASE_1          = RESETP3D,
                                BASE_2          = RESETP3D,
                                INFO            = 2,
                                );
MACSTP3a    = MAC_MODES(        BASE_1          = RESSTP3a,
                                BASE_2          = RESSTP3a,
                                INFO            = 2,
                                );
MACSTP3s    = MAC_MODES(        BASE_1          = RESSTP3s,
                                BASE_2          = RESSTP3s,
                                INFO            = 2,
                                );
MACREP2D    = MAC_MODES(        BASE_1          = RESETP2D,
                                BASE_2          = RESETP2D,
                                INFO            = 2,
                                );
MACSTP2a    = MAC_MODES(        BASE_1          = RESSTP2a,
                                BASE_2          = RESSTP2a,
                                INFO            = 2,
                                );
MACSTP2s    = MAC_MODES(        BASE_1          = RESSTP2s,
                                BASE_2          = RESSTP2s,
                                INFO            = 2,
                                );
baPAK3Dd = extract_modes(RESSTP3a, 'PAK experimental mode-shapes in air, expanded on static-shapes of …
mac_modes(baPAK3Dd,baPAK3Dd)
baPAK3Dd = extract_modes(RESSTP3s, 'PAK experimental mode-shapes in air, expanded on static-shapes of …
mac_modes(baPAK3Dd,baPAK3Dd)
baPAK3Dd = extract_modes(RESETP3D, 'PAK experimental mode-shapes in air, expanded on mode-shapes of 3D…
mac_modes(baPAK3Dd,baPAK3Dd)

baPAK2Dd = extract_modes(RESSTP2a, 'PAK experimental mode-shapes in air, expanded on static-shapes of …
mac_modes(baPAK2Dd,baPAK2Dd)
baPAK2Dd = extract_modes(RESSTP2s, 'PAK experimental mode-shapes in air, expanded on static-shapes of …
mac_modes(baPAK2Dd,baPAK2Dd)
baPAK2Dd = extract_modes(RESETP2D, 'PAK experimental mode-shapes in air, expanded on mode-shapes of 2D…
mac_modes(baPAK2Dd,baPAK2Dd)
rbPAK3Dd = extract_modes(RESETP3D, 'Reduced PAK experimental mode-shapes expanded on mde-shapes of 3D …
mac_modes(rbPAK3Dd,rbPAK3Dd)
rbPAK2Dd = extract_modes(RESETP2D, 'Reduced PAK experimental mode-shapes expanded on mode-shapes of 2D…
mac_modes(rbPAK2Dd,rbPAK2Dd)
mac_modes(rbPAK3Dd,rbPAK2Dd)
MACREL3D    = MAC_MODES(        BASE_1          = RESETL3D,
                                BASE_2          = RESETL3D,
                                INFO            = 2,
                                );
MACREL2D    = MAC_MODES(        BASE_1          = RESETL2D,
                                BASE_2          = RESETL2D,
                                INFO            = 2,
                                );
baLMS3Dd = extract_modes(RESETL3D, 'LMS experimental mode-shapes in air, expanded on mode-shapes of 3D…
mac_modes(baLMS3Dd,baLMS3Dd)
baLMS2Dd = extract_modes(RESETL2D, 'LMS experimental mode-shapes in air, expanded on mode-shapes of 2D…
mac_modes(baLMS2Dd,baLMS2Dd)
rbLMS3Dd = extract_modes(RESETL3D, 'Reduced LMS experimental mode-shapes in air, expanded on mode-shap…
mac_modes(rbLMS3Dd,rbLMS3Dd)
rbLMS2Dd = extract_modes(RESETL2D, 'Reduced LMS experimental mode-shapes in air, expanded on mode-shap…
mac_modes(rbLMS2Dd,rbLMS2Dd)
mac_modes(rbLMS3Dd,rbLMS2Dd)
MACREF3D    = MAC_MODES(        BASE_1          = RESETF3D,
                                BASE_2          = RESETF3D,
                                INFO            = 2,
                                );
MACREF2D    = MAC_MODES(        BASE_1          = RESETF2D,
                                BASE_2          = RESETF2D,
                                INFO            = 2,
                                );
baLFL3Dd = extract_modes(RESETF3D, 'LMS experimental mode-shapes in water, expanded on mode-shapes of …
mac_modes(baLFL3Dd,baLFL3Dd)
baLFL2Dd = extract_modes(RESETF2D, 'LMS experimental mode-shapes in water, expanded on mode-shapes of …
```

```
mac_modes(baLFL2Dd,baLFL2Dd)
rbLFL3Dd = extract_modes(RESETF3D, 'Reduced LMS experimental mode-shapes in water, expanded on mode-sh…
mac_modes(rbLFL3Dd,rbLFL3Dd)
rbLFL2Dd = extract_modes(RESETF2D, 'Reduced LMS experimental mode-shapes in water, expanded on mode-sh…
mac_modes(rbLFL2Dd,rbLFL2Dd)
mac_modes(rbLFL3Dd,rbLFL2Dd)


baPAK3Ds = extract_modes(RESETP3D, 'PAK experimental mode-shapes in air, expanded on mode-shapes of 3D…
mac_modes(baPAK3Ds,baPAK3Ds)
to_file_print(baPAK3Ds,'MAINtmp3')
baPAK2Ds = extract_modes(RESETP2D, 'PAK experimental mode-shapes in air, expanded on mode-shapes of 2D…
mac_modes(baPAK2Ds,baPAK2Ds)
to_file_print(baPAK2Ds,'MAINtmp2')
rbPAK3Ds = extract_modes(RESETP3D, 'Reduced PAK experimental mode-shapes in air, expanded on mode-shap…
mac_modes(rbPAK3Ds,rbPAK3Ds)
rbPAK2Ds = extract_modes(RESETP2D, 'Reduced PAK experimental mode-shapes in air, expanded on mode-shap…
mac_modes(rbPAK2Ds,rbPAK2Ds)
mac_modes(rbPAK3Ds,rbPAK2Ds)

baLMS3Ds = extract_modes(RESETL3D, 'LMS experimental mode-shapes in air, expanded on mode-shapes of 3D…
mac_modes(baLMS3Ds,baLMS3Ds)
baLMS2Ds = extract_modes(RESETL2D, 'LMS experimental mode-shapes in air, expanded on mode-shapes of 2D…
mac_modes(baLMS2Ds,baLMS2Ds)
rbLMS3Ds = extract_modes(RESETL3D, 'Reduced LMS experimental mode-shapes in air, expanded on mode-shap…
mac_modes(rbLMS3Ds,rbLMS3Ds)
rbLMS2Ds = extract_modes(RESETL2D, 'Reduced LMS experimental mode-shapes in air, expanded on mode-shap…
mac_modes(rbLMS2Ds,rbLMS2Ds)
mac_modes(rbLMS3Ds,rbLMS2Ds)

baLFL3Ds = extract_modes(RESETF3D, 'LMS experimental mode-shapes in water, expanded on mode-shapes of …
mac_modes(baLFL3Ds,baLFL3Ds)
baLFL2Ds = extract_modes(RESETF2D, 'LMS experimental mode-shapes in water, expanded on mode-shapes of …
mac_modes(baLFL2Ds,baLFL2Ds)
rbLFL3Ds = extract_modes(RESETF3D, 'Reduced LMS experimental mode-shapes in water, expanded on mode-sh…
mac_modes(rbLFL3Ds,rbLFL3Ds)
rbLFL2Ds = extract_modes(RESETF2D, 'Reduced LMS experimental mode-shapes in water, expanded on mode-sh…
mac_modes(rbLFL2Ds,rbLFL2Ds)
mac_modes(rbLFL3Ds,rbLFL2Ds)

mac_modes(baLFL2Ds,baPAK2Ds)
mac_modes(baLFL3Ds,baPAK3Ds)
mac_modes(baLFL2Dd,baPAK2Dd)
mac_modes(baLFL3Dd,baPAK3Dd)
mac_modes(baLFL2Ds,baLMS2Ds)
mac_modes(baLFL3Ds,baLMS3Ds)
mac_modes(baLFL2Dd,baLMS2Dd)
mac_modes(baLFL3Dd,baLMS3Dd)

mac_modes(baPAK3Dd,baLMS3Dd)
mac_modes(baPAK3Ds,baLMS3Ds)

#=================================================================================================#
#   Expanding on already expanded experimental mode-shapes
#=================================================================================================#
REXP2L3D    = PROJ_MESU_MODAL(  MODELE_CALCUL   = _F(   BASE            = RESETP3D,
                                                        MODELE          = MODELN3D,
                                                        ),
                                MODELE_MESURE   = _F(   MESURE          = MODEXLMS,
                                                        MODELE          = MODELLMS,
                                                        NOM_CHAM        = 'EPSI_NOEU_DEPL',
                                                        ),
                                RESOLUTION      = _F(   METHODE         = 'SVD',
                                                        EPS             = 1.0E-5,
                                                        ),
                                NOM_PARA        = (     'AMOR_GENE','MASS_GENE','AMOR_REDUIT'
                                                        ),
                                );

RESP2L3D    = REST_GENE_PHYS(   RESU_GENE       = REXP2L3D,
                                TOUT_ORDRE      = 'OUI',
                                TOUT_CHAM       = 'OUI',
```

```
                                );

RESP2L3D    = CALC_ELEM(        reuse           = RESP2L3D,
                                RESULTAT        = RESP2L3D,
                                GROUP_MA        = 'TU_MA3D',
                                OPTION          = 'EPSI_ELNO_DEPL'
                                );

RESP2L3D    = CALC_NO(          reuse           = RESP2L3D,
                                RESULTAT        = RESP2L3D,
                                GROUP_MA        = 'TU_MA3D',
                                OPTION          = 'EPSI_NOEU_DEPL'
                                );

baLMS3Ds = extract_modes(RESP2L3D, 'LMS experimental mode-shapes in air, expanded on earlier expanded …
mac_modes(baLMS3Ds,baLMS3Ds)

IMPR_GENE( UNITE=8, GENE=_F(RESU_GENE=REGENP3D))
IMPR_GENE( UNITE=8, GENE=_F(RESU_GENE=REGENL3D))
IMPR_GENE( UNITE=8, GENE=_F(RESU_GENE=REXP2L3D))

CALC_ESSAI( UNITE_RESU = 8,
            INTERACTIF = 'OUI',
            );


#=================================================================================================#
FIN();
#=================================================================================================#
```

```
#=================================================================================================#
#   Identifying the Forces.
#
#   The identification process is done in CALC_ESSAI. Before that a model must be made representing
#   the point forces, modelling the fluid force.
#
#
#=================================================================================================#
#=================================================================================================#
#   Begin
#=================================================================================================#

POURSUITE(      PAR_LOT         = 'NON',
#                IMPR_MACRO      = 'OUI',
                );
#=================================================================================================#
#   Building the 'Forces model', with this model one places the modelling forces
#   on certain possitions
#=================================================================================================#

#PRE_IDEAS(      UNITE_IDEAS     = 30,
#                UNITE_MAILLAGE  = 20,
#                );

MAYAFOR     = LIRE_MAILLAGE(    UNITE           = 20
                                );

MAYAFOR     = DEFI_GROUP(       reuse           = MAYAFOR,
                                MAILLAGE        = MAYAFOR,
                                CREA_GROUP_MA   = _F(   NOM             = 'FMC',
                                                        TOUT            = 'OUI',
                                                        ),
                                CREA_GROUP_NO   = (_F(   NOM             = 'FAP1',
                                                        NOEUD           = 'N1'
                                                        ),
                                                   _F(   NOM             = 'FAP2',
                                                        NOEUD           = 'N2'
                                                        ),
                                                   _F(   NOM             = 'FAP3',
                                                        NOEUD           = 'N3'
                                                        ),
                                                    ),
                                );

MODELFOR    = AFFE_MODELE(      MAILLAGE        = MAYAFOR,
                                AFFE            = _F(   TOUT            = 'OUI',
                                                        PHENOMENE       = 'MECANIQUE',
                                                        MODELISATION    = 'DIS_T',
                                                        ),
                                );

CARAFOR     = AFFE_CARA_ELEM(   MODELE          = MODELFOR,
                                DISCRET         = _F(   GROUP_MA        = 'FMC',
                                                        CARA            = 'K_T_D_L',
                                                        VALE            = ( 100000.,
                                                            100000.,
                                                            100000.,
                                                            ),
                                                        ),
                                );

MATFOR      = AFFE_MATERIAU(    MAILLAGE        = MAYAFOR,
                                MODELE          = MODELFOR,
                                AFFE            = _F(   GROUP_MA        = 'FMC',
                                                        MATER           = MATCRAY,
                                                        ),
                                );


KELFOR      = CALC_MATR_ELEM(   OPTION          = 'RIGI_MECA',
                                MODELE          = MODELFOR,
                                CARA_ELEM       = CARAFOR,
```

```
                              );

MELFOR      = CALC_MATR_ELEM(   OPTION          = 'MASS_MECA',
                                MODELE          = MODELFOR,
                                CARA_ELEM       = CARAFOR,
                                );

NUMFOR      = NUME_DDL(         MATR_RIGI       = KELFOR,
                                );

KASSFOR     = ASSE_MATRICE(     MATR_ELEM       = KELFOR,
                                NUME_DDL        = NUMFOR,
                                );


MASSFOR     = ASSE_MATRICE(     MATR_ELEM       = MELFOR,
                                NUME_DDL        = NUMFOR,
                                );

#=======================================================================================================#
#   Loading the re-normalised modeshapes in strain in air obtained by PAK, as 'displacement'-mode
#   shapes.
#=======================================================================================================#
C_PHI3D     = LIRE_RESU(        TYPE_RESU       = 'MODE_MECA',
                                FORMAT          = 'IDEAS',
                                MODELE          = MODELN3D,
                                UNITE           = 30,
                                NOM_CHAM        = 'DEPL',
                                MATR_A          = KASN3D,
                                MATR_B          = MASN3D,
                                FORMAT_IDEAS    = _F(   NOM_CHAM        = 'DEPL',
                                                        NUME_DATASET    = 55,
                                                        RECORD_6        = (1,2,2,8,2,3,),
                                                        POSI_ORDRE      = (7,4,),
                                                        POSI_NUME_MODE  = (7,4),
                                                        POSI_FREQ       = (8,1,),
                                                        POSI_MASS_GENE  = (8,2),
                                                        POSI_AMOR_GENE  = (8,3),
                                                        NOM_CMP         = ('DX','DY','DZ'),
                                                        ),
                                TOUT_ORDRE      = 'OUI',
                                );

C_PHI3D     = NORM_MODE(        reuse           = C_PHI3D,
                                MODE            = C_PHI3D,
                                NORME           = 'MASS_GENE',
                                );

#=======================================================================================================#
#   Loading the re-normalised modeshapes in displacemnt in air obtained by PAK
#=======================================================================================================#
PHI3D_B     = LIRE_RESU(        TYPE_RESU       = 'MODE_MECA',
                                FORMAT          = 'IDEAS',
                                MODELE          = MODELN3D,
                                UNITE           = 33,
                                NOM_CHAM        = 'DEPL',
                                MATR_A          = KASN3D,
                                MATR_B          = MASN3D,
                                FORMAT_IDEAS    = _F(   NOM_CHAM        = 'DEPL',
                                                        NUME_DATASET    = 55,
                                                        RECORD_6        = (1,2,2,8,2,3,),
                                                        POSI_ORDRE      = (7,4,),
                                                        POSI_NUME_MODE  = (7,4),
                                                        POSI_FREQ       = (8,1,),
                                                        POSI_MASS_GENE  = (8,2),
                                                        POSI_AMOR_GENE  = (8,3),
                                                        NOM_CMP         = ('DX','DY','DZ'),
                                                        ),
                                TOUT_ORDRE      = 'OUI',
                                );

PHI3D_B     = NORM_MODE(        reuse           = PHI3D_B,
```

```
                              MODE            = PHI3D_B,
                              NORME           = 'MASS_GENE',
                              );

Z_FREQ_W    = LIRE_RESU(      TYPE_RESU       = 'MODE_MECA',
                              FORMAT          = 'IDEAS',
                              MODELE          = MODELPAK,
                              UNITE           = 32,
                              NOM_CHAM        = 'DEPL',
                              MATR_A          = KASSPAK,
                              MATR_B          = MASSPAK,
                              FORMAT_IDEAS    = _F(   NOM_CHAM        = 'DEPL',
                                                      NUME_DATASET    = 55,
                                                      RECORD_6        = (1,2,2,8,2,3,),
                                                      POSI_ORDRE      = (7,4,),
                                                      POSI_NUME_MODE  = (7,4),
                                                      POSI_FREQ       = (8,1,),
                                                      POSI_MASS_GENE  = (8,2),
                                                      POSI_AMOR_GENE  = (8,3),
                                                      NOM_CMP         = ('DX','DY','DZ'),
                                                      ),
                              TOUT_ORDRE      = 'OUI',
                              );


#=======================================================================================================#
#   Loading the spectral density containing the starin measurements
#=======================================================================================================#

DYNAX       = LIRE_INTE_SPEC(   FORMAT          = 'IDEAS',
                                FORMAT_C        = 'REEL_IMAG',
                                UNITE           = 31,
                                NOM_RESU        = 'DEPL',
                                );
CALC_ESSAI( UNITE_RESU = 8,
            INTERACTIF = 'OUI',
            RESU_IDENTIFICATION = _F( TABLE = CO('EFFORT'))
#            RESULTATS  = (_F(TABLE = CO("TAB_01"),TYPE_TABLE='TABLE'),
#                          _F(TABLE = CO("TAB_02"),TYPE_TABLE='TABLE_FONCTION'),
#                          _F(TABLE = CO("TAB_03"),TYPE_TABLE='TABLE_FONCTION'),
#                          ),
            );

#IMPR_RESU( FORMAT = 'RESULTAT', UNITE = 8,
#           RESU = _F( RESULTAT = RELMSREX, TOUT_PARA = 'OUI',FORM_TABL='NON',))
#
#IMPR_RESU( UNITE = 31, FORMAT = 'IDEAS',
#           RESU = _F(RESULTAT = RELMS_BA))
#
#
#IMPR_RESU( UNITE = 32, FORMAT = 'MED',
#           RESU = _F(MAILLAGE = MAIN3D, RESULTAT = RESETLMX))

#=======================================================================================================#
FIN();
#=======================================================================================================#
```

```
#@ AJOUT modes Utilitai

#-----------------------------------------------------------------------------------------------
#   Extreacting the shapes from a RESULTAT
#   nom_result      = name of the RESULTAT you want to extract the nodes from.
#   name            = 'name' which will appear in the graph
#   type            = 'Strain' or 'Displacement' extracting the shapes in strain or displacement.
#   nodenumbers     = "'Groupname'" or string with 'nodegroup' name option to extract specific nodes.
#                     this double "' is there because it must be fed to the extraction as 'lala'
#   order           = and their specific order
#-----------------------------------------------------------------------------------------------
import Numeric
def extract_modes(nom_result, name, typ, nodenumbers, order):
    from Cata.cata import RECU_TABLE;
    from Cata.cata import CREA_CHAMP;
    from Cata.cata import DETRUIRE;
    from Cata.cata import _F;
    __freq      = RECU_TABLE(CO=nom_result,
                         NOM_PARA='FREQ',);
    afreq       = __freq.EXTR_TABLE().Array('NUME_ORDRE','FREQ');
    nr_modes    = afreq.shape[0];
    base        = range(nr_modes);
    DETRUIRE(   CONCEPT     = _F(NOM = __freq),INFO = 1);

    if typ      == 'strain':
        for i in range(nr_modes):
            __CHANO     = CREA_CHAMP(   TYPE_CHAM       = 'NOEU_EPSI_R',
                                        OPERATION       = 'EXTR',
                                        RESULTAT        = nom_result,
                                        NOM_CHAM        = 'EPSI_NOEU_DEPL',
                                        NUME_ORDRE      = i+1,
                                        );
            exec("champ_xx                = __CHANO.EXTR_COMP('EPXX', ["+nodenumbers+"], 1,);");
            exec("champ_yy                = __CHANO.EXTR_COMP('EPYY', ["+nodenumbers+"], 1,);");
            exec("champ_zz                = __CHANO.EXTR_COMP('EPZZ', ["+nodenumbers+"], 1,);");
            #exec("champ_xy                = __CHANO.EXTR_COMP('EPXY', ["+nodenumbers+"], 1,);");
            #exec("champ_yz                = __CHANO.EXTR_COMP('EPYZ', ["+nodenumbers+"], 1,);");
            #exec("champ_xz                = __CHANO.EXTR_COMP('EPXZ', ["+nodenumbers+"], 1,);");
            if  order == '':
                order               = range(len(champ_yy.valeurs.tolist()));
            vale                    = Numeric.choose(order,champ_xx.valeurs.tolist()).tolist()+Numeric…
            base[i]                 = Numeric.array(vale)
            DETRUIRE(   CONCEPT     = _F(NOM = __CHANO),INFO = 1);

    elif typ    == 'displacement':
        for i in range(nr_modes):
            __CHANO     = CREA_CHAMP(   TYPE_CHAM       = 'NOEU_DEPL_R',
                                        OPERATION       = 'EXTR',
                                        RESULTAT        = nom_result,
                                        NOM_CHAM        = 'DEPL',
                                        NUME_ORDRE      = i+1,
                                        );

            exec("champ_x                 = __CHANO.EXTR_COMP('DX', ["+nodenumbers+"], 1,);");
            exec("champ_y                 = __CHANO.EXTR_COMP('DY', ["+nodenumbers+"], 1,);");
            exec("champ_z                 = __CHANO.EXTR_COMP('DZ', ["+nodenumbers+"], 1,);");

            if (nodenumbers =='') & (nom_result.nom[-2:]=='2D'):
                exec("champ_rx                = __CHANO.EXTR_COMP('DRX', ["+nodenumbers+"], 1,);");
                exec("champ_ry                = __CHANO.EXTR_COMP('DRY', ["+nodenumbers+"], 1,);");
                exec("champ_rz                = __CHANO.EXTR_COMP('DRZ', ["+nodenumbers+"], 1,);");
            if  order == '':
                order               = range(len(champ_y.valeurs.tolist()));
            vale                    = Numeric.choose(order,champ_x.valeurs.tolist()).tolist()+Numeric.…
            if (nodenumbers =='') & (nom_result.nom[-2:]=='2D'):
                vale                    = vale + Numeric.choose(order,champ_rx.valeurs.tolist()).tolis…

            base[i]                 = Numeric.array(vale)

            DETRUIRE(   CONCEPT     = _F(NOM = __CHANO),INFO = 1);

    else:
```

```
            print 'No type selected!';

    result    = [base, afreq, typ, name]; #, numno];
    return result

#-----------------------------------------------------------------------------------------------
#   CALCULATING THE MAC NUMBER WITHOUT A MASS-MATRIX
#-----------------------------------------------------------------------------------------------
def mac_modes(base_1, base_2):
    import string
    if base_1[2]<>base_2[2]:
        print 'No MAC posible, one Base in Strain, one Base in Displacement!';return
    if base_1   == base_2:
        title   = 'Auto MAC';
        title   = title + ' of ' + base_1[3] + ' in '+ base_1[2];
    else:
        title   = 'Cross MAC';
        title   = title + ' of ' + base_1[3] + ' vs ' + base_2[3] + ' in '+ base_1[2];

    mode_1      = base_1[0];
    mode_2      = base_2[0];
    mac_entry   = range(len(mode_1)*len(mode_2));
    mac_string  = range(len(mode_1));
    Latex_2     = range(len(mode_1));
    monr_2      = '';
    freq_2      = '';
    Latex_0     = '\\multicolumn{2}{||r}{}& nr. ';
    Latex_1     = '\\multicolumn{1}{||r}{}& nr. & [freq.] ';
    i           = 0;
    for j in range(len(mode_2)):
        monr_2          = monr_2 + string.rjust(str(base_2[1][j][0])[0:(string.index(str(base_2[1][j][…
        freq_2          = freq_2 + string.rjust(str(base_2[1][j][1])[0:(string.index(str(base_2[1][j][…
        Latex_0         = Latex_0 + ' & ' + string.rjust(str(base_2[1][j][0])[0:(string.index(str(base…
        Latex_1         = Latex_1 + ' & [' + (str(round(base_2[1][j][1],2))+'00')[0:(string.index(str(…
        for i in range(len(mode_1)):
            mac_entry[(i*len(mode_2))+j]   = (abs(Numeric.innerproduct(mode_1[i],mode_2[j])))**2/(abs(…
            if j==0:
                mac_string[i]   = string.rjust((str(round(mac_entry[i*len(mode_2)+j],3))+'000')[0:5],7…
                Latex_2[i]      = ' & ' + string.rjust((str(round(mac_entry[i*len(mode_2)+j],3))+'000'…
            else:
                mac_string[i]   = mac_string[i] + string.rjust((str(round(mac_entry[i*len(mode_2)+j],3…
                Latex_2[i]      = Latex_2[i] + ' & ' + string.rjust((str(round(mac_entry[i*len(mode_2)…
    print '+--------------' + (base_2[1].shape[0]*8-1) * '-' + '+'
    print '| ' + title[0:(base_2[1].shape[0]*8+12)] + (base_2[1].shape[0]*8+12-len(title)) * ' ' + '|'
    print '+-------------+' + (base_2[1].shape[0]*8-1) * '-' + '+'
    print '|vs.          |' + base_2[3] + (base_2[1].shape[0]*8-len(base_2[3])-1) * ' ' + '|'
    print '|'  + base_1[3][0:13] + (13-len(base_1[3][0:13]))*' '+ '|' + (base_2[1].shape[0]*8-1) * ' '…
    print '+-------------+' + (base_2[1].shape[0]) * '-------+'
    print '|         freq|' + freq_2
    print '|    freq mode|' + monr_2
    print '+-------------+' + (base_2[1].shape[0]) * '-------+'
    for i in range(len(mode_1)):
        print '|' + string.rjust(str(base_1[1][i][1])[0:(string.index(str(base_1[1][i][1]),'.')+3)],8)…
    print '+-------------+' + (base_2[1].shape[0]) * '-------+'
    print
    print
    print '%% This a the tabel for LATEX, you can copy the code from this line until the next'
    print '%% ----------------------------------------------------------------------------------------------…
    print '\\begin{table}[htb!]'
    print '\\begin{center}'
    print '\\caption{' + title + '}'
    print '\\label{tab:' + title + '}'
    print '\\begin{tabular}{||l@{\!}|r@{\:} x{1.2cm}||'+len(mode_2)*'x{1.2cm}|'+'|}'
    print '\\multicolumn{'+str(len(mode_2)+3)+'}{c}{~}\\\[-1em]'
    print '\\hhline{|t:===:t:*{'+str(len(mode_2))+'}{=}'+':t|}'
    print '\\multicolumn{3}{||l||}{vs.} & \\multicolumn{' + str(len(mode_2)) + '}{c||}{' + base_2[3] +…
    print '\\hhline{||~~~||*{'+str(len(mode_2))+'}{-}||}'
    print Latex_0 + '\\tn'
    print Latex_1 + '\\tn'
    print '\\hhline{|:===::*{'+str(len(mode_2))+'}{=}:|}'
    for i in range(len(mode_1)-1):
        if i == 0:
```

```
            print '\\multirow{' + str(len(mode_1)) + '}{1.5cm}{' + base_1[3] + '}',
        print ' & ' + string.rjust(str(base_1[1][i][0])[0:(string.index(str(base_1[1][i][0]),'.'))],3)…
        print '\\hhline{||~--||*{'+str(len(mode_2))+'}{-}||}'
    i = i + 1;
    print ' & ' + string.rjust(str(base_1[1][i][0])[0:(string.index(str(base_1[1][i][0]),'.'))],3) + '…
    print '\\hhline{|b:===:b:*{'+str(len(mode_2))+'}{=}:b|}'
    print '\\end{tabular}'
    print '\\end{center}'
    print '\\end{table}'
    print '%% ------------------------------------------------------------------------------------------…
    print
    print
```

```
import Numeric
import string
import aster
def to_file_print(base, mesh):
    #print len(base[0][1])/3.
    #print len(base[0][1])/3
    #print len(base[0][1])
    freq_eau_LMS        = (54.399815,55.371490,99.025183,130.347128,135.832514,179.222010);
    amor_eau_LMS        = (1.96362e-02,1.90986e-02,7.55419e-03,6.86819e-03,5.87166e-03,4.41956e-03);
    freq_air_PAK        = (base[1][0][1],base[1][2][1],base[1][4][1],base[1][6][1]);
    print freq_air_PAK
    modal_fac_PAK       = (2*freq_air_PAK[0]/(freq_eau_LMS[0]+freq_eau_LMS[1]),freq_air_PAK[1]/freq_ea…
    freq_NEW            = ((freq_eau_LMS[0]+freq_eau_LMS[1])/2,(freq_eau_LMS[0]+freq_eau_LMS[1])/2,fre…
    amor_NEW            = ((amor_eau_LMS[0] + amor_eau_LMS[1])/2.,(amor_eau_LMS[0] + amor_eau_LMS[1])/…
    toto = str(mesh+'.NOMNOE                        ')[0:32];
    noeudsnr = aster.getvectjev(toto);
    print '--------------------------------Copy from this line to the next--------------------------…
    for i in range(len(base[0])):
        print '    -1'
        print '    55'
        print 'NONE'
        print 'GOP:SOP'
        print 'Frequency = '+str(round(base[1][i][1],1))
        print 'Damping = '+str(round(base[1][i][1],1))
        print 'NONE'
        print '         1         2         2         8         2         3'
        print '         2         4         0      ' + string.rjust((str(base[1][i][0])[:-2]),4)
        print string.rjust((str("%E" % freq_NEW[i])[:-5] + str("%E" % freq_NEW[i])[-4:]),13) + string.…
        for j in range(len(base[0][i])/3):
            print string.rjust(str(int(noeudsnr[j][1:])),10)
            print string.rjust((str("%E" % base[0][i][j])[:-5] + str("%E" % base[0][i][j])[-4:]),13) +…
        print '    -1'
    print '-----------------------------------------the next--------------------------------------…
    return
```

Bibliography

[1] C. Bodel, *Identification d'efforts fluides appliqués à un tube guide de grappe commande, modèle EPR. Méthodologie et résultat*, EDF R&D, May 2008.

[2] David C. Lay, *Linear Algebra and Its Applications*, Addison Wesley Publishing Company, 1994.

[3] Singiresu S. Rao, *Mechanical Vibrations*, Fourth edition in S.I. units, Pearson Education, April 2003.

[4] Daniel J. Rixen, *Mechanical Analyses for Engineering*, TU Delft, Faculty of 3ME, Lecture Notes with the course WB1450 Mechanical Analyses for Engineering, 2008.

[5] Daniel J. Rixen, *Numerical methods in Engineering Dynamics*, TU Delft, Faculty of 3ME, Lecture Notes with the course WB1416 Numerical methods in Engineering Dynamics, 2007.

[6] P.Th.L.M. van Woerkom, *Notes on Linear Vibration Theory*, TU Delft, Faculty of 3ME, Lecture Notes with the course WBMT1216 Dynamics 2, January 2003.

[7] S. Graham Kelly, *Fundamentals of Mechanical Vibration*, Second edition, McGraw-Hill, March 2000.

[8] Robert D. Blevins, *Flow-Induced Vibrations*, Krieger Publishing Company, 1993.

[9] Ward Heylen, Stefan Lammens, Paul Sas, *Modal Analysis Theory and Testing* KU Leuven, Faculty of Engineering, Department of Mechanical Engineering, 1999.

[10] C.G. Rodriguez, E. Egusquiza, X. Escaler, Q.W. Liang, F. Avellan, *Experimental investigation of added mass effects on a Francis turbine runner in still water*, Elsevier, Journal of Fluids and Structures 22, 2006.

[11] Yi Liu, W. Steve Shepard Jr., *Dynamics force identification based on enhanced least squares and total least-squares schemes in the frequency domain*, Elsevier, Journal of Sound and Vibration 282, 2005.

[12] John C. O'Callahan, Peter Avitabile, Richard K. Leung, *Development Of Mass and Stiffness Matrices For An Analytical Model Using Experimental Modal Data*, Second International Modal Analysis Conference, Orlando, Florida, Februari 1984

[13] John C. O'Callahan, Richard K. Leung, *Optimization Of Mass And Stiffness Matrices Using A Generalized Inverse Technique On The Measured Modes*, Third International Modal Analysis Conference, Orlando, Florida, January 1985

[14] S. Petit Jean, J.C. Chambrin, *EPR CRGA Optimication Test Report (Magaly Phase 1 tests)*, second edition, FRAMATOME ANP, may 2005.

[15] J.W.M. de Jong, *Vibrations of a Graphite Rod*, EDF R&D, April 2009.

[16] `http://www.wikipedia.org`

[17] `http://www.sdtools.com`